УНІВЕРСИТЕТ ДЕРЖАВНОЇ ФІСКАЛЬНОЇ СЛУЖБИ УКРАЇНИ

СЕРІЯ «НА ДОПОМОГУ СТУДЕНТУ УДФСУ»

Серію «На допомогу студенту УДФСУ» засновано 2016 року.

Постіл С. Д.

CASE-ТЕХНОЛОГІЇ. МІЖДИСЦИПЛІНАРНЕ ІНФОРМАЦІЙНЕ МОДЕЛЮВАННЯ

Навчальний посібник

Ірпінь
2018

УДК 004.9(075.8)
ББК 32.973я73
П 63

Рекомендовано до друку Вченою радою
Університету державної фіскальної служби України
(протокол № 17 від 21 грудня 2017 року)

Рецензенти:

Ріппа С. П., д.е.н., професор, директор ННІ ІТМ Університету державної фіскальної служби України;

Файнзільберг Л. С., д.т.н., професор, г.н.с. Міжнародного наук.-навч. центру інформаційних технологій та систем НАН та МОН України.

Постіл С. Д.

П 63 CASE-технології. Міждисциплінарне інформаційне моделювання : навчальний посібник / С. Д. Постіл. – Ірпінь : Університет ДФС України, 2018. – 304 с. – (Серія «На допомогу студенту УДФСУ», т. 18).

ISBN 978-966-337-471-0

У навчальному посібнику зібрано і систематизовано регламентні проекти і документи щодо життєвого циклу програмного продукту та підготовки фахівців з інформаційних технологій. Значна увага приділена опису концептуальних основ інженерії програмного продукту (SWEBOK), проекту Computing Curricula щодо викладання інформаційних технологій, стандартам освіти України в ІТ-області, а також стандартам проектування, реалізації та експлуатації продуктів ІТ-індустрії.

Його перевагою є наявність теоретичного матеріалу з методологій інформаційного моделювання, опису CASE-засобів структурного аналізу та проектування для їх освоєння, а також лабораторні роботи з реалізації «Наскрізного індивідуального проекту» на основі індивідуального об'єкта дослідження при вивченні різних навчальних дисциплін.

Призначений для студентів та викладачів, які здійснюють підготовку за спеціальностями на базі інформаційного моделювання з використанням CASE-засобів структурного аналізу та проектування.

УДК 004.9(075.8)
ББК 32.973я73

© Постіл С. Д., 2018
© Університет державної фіскальної служби України, 2018

ISBN 978-966-337-471-0

Зміст

СПИСОК СКОРОЧЕНЬ 7

ВСТУП 10

ЧАСТИНА 1. ІНФОРМАЦІЙНЕ МОДЕЛЮВАННЯ 12
РОЗДІЛ 1. Освітні та професійні стандарти в галузі ІТ 12
1.1. Взаємодія ІТ-освіти та ІТ-індустрії 12
1.2. Міжнародні стандарти освіти в ІТ-області 14
1.3. Стандарти освіти України в ІТ-області 18
1.4. Міжнародний та національний стандарти проектування, реалізації та експлуатації продуктів ІТ-індустрії 20
1.5. Лабораторна робота 1. Освітні та професійні стандарти в галузі інформаційних технологій 21
РОЗДІЛ 2. Інженерія програмного продукту 24
2.1. Формування вимог до програмного продукту 25
2.2. Проектування програмного забезпечення 28
2.3. Реалізація програмного забезпечення 31
2.4. Тестування програмного забезпечення 34
2.5. Супровід програмного забезпечення 37
2.6. Інструменти інженерії ПЗ 40
2.7. Лабораторна робота 2. Методологія формування вимог на прикладі ІС «Компетентність студента вишу» 43
2.8. Лабораторна робота 3. Аналіз предметної області та формування вимог до ІС індивідуального об'єкта дослідження . 53
РОЗДІЛ 3. Міждисциплінарне інформаційне моделювання 56
3.1. Інформаційне моделювання 56
3.2. Міждисциплінарне ІМ у навчальному процесі 58
3.3. Архітектура моделей даних інформаційної системи 63
3.4. Проектування баз даних інформаційних систем 67
3.5. Лабораторна робота 4. Створення бази даних та інтерфейсу доступу мовними та інструментальними засобами .. 77
3.6. Робота в MS Access 2016 зі створення БД 78

ЧАСТИНА 2. МЕТОДОЛОГІЇ СТРУКТУРНОГО АНАЛІЗУ І ПРОЕКТУВАННЯ 91
РОЗДІЛ 4. Моделювання за допомогою CASE-засобу AllFussion PM пакета AllFussion Modeling Suite Computer Associates 91
4.1. Теоретичні аспекти методології IDEF0 91
4.2. Технологія побудови моделі за допомогою CASE-засобу AllFussion PM на прикладі системи «Компетентність студента вишу» 100
4.3. Лабораторна робота 5. Методологія IDEF0 зі створення функціональної моделі бізнес-процесів індивідуального об'єкта дослідження та розвиток «Індивідуального проекту» 108
4.4. Теоретичні аспекти методології діаграм потоків даних (DFD) 112
4.5. Технологія побудови моделі на основі DFD за допомогою AllFussion Process Modeller на прикладі системи «Компетентність студента вишу» 116
4.6. Лабораторна робота 6. Методологія DFD зі створення діаграм потоків даних індивідуального об'єкта дослідження та розвиток системи «Індивідуальний проект» 123
4.7. Теоретичні аспекти методології IDEF3 125
4.8. Технологія реалізації методології IDEF3 на прикладі інформаційної системи «Компетентність студента вишу» 129
4.9. Лабораторна робота 7. Методологія IDEF3 з опису взаємозв'язків між процесами індивідуального об'єкта та розвиток системи «Індивідуальний проект» 137
4.10. Технології створення звітів у середовищі CASE-засобу AllFussion PM 139
4.11. Лабораторна робота 8. Створення звітів у середовищі CASE-засобу AllFussion PM на основі системи «Індивідуальний проект» 146
РОЗДІЛ 5. Моделювання за допомогою CASE-засобу AllFusion ERwin DM пакета AllFussion Modeling Suite CA 148
5.1. Теоретичні аспекти методології IDEF1X 148

5.2. Технологія реалізації методології IDEF1X у середовищі CASE-засобу AllFussion ERwin DM 154
5.3. Лабораторна робота 9. Методологія IDEF1X зі створення проекту БД за допомогою CASE-засобу AllFusion ERwin DM.. 175
5.4. Лабораторна робота 10. Впровадження методології IDEF1X створенням БД «Наскрізний індивідуальний проект» за допомогою CASE-засобу AllFusion ERwin DM..... 176
5.5. Методологія вимірного моделювання сховищ даних 179
5.6. Лабораторна робота 11. Розробка вимірної моделі СД індивідуального об'єкта дослідження і розвиток «Наскрізного індивідуального проекту» 192
5.7. Технологія реалізації вимірної моделі сховища даних за допомогою CASE-засобу AllFusion ERwin DM 194
5.8. Лабораторна робота 12. Створення проекту вимірної моделі сховища даних на задачі-прототипі за допомогою CASE-засобу Erwin DM пакета AllFussion Modeling Suite 208
5.9. Лабораторна робота 13. Створення проекту у середовищі CASE-засобу ERwin DM вимірної моделі сховища даних індивідуального об'єкта дослідження і розвиток «Наскрізного індивідуального проекту» 210
РОЗДІЛ 6. Аналіз, удосконалення та інтеграція моделей за допомогою CASE-пакета AllFussion Modeling Suite CA 214
6.1. Структура та принцип роботи CASE-засобу AllFusion Data Model Validator (DMV) пакета AllFussion Modeling Suite CA 214
6.2. Особливості помилок і недоліків моделювання колонок, ключів, індексів, обмежень, зв'язків таблиць БД 216
6.3. Технологія пошуку і виправлення помилок моделювання БД за допомогою AllFusion DMV 230
6.4. Навчальна опція з моделювання даних в AllFusion DMV .. 238
6.5. Порівняння моделі даних і об'єднання DDL-скриптів 239
6.6. Лабораторна робота 14. Аналіз і вдосконалення моделі даних «Наскрізний індивідуальний проект»

в CASE-засобі Data Modeler Valadator (DMV) пакета AllFussion Modeling Suite 240
6.7. Інтегративні можливості CASE-пакета структурного аналізу і проектування AllFussion Modeling Suite CA 242
6.8. Експорт моделі даних з AllFusion ERwin DM в AllFusion PM 248
6.9. Створення сутностей і атрибутів в AllFusion PM та їх експорт в AllFusion ERwin DM 250
6.10. Лабораторна робота 15. Інтеграція об'єктів моделі процесів і моделі даних реалізацією механізму двонаправленого зв'язку в AllFusion PM – AllFusion ERwin DM 254

ТЕСТИ 257

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 283

ТЕРМІНОЛОГІЧНИЙ СЛОВНИК 285

ДОДАТКИ 288

Список скорочень

ADL (Architecture Description Language) – мова опису архітектури
ANSI (AMERICAN NATIONAL STANDARD INSTITUTE) – Національний інститут стандартизації США
CASE (Computer Assisted System/Software Enginieering) – технології автоматизованого проектування систем/ПЗ
CC (COMPUTING CURRICULA) – навчальні програми комп'ютерних наук
CDIO (Conceive-Design-Implement-Operate) – концепція освіти в контексті життєвого циклу продукту
DDE (Dynamic Data Exchange) – протокол динамічного обміну даними
DDL (Data Definition Language) – мова опису даних
DFD (Data Flow Diagrams) – діаграми потоків даних
Dimensional – розмірна модель сховища даних
DM (Data Modeller) – підсистема моделювання даних
DMV (AllFusion Data Model Validator) – система пошуку і виправлення помилок моделі даних
DSS (Decision Support Systems) – системи підтримки прийняття рішень
DW (Data WareHouse) – сховище даних
ERD (Entity-Relations Diagrams) – методологія «сутність-зв'язок»
ERP (Enterprise Resource Planning) – системи планування ресурсів
IDEF (Integrated Computer-Aided Manufacturing) – сімейство методологій
IDEF0 – методологія функціонального моделювання
IDEF1X – методологія моделювання баз даних
IDEF3 – методологія опису послідовності виконання і сценаріїв процесів
IDL (Interface Description Language) – мова опису інтерфейсу
ISO (International Organization for Standardization) – міжнародний стандарт
NQA (Malcolm Baldrige National Quality Award) – національний стандарт якості США

ODBC (Open Data Base Connectivity) – система драйверів доступу до БД
OLAP (ON-Line Analytical Processing) – системи аналітичної обробки даних
OLE (Object Linking and Embedded) технологія впровадження/зв'язування об'єктів різних додатків
OLTP (On-Line Transaction Processing) – системи оперативної обробки даних
PM (Process Modeller) – підсистема моделювання процесів
RUP (Rational Unified Process) – раціональний уніфікований процес
SQL (Structured Query Language) – мова структурованих запитів
SPARK (Standards Planning and Requirements Committee) – комітет планування стандарту і норм
SWEBOK (SOFTWARE ENGINEERING BODY OF KNOWLEDGE) – об'єднання знань з програмної інженерії
UML (Unified Modeling Language) – уніфікована мова моделювання
VBA (Visual BASIC for Application) – мова програмування
WWW (World Wide Web або просто Web) – найбільш потужна мережна інформаційна система в світі
АІС – автоматизована інформаційна система
АМПД – архітектури моделей подання даних
БД – база даних
ДСТУ – державний стандарт України
ЕОМ – електронна обчислювальна машина
ЕОС – електронно-обчислювальна система
ЕТ – електронна таблиця
ЖЦ – життєвий цикл
ІБ – інформаційна база
ІЛМ – інформаційно-логічна модель
ІМ – інформаційне моделювання
ІПС – інформаційно-пошукова система
ІС – інформаційна система
ІТ – інформаційна технологія
НДІ – нормативно-довідкова інформація

НДР – науково-дослідна робота
БКНФ – нормальна форма Бойса-Кодда
1НФ – перша нормальна форма
2НФ – друга нормальна форма
3НФ – третя нормальна форма
4НФ – четверта нормальна форма
5НФ – п'ята нормальна форма
ОС – операційна система
ПЗ – програмне забезпечення
ПК – персональний комп'ютер
ПО – предметна область
ПП – програмний продукт
ПС – предметна сфера
РСОД – розподілена система обробки даних
СД (data warehouses) – сховища даних
ССВО – система стандартів вищої освіти
СППП (Decision Support System) – система підтримки прийняття рішень
СУБД – система управління базами даних
ТЕІ – техніко-економічна інформація
ТЗ – технічне завдання

ВСТУП

Вітчизняна інженерна освіта у сфері інформаційних технологій (ІТ) передбачає використання нових технологій освіти (концепцію CDIO) та сучасних педагогічних технологій: змішана очно-дистанційна; інтегративна (міждисциплінарні і трансдисциплінарні підходи, методи синергетики); проектна (практико-орієнтовані та проблемно-орієнтовані підходи, метод проблемного навчання) та інші.

За міжнародною концепцією CDIO (Conceive-Design-Implement-Operate) студентам надається освіта, яка формує інженерні основи, викладені в контексті життєвого циклу реальних систем, процесів і продуктів: «Задумай (Ідея)» – «Спроектуй (Проект)» – «Реалізуй (Реалізація)» – «Керуй (Управління, Експлуатація)».

Враховуючи складність проблеми освоєння усіх етапів життєвого циклу програмного продукту (ІТ-проекту) і значний обсяг робіт, створення повномасштабного ІТ-проекту та досягнення навчальної мети з підготовки компетентного фахівця в межах однієї дисципліни є проблематичним. Тому виникає потреба в організації інтегративного підходу з реалізацією горизонтальних і вертикальних міждисциплінарних зв'язків під час викладання декількох комп'ютерних дисциплін. У результаті цього студенти виконують різні ролі з реалізації всіх етапів життєвого циклу проекту (від ідеї до експлуатації) у процесі створення «Наскрізного індивідуального проекту» на основі наскрізного індивідуального об'єкта дослідження.

Нині широко використовуються CASE-системи (Computer-Aided System/Software Engineering), які дозволяють автоматизувати процес розробки ІТ-проекту.

У першій частині навчального посібника розглянуто: 1) базові основи інформаційного моделювання; концепцію навчальної технології зі створення «Наскрізного індивідуального проекту» в процесі міждисциплінарного інформаційного моделювання; 2) освітні та професійні стандарти в галузі інформаційних

технологій; 3) теоретичні основи інженерії програмного продукту; 4) архітектуру моделей даних інформаційної системи.

У другій частині посібника розглянуто: 1) теоретичні аспекти методологій інформаційного моделювання (IDEF0, DFD, IDEF3, IDEF1X, Dimensional); 2) CASE-засоби структурного аналізу та проектування для освоєння даних методологій моделювання; 3) аналіз, удосконалення та інтеграція моделей за допомогою CASE-пакета AllFussion Modeling Suite Computer Associates; 4) комплекси лабораторних робіт, виконання яких здійснюється в два етапи: ознайомлення з необхідною методологією, інструментальним засобом і технологією на прототипі; закріплення отриманих знань, умінь та навичок у процесі реалізації «Наскрізного індивідуального проекту» студента на основі індивідуального наскрізного об'єкта дослідження.

Навчальний посібник призначений для студентів, які прагнуть освоїти методології інформаційного моделювання з використанням CASE-засобів структурного аналізу та проектування. Навчальний посібник також може бути корисним для викладачів відповідних дисциплін.

ЧАСТИНА 1
ІНФОРМАЦІЙНЕ МОДЕЛЮВАННЯ

РОЗДІЛ 1
ОСВІТНІ ТА ПРОФЕСІЙНІ СТАНДАРТИ В ГАЛУЗІ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

1.1. Взаємодія стандартів ІТ-освіти та ІТ-індустрії

Загальна інформатизація – незаперечний факт нашого часу. Область застосування інформаційних технологій (ІТ) продовжує стрімко розширюватися, що викликає зростання потреби в ІТ-фахівцях, причому все більше фахівців потрібно не тільки компаніям-розробникам програмного забезпечення, обчислювальної техніки і нових інформаційних технологій, але і компаніям, чия основна діяльність з розробкою ІТ-продуктів і послуг прямо не пов'язана, з чого випливає необхідність диверсифікації підготовки професіоналів цієї області.

Дослідження аналітичних даних, що відображають динаміку економічного зростання провідних країн і показники розвитку в них вищої освіти, виявило нове важливе явище – посилення професійного компонента у підготовці фахівців у вищій школі (виш) на всіх рівнях – від бакалаврату до докторантури. Професіоналізація завжди була домінуючою функцією вітчизняної освіти. Система стандартів вищої освіти України значною мірою повинна грунтуватися на нормативних документах сфери праці.

Вплив нормативних документів сфери праці на формування змісту складових системи стандартів вищої освіти України відображено на рис. 1.1.

Список затребуваних професій широкий – від розробників до інженерів з обслуговування програмних систем, системних архітекторів, адміністраторів мереж і баз даних, фахівців з впровадження, обслуговування та підтримки корпоративного ПЗ, дизайнерів інтерфейсу та ін.

Динамічність та інноваційність ІТ-галузі вимагає постійного оновлення навчальних планів і змісту навчальних програм із ІТ-дисциплін. Існуючі освітні стандарти надають можливість гнучко формувати основні освітні програми (навчальні плани, навчальні програми та ін.), актуалізувати їх зміст за необхідності за рахунок відповідних розділів навчального плану.

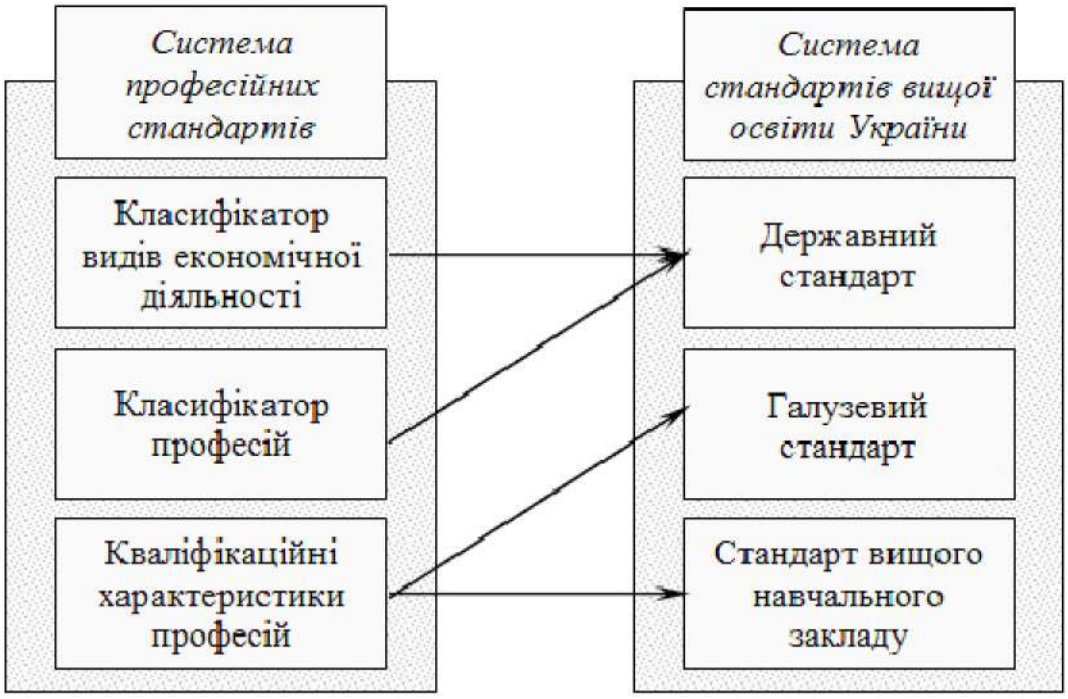


Рис. 1.1. Взаємодія професійних та освітніх стандартів

На сьогодні рішення проблеми визначення змісту профільної підготовки ІТ-фахівця, затребуваного на ринку праці, диктує необхідність визначення набору професійних компетентностей і механізмів їх формування за кваліфікаціями, що, у свою чергу, вимагає наявності високих особистісно-професійних компетентностей щодо визначення змісту підготовки в ІТ-області у професорсько-викладацького складу кафедр.

Розробка типової навчальної програми з дисципліни – складне завдання, яке не має чіткого алгоритму рішення. Нормативні документи лише наказують її структуру і вказують, що при розробці навчальних програм необхідно виходити з того, що їх зміст має забезпечити формування у випускників закладів вищої освіти знань, умінь і навичок відповідно до освітніх стандартів.

Вихідними документами для розробки типової навчальної програми є освітній стандарт і типовий навчальний план за спеціальністю. Семантична ж сторона (зміст навчального матеріалу, його розбиття на розділи, теми, їх будова, розподіл аудиторних годин між ними та самостійної роботи) покладається на учасника (або колектив авторів) програми. Якщо навчальна програма розробляється з нової дисципліни або для нової цільової аудиторії (наприклад, студентів практико-орієнтованої магістратури), то складність поставлених завдань зростає багаторазово.

1.2. Міжнародні стандарти освіти в IT-області

Перша спроба підходу до стандартизації навчання в IT-області реалізована у створеному в 1968 році Асоціацією з обчислювальної техніки ACM (The Association for Computing Machinery) звіті – Навчальний план-68. У ньому були приведені детальні рекомендації для складання навчальних програм з інформатики, а також набір описів курсів і величезна бібліографія для кожної тематичної області.

У 1970-х роках ACM і Комп'ютерна спільнота інституту інженерів з електротехніки та електроніки (IEEE-CS) (The Computer Society of the Institute for Electrical and Electronic Engineers) створили комітети з розробки виправлених і уточнених навчальних планів з інформатики. У 1977 році звіт про програми з комп'ютерної науки і проектування випустило комп'ютерне співтовариство – IEEE Computer Society (IEEE-CS).

Звіт ACM Навчальний план-68 був замінений на розширену версію Навчальний план-78, яка мала значний вплив на освіту у сфері інформатики. Зокрема, в Навчальному плані-78 був запропонований ряд стандартних курсів, що описують основні знання інформатики.

Реагуючи на швидкий розвиток комп'ютерної науки, IEEE-CS оновило свій звіт у 1983 році (EAB 83).

Наприкінці 1980-х років IEEE-CS і ACM об'єднали свої сили, і в результаті спільних дій був опублікований документ Computing Curricula (далі – CC 1991). Звіт CC 1991 був більш повним, ніж його попередники, але його підхід був іншим.

На відміну від Навчального плану-78 та звіту IEEE-CS за 1983 рік, кожен з яких був сконцентрований на визначенні змісту стандартного набору курсів, CC 1991 розділив сукупність знань з інформатики на окремі розділи знань.

Кожен розділ знань у CC 1991 відповідає деякій темі, яка повинна бути викладена в університетських курсах навчання, але при цьому навчальним закладам надана значна свобода в угрупуванні розділів знань у власних планах викладання згідно зі своїми специфічними потребами.

Наприкінці 1990-х років стало зрозуміло, що область знань, пов'язана з інформаційними технологіями, досить сильно розрослася і її важко повністю висвітлити в межах одного напряму. Були виділені п'ять напрямів підготовки фахівців:

1. Computer Science («Інформатика»);
2. Computer Engineering («Комп'ютерна інженерія»);
3. Information Systems («Інформаційні системи»);
4. Software Engineering («Програмна інженерія»);
5. Information Technology («Інформаційні технології»).

Починаючи з 2001 року, що кожного з напрямів розроблені документи, що містять набір рекомендацій для викладання спеціальностей за напрямами. Динаміка їх розроблення та оновлення подана в табл. 1.1.

Таблиця 1.1

Динаміка розроблення та оновлення документів щодо рекомендацій для викладання спеціальностей за напрямами

Найменування розробки	Розробники	Рік в.
Узагальнений звіт за напрямами		
Computing Curricula 2005: The Overview Report describes general principles and commonalities among all of the specific disciplines described in the other four volumes. Комп'ютинг 2005. Узагальнення керівних принципів за розділами: обчислювальна техніка (на базі звіту 2004), інформатика (2001), інформаційні системи (2002), інформаційні технології (2006),	Спільний проект освітньої ради ACM (*ACM Education board*), Ради з освітньої діяльності комп'ютерного співтовариства IEEE (*IEEE Computer Society Educational Activities Board*) та Асоціації з інформа-	2005

програмна інженерія (2004)	ційних систем (*The Association for Information Systems*)	
Інформатика		
Computer Science Curriculum 2008: *An Interim Revision of CS 2001.* Оновлена версія 2001 року _____ Computing Curricula 2001 *Computer Science.* Рекомендації з викладання інформатики в університетах (офіційне перекладне видання російською мовою, 2002)	Спільний проект освітньої ради ACM та Ради з освітньої діяльності комп'ютерного співтовариства IEEE	2008
Інформаційні системи		
Information Systems 2010: Curriculum Guidelines for Undergraduate Degree Programs in Information Systems. (Інформаційні системи 2010: Рекомендації з викладання спеціальності Інформаційні системи в університетах – рівень бакалаврату)	Спільний проект освітньої ради ACM та Ради з освітньої діяльності комп'ютерного співтовариства IEEE	2010
MSIS 2006: Model Curriculum and Guidelines for Graduate Degree Programs in Information Systems (Рекомендації з викладання спеціальності Інформаційні системи в університетах – рівень магістратури)	Спільний проект освітньої ради ACM та Асоціації професіоналів в області інформаційних технологій – *AITP (Association of Information Technology Professionals)*	2006
Information Systems 2002: Curriculum and Guidelines for Undergraduate Degree Programs in Information Systems. (Інформаційні системи 2002: Рекомендації з викладання спеціальності Інформаційні системи в університетах – рівень бакалаврату)	Спільний проект освітньої ради ACM, Асоціації з інформаційних систем – AIS (*Association for Information Systems*) та Асоціації професіоналів в області інформаційних технологій AITP	2002
Програмна інженерія		
Graduate Software Engineering 2009 (GSwE 2009);	Спільний проект освітньої ради ACM	2009

Curriculum guidelines for graduate degree programs in Software Engineering. (Рекомендації з викладання програмної інженерії – рівень магістратури)	та Ради з освітньої діяльності комп'ютерного співтовариства IEEE	
Software Engineering 2004: Curriculum Guidelines For Undergraduate Degree Programs in Software Engineering (Рекомендації з викладання програмної інженерії – рівень бакалаврату (*офіційний переклад*)		2004
Обчислювальна техніка		
Computer Engineering 2004: Curriculum Guidelines for Undergraduate Degree Programs in Computer Engineering (Обчислювальна техніка 2004: Рекомендації з викладання спеціальності Обчислювальна техніка – рівень бакалаврату)	Спільний проект освітньої ради ACM та Ради з освітньої діяльності комп'ютерного співтовариства IEEE	2004
Інформаційні технології		
Information Technology 2008 Curriculum Guidelines for Undergraduate Degree Programs in Information Technology (Інформаційні технології 2008 Рекомендації з викладання спеціальності інформаційні технології – рівень бакалаврату)	Спільний проект освітньої ради ACM та Ради з освітньої діяльності комп'ютерного співтовариства IEEE	2008

Одним із базових проектів зі систематизації накопичених знань в області програмної інженерії є SWEBOK (Software Engineering Body of Knowledge). Цей документ підготовлений співтовариством IEEE Computer Society.

Призначення SWEBOK полягає в об'єднанні знань з розробки програмного забезпечення і їх структуруванні. Він покликаний вирішувати такі завдання при підготовці фахівців у галузі ПЗ: визначити необхідний набір знань і рекомендовані практики; визначити етичні та професійні стандарти; визначити навчальну програму для студентів, аспірантів, що продовжують навчання.

З програмної інженерії в документі SWEBOK 2004 виділені 10 областей знань:

1. Software Requirements – вимоги до ПЗ;
2. Software Design – проектування ПЗ;
3. Software Construction – конструювання ПЗ;
4. Software Testing – тестування ПЗ;
5. Software Maintenance – супровід ПЗ;
6. Software Configuration Management – управління конфігурацією;
7. Software Engineering Management – управління ІТ проектом;
8. Software Engineering Process – процес програмної інженерії;
9. Software Quality – якість ПЗ;
10. Software Engineering Tools and Methods – методи та інструменти.

У кожній з областей за ієрархічним принципом виділяються основні розділи і, можливо, більш дрібні компоненти. Кожній галузі знань присвячена окрема глава, структурована за ієрархічним принципом на розділи і рубрики, і завершується найширшими списками. Така організація документа, орієнтація його на існуючі стандарти в області ПЗ, глосарій можуть надати неоціненну допомогу при розробці навчальних програм не тільки в області розробки ПЗ, а й ІТ у цілому.

Використання при створенні навчально-програмної документації для підготовки ІТ-фахівців світового досвіду, систематизованого у вищерозглянутих документах, дозволяє інтегрувати вітчизняну ІТ-освіту в міжнародний контекст і забезпечує зв'язок змісту з реальними потребами ІТ-галузі та економіки в цілому як у країні, так і за кордоном.

1.3. Стандарти освіти України в ІТ-області

Галузеві стандарти України розроблені в 2015 році робочою групою, до складу якої увійшли представники провідних технічних вишів та ІТ-компаній. Вони відображають сучасні вимоги до фахівця з розробки програмного забезпечення, визначено перелік знань, умінь і навичок, якими повинен він володіти.

Базова підготовка розробника програмного забезпечення повинна включати в себе математичну підготовку, знання в області програмної інженерії, програмування, технологій і процесів

розробки програмного забезпечення, знання про бази даних і роботу з ними, знання про комп'ютерні мережі, засоби зв'язку, адміністрування програмного забезпечення, знання найбільш поширених операційних систем і особливостей роботи з ними.

На основі загальноприйнятих галузевих стандартів розробляються і коректуються навчальні програми у виші за спеціальностями, пов'язаними з програмуванням.

Професійні стандарти регулярно переглядаються та оновлюються, щоб відповідати новітнім трендам розробки ПЗ. На місце старих технологій приходять нові. Завдання фахівця – підвищувати свій професіоналізм, безперервно вдосконалюючись, навчаючись і підтверджуючи свої навички відповідними сертифікатами.

«Рада з конкурентоспроможності індустрії комунікаційно-інформаційних технологій України» опублікувала у себе на сайті кілька розроблених учасниками ради документів, які призначені як для студентів ІТ-спеціальностей, так і для академічного середовища. Сенс документів досить простий – безпосередні учасники ринку розробки ПЗ (GlobalLogic, «EPAM України», «1С Україна») описали, які вимоги чекають випускників університетів у реальному житті.

Для цього було розроблено проекти 5 стандартів, керуючись якими можна приблизно зрозуміти, чого саме чекають від молодої людини роботодавці. Це такі стандарти:

1. Спеціаліст з розробки програмного забезпечення.
2. Керівник проектів в області інформаційних технологій.
3. Менеджер продуктів в області інформаційних технологій.
4. Спеціаліст з інформаційних систем.
5. Спеціаліст з інформаційних ресурсів.

Структура документів достатньо проста. Крім нормативних вимог, у кожному з них наводиться таблиця, що складається з трьох стовпців: «Трудова функція», «Знання», «Уміння».

1.4. Міжнародні та національні стандарти проектування, реалізації та експлуатації продуктів ІТ-індустрії

Міжнародний стандарт розроблений міжнародною організацією стандартизації. Найвідомішою з них є International Organization for Standardization (ISO).

Міжнародний стандарт можна застосовувати прямо чи модифікувати для кращої відповідності місцевим умовам. Результатом модифікації міжнародних стандартів є створення еквівалентних державних стандартів.

Міжнародні стандарти допомагають долати технічні бар'єри в міжнародній торгівлі, що спричиняються відмінностями стандартів, розроблених окремо кожною нацією.

Об'єднання міжнародних ринків, підвищення вимог до якості і жорстка конкуренція привело до появи двох рівнобіжних стандартів якості – ISO 9000, більш поширеного в Європі, і національного стандарту якості NQA (Malcolm Baldrige National Quality Award), досить популярного в США. Обидва стандарти містять приблизно ті самі вимоги, торкаються тих самих критеріїв якості і, отже, майже еквівалентні, а тому компанії можуть обирати будь-який із них виходячи з власних міркувань. У дійсності між двома згаданими концепціями існує принципова різниця в орієнтації, цілях і внутрішньому змісті (на думку експертів, ISO 9000 перекриває менше четверті критеріїв, що увійшли у стандарт NQA).

Основна мета стандарту NQA – навчити організації створювати конкурентоспроможності продукти.

Основна мета ISO 9000 – примусити їх жорстко відповідати описам технологічних процесів, які вони самі ж і складають. Реєстрація в ISO свідчить про те, що компанія підтримує «документовану практику». Однак якість кінцевого продукту, вимоги ринку і технологічний рівень самих виробничих процесів виявляються поза інтересами ISO.

Стандарт NQA орієнтований на види діяльності, що сприяють підвищенню конкурентоспроможності компаній і передбачають для досягнення цього різні способи: за рахунок орієнтації на ринок і клієнтів, націленості на кінцевий результат, постійного удосконалювання виробничих процесів, тісної прив'язки до загальної стра-

тегії бізнесу, інтеграції процесів на основі аналізу, підвищення кваліфікації персоналу, розширення інформаційних зв'язків.

Існуючі розходження між стандартами ISO 9000 і NQA не означають, що одна система виключає іншу. Навпаки, багато провідних міжнародних компаній успішно їх поєднують.

Національний стандарт України представляє собою тотожний переклад ISO 9000:2005 Quality management systems – Fundamentals and vocabulary (Системи управління якістю. Основні положення та словник термінів).

Технічний комітет, відповідальний за цей стандарт, – ТК 93 «Системи управління якістю, довкіллям та безпечністю харчових продуктів» (підкомітет ПК 93/1 «Системи управління якістю»).

Роботу комітетів курують найбільші організації – члени ISO: AFNOR, ANSI, BSI, CSBTS, DIN, SIS та ін. Прийняття стандарту можливо тільки після досягнення консенсусу між кваліфікованою більшістю членів комітету.

Центральний секретаріат ISO публікує вихідні версії стандартів, підготовлені комітетами, і розсилає їхнім членам ISO для ознайомлення і голосування. Регулярно видається бюлетень «ISO Momento», що містить докладну інформацію про структуру і діяльність усіх технічних комітетів.

1.5. Лабораторна робота 1

Тема. Освітні та професійні стандарти в галузі ІТ

Мета роботи:

- ознайомитись з освітніми стандартами;
- ознайомитись зі стандартами проектування, реалізації та експлуатації інформаційних систем;
- ознайомитись з вимогами до освіти, навчання та досвіду роботи фахівця залежно від рівня кваліфікації;
- освоїти методологію встановлення рівня знань, умінь та навичок залежно від завдань діяльності, які освоює студент при виконанні самостійних робіт, для різних категорій працівників в ІТ-сфері.

Завдання

1. Ознайомитися з освітніми стандартами.

2. Ознайомитися з професійним стандартом (фахівець з розробки програмного забезпечення [18]).

3. Ознайомитись із основними концепціями оцінки якості ПЗ.

4. Ознайомитись з вимогами до освіти, навчання та досвіду роботи фахівця з розробки ПЗ залежно від рівня кваліфікації.

5. Встановити код та назву класу професій за класифікатором професій ДК 003:2010 освітньої спеціальності відповідно до варіанта (додаток Б. 1).

6. Встановити рівні кваліфікації (за галузевою рамкою кваліфікації) залежно від завдань діяльності фахівця з розробки ПЗ для різних функціональних областей (додаток Б. 2).

7. Чи освоює їх студент при виконанні лабораторних, курсових та дипломних робіт?

8. Ознайомитись з класифікатором завдань діяльності та трудових функцій фахівця з розробки програмного забезпечення.

9. Встановити трудові функції (код, назва) залежно від функціональних областей для різних завдань діяльності фахівця з розробки ПЗ (додаток Б. 2).

10. Чи освоює їх студент при виконанні лабораторних, курсових та дипломних робіт?

11. Встановити рівень знань, умінь та навичок залежно від завдань діяльності для різних трудових функцій (додаток Б. 3).

12. Чи освоює їх студент при виконанні лабораторних, курсових та дипломних робіт для фахівця з розробки ПЗ.

Хід роботи

1. Проведення аналізу професійного стандарту.

2. Проведення аналізу індивідуального завдання.

3. Виконати всі пункти, які передбачені у завданні.

4. Враховуючи індивідуальні особливості об'єкта дослідження, допускається розширення і поглиблення робіт «Індивідуального проекту».

Оформлення звіту

Звіт про виконану роботу оформляється згідно із «Загальними рекомендаціями до виконання лабораторних робіт та оформлення звіту» (додаток А: А. 1, А. 2).

Контрольні запитання

1. Стандарти створюються відповідно до скількох і яких принципів?
2. Чи можливо підправляти міжнародний стандарт?
3. При якій кількості критеріїв будується оцінка стандарту?
4. Яка основна мета стандартів ISO і NQA?
5. Скільки цілей переслідувало створення стандартів ISO і NQA?
6. Як будується оцінка та на якій кількості категорій?
7. Як ведуться роботи в області інформаційних технологій ISO та IEC?
8. Що набуває першорядного значення при створенні ПЗ?
9. Чи допомагають міжнародні стандарти долати технічні бар'єри в міжнародній торгівлі?
10. На що орієнтований стандарт ISO і NQA?
11. Де найбільш популярний результат стандартизації ISO і NQA?

РОЗДІЛ 2
ІНЖЕНЕРІЯ ПРОГРАМНОГО ПРОДУКТУ

Колективне розроблення великих проектів програмного продукту як комерційного програмного забезпечення обумовило розвиток інженерних, технологічних методів і засобів регламентованого проектування ПЗ з урахуванням організаційних процесів ЖЦ.

Інженерія програмного забезпечення – це технологія побудови комп'ютерних програмних систем на інженерній основі за методами, засобами та інструментами програмування, сучасними стандартами процесів ЖЦ, менеджменту та керування якістю. Особливістю виробництва нових систем є технологія їх проектування від аналізу предметної області до утворення коду для виконання на комп'ютерах.

Ця технологія використовує стандарти (наприклад, ISO/IEC 12207, ДСТУ 9126), що регламентують процеси ЖЦ, інженерію вимог, тестування і забезпечення якості шляхом перевірки показників на процесах ЖЦ і кінцевого продукту для їхнього оцінювання.

Створення *програмного продукту* – це розробка комп'ютерних систем методами та засобами програмування за такими загальними кроками:

– опис вимог;
– опис специфікацій системи;
– проектування системи;
– конструювання (реалізація) системи;
– тестування, оцінка надійності та якості системи.

Характерною ознакою виробництва програмних продуктів стала поява нових категорій фахівців, крім програмістів, а саме, менеджерів, керівників команди розробників, інженерів служби ведення бібліотек, технологів, тестувальників і різних контролерів проміжних результатів проектування на процесах ЖЦ.

Ядро знань SWEBOK і регламентовані процеси стандарту ISO/IEC 12207 узгоджені між собою. Вони утворюють практичний базис інженерії виробництва програмного продукту.

У ядрі знань SWEBOK викладено концептуальні знання й інженерні підходи до керування проектуванням програмного продукту, а в стандартах – загальні правила і регламентовані процеси послідовного його розроблення.

Інженерія програмного продукту – це застосування систематичного, дисциплінованого та вимірюваного підходу до розроблення, експлуатації і супроводження ПД із застосуванням інженерних методів до розроблення ПД.

Інженерію програмного продукту розглядатимемо з двох точок зору:

– як інженерну діяльність, у якій інженери різних категорій виконують роботи в рамках проекту, використовуючи відповідні теоретичні методи і засоби, що рекомендовані у ядрі знань SWEBOK, а також стандарти процесів проектування цільових об'єктів за вибраними методами;

– як систему керування проектом, якістю і ризиками з використанням правил і положень стандартів ЖЦ, якості та менеджменту проекту.

Ядро знань SWEBOK – стислий опис концептуальних основ інженерії програмного продукту. Структурно ділиться на 10 розділів (knowledge areas), які умовно можна розкласти за двома категоріями: проектування продукту й інженерна діяльність.

Створення продукту – це методи і засоби розробки (формування вимог, проектування, конструювання, тестування, супровід).

Інженерна діяльність – методи керування проектом, конфігурацією і якістю та базовим процесом організації-розробника.

Загальне призначення *інженерії програмного продукту* – практичне виготовлення комп'ютерних програм, систем і інструментів із застосуванням теоретичних та інженерних методів.

2.1. Формування вимог до програмного продукту

Область знань «Вимоги до ПЗ (Software Requirements)» складається з таких розділів:

1) інженерія вимог (Requirement Engineering);

2) виявлення вимог (Requirement Elicitation);

3) аналіз вимог (Requirement Analysis);
4) специфікація вимог (Requirement Specification);
5) валідація вимог (Requirement validation);
6) керування вимогами (Requirement Management).

Вимоги до ПЗ – сукупність властивостей, які повинно мати ПЗ. Призначені для адекватного визначення функцій, умов і обмежень виконання ПЗ, а також обсягів даних, технічного забезпечення і середовища його виконання.

Вимоги відображають потреби людей (замовників, користувачів, розробників), зацікавлених у створенні ПЗ. Замовник і розробник спільно виявляють вимоги, аналізують, переглядають, визначають необхідні обмеження й умови, а також описують їх. Розрізняють вимоги до продукту і процесу, а також функціональні, нефункціональні і системні вимоги. Вимоги до продукту і до процесу визначають умови виконання і режими роботи ПЗ в операційному середовищі, обмеження на структуру і пам'ять комп'ютерів та принципи взаємодії програм.

Функціональні вимоги визначають призначення і функції системи, тобто умови щодо виконання ПЗ, його переносності і доступу до даних. Системні вимоги описують вимоги до програмної системи, яка складається з взаємозалежних програмних і апаратних підсистем і різних застосувань. Вимоги можуть бути кількісні (наприклад, кількість оброблених запитів на секунду, середній показник помилок та ін.). Значна частина вимог стосується атрибутів якості: безвідмовність, надійність тощо, а також захисту і безпеки як ПЗ, так і даних.

Інженерія вимог до ПЗ – це дисципліна аналізу і документування вимог до ПЗ, що полягає в перетворенні запропонованих замовником вимог до системи на опис вимог до ПЗ і їх валідації. Інженерія базується на моделі процесу визначення вимог і діяльності осіб, що забезпечують керування і формування вимог, а також на методах досягнення показників якості.

Модель процесу визначення вимог – це схема процесів ЖЦ, що виконуються від початку проекту і доти, доки не будуть визначені і погоджені вимоги. Таким процесом може бути маркетинг і перевірка виконання вимог у цьому проекті.

Керування вимогами до ПЗ полягає в контролі за виконанням вимог і плануванні використання ресурсів (людських, програмних, технічних, часових, вартісних) у процесі розроблення проміжних робочих продуктів на процесах ЖЦ і продукту в цілому.

Якість і процес поліпшення вимог – це процес формулювання характеристик і атрибутів якості (надійність, реактивність та ін.), які повинно мати ПЗ, методи їх досягнення на процесах ЖЦ і оцінювання отриманих результатів.

Виявлення вимог – це процес витягування інформації з різних джерел (договорів, матеріалів аналітиків з декомпозиції завдань і функцій системи й ін.), проведення технічних заходів (співбесід, збирання пропозицій та ін.) для формування окремих вимог до продукту і до процесу розроблення. Вимоги погоджуються із замовником.

Аналіз вимог – процес вивчення потреб і цілей користувачів, класифікація і перетворення їх на вимоги до системи, апаратури і ПЗ, встановлення і вирішення конфліктів між вимогами, визначення пріоритетів, меж системи і принципів взаємодії із середовищем функціонування.

Специфікація вимог до ПЗ – процес формалізованого опису функціональних і нефункціональних вимог, вимог до характеристик якості відповідно до стандарту якості ISO/IEC 9126, які відпрацьовуватимуться на процесах ЖЦ ПЗ. У специфікації вимог відображається структура ПЗ, вимоги до функцій, якості та документації, а також задається архітектура системи і ПЗ, алгоритми, логіка керування і структура даних. Специфікуються також системні вимоги, нефункціональні вимоги і вимоги до взаємодії з іншими компонентами і платформами (БД, СУБД, маршаллінг даних, мережа й ін.).

Валідація вимог – це перевірка викладених у специфікації вимог, що виконується для того, щоб шляхом відстеження джерел вимог переконатися, що вони визначають саме цю систему. Замовник і розробник ПЗ проводять експертизу сформованого варіанта вимог для того, щоб розробник міг далі продовжувати проектування ПЗ. Один з методів валідації – прототипування,

тобто швидке відпрацьовування окремих вимог на конкретному інструменті й дослідження масштабів зміни вимог, вимірювання обсягу функціональності і вартості, а також створення моделей оцінки зрілості вимог.

Верифікація вимог – це процес перевірки правильності специфікацій вимог щодо їх відповідності потребам, несуперечності, повноти і можливості реалізації, а також узгодженості зі стандартами. Як результат перевірки вимог, складається погоджений вихідний документ, що встановлює повноту і коректність вимог до ПЗ, а також можливість продовження його проектування.

Керування вимогами – це керування процесами формування вимог на всіх процесах ЖЦ, а також змінами й атрибутами вимог, проведення моніторингу – відновлення джерела вимог. Керування змінами виникає після того, як ПЗ починає працювати в заданому середовищі і виявляє помилки щодо трактування вимог, невиконання деякої окремої вимоги тощо. Невід'ємною складовою процесу керування є *трасування вимог* для відстеження правильності встановлення і реалізації вимог до системи і ПЗ на процесах ЖЦ, а також зворотний процес відстеження в отриманому продукті реалізованих вимог. Для уточнення деяких вимог або додавання нової вимоги складається план зміни вимог, що погоджується із замовником. Внесені зміни спричиняють і зміни у створеному продукті або окремих його компонентах.

2.2. Проектування програмного забезпечення

Проектування ПЗ – це процес визначення архітектури, набору компонентів, їх інтерфейсів, інших характеристик системи і кінцевого складу програмного продукту.

Область знань «Проектування ПЗ (Software Design)» складається з таких розділів:

1) базові концепції проектування ПЗ (Software Design Basic Concepts);

2) ключові питання проектування ПЗ (Key Issue in Software Design);

3) структура й архітектура ПЗ (Software Structure and Architecture);

4) аналіз і оцінка якості проектування ПЗ (Software Design Quality Analysis and Evaluation);

5) нотації проектування ПЗ (Software Design Notations);

6) стратегія і методи проектування ПЗ (Software Design Strategies and Methods).

Базова концепція проектування ПЗ – це методологія проектування архітектури за допомогою різних методів (об'єктного, компонентного й ін.), процеси ЖЦ (стандарт ISO/IEC 12207) і техніки – декомпозиція, абстракція, інкапсуляція та ін. На початкових стадіях проектування предметна область декомпозується на окремі об'єкти (при об'єктно-орієнтованому проектуванні) або на компоненти (при компонентному проектуванні). Для подання архітектури програмного забезпечення обираються відповідні артефакти (нотації, діаграми, блок-схеми і методи).

Ключові питання проектування – це декомпозиція програм на функціональні компоненти для незалежного й одночасного їхнього виконання, розподіл компонентів у середовищі функціонування і їх взаємодія між собою, забезпечення якості й живучості системи та ін.

Проектування архітектури ПЗ проводиться архітектурним стилем, заснованим на визначенні основних елементів структури – підсистем, компонентів, об'єктів і зв'язків між ними.

Архітектура проекту – високорівневе подання структури системи і специфікація її компонентів. Архітектура визначає логіку системи через окремі компоненти системи настільки детально, наскільки це необхідно для написання коду, а також визначає зв'язки між компонентами. Існують і інші види подання структур, засновані на проектуванні зразків, шаблонів, сімейств програм і каркасів програм.

Один з інструментів проектування архітектури – *патерн (шаблон).* Це типовий конструктивний елемент ПЗ, що задає взаємодію об'єктів (компонентів) проектованої системи, а також ролі і відповідальності виконавців. Основна мова опису – UML. Патерн може бути *структурним,* що містить у собі структуру типової композиції з об'єктів і класів, об'єктів, зв'язків та ін.; *поведінковим,* що визначає схеми взаємодії класів об'єктів і їх поведінку,

задається діаграмами діяльності, взаємодії, потоків керування й ін.; *погоджувальним,* що відображає типові схеми розподілу ролей екземплярів об'єктів і способи динамічної генерації структур об'єктів і класів.

Аналіз і оцінка якості проектування ПЗ – це заходи щодо аналізу сформульованих у вимогах атрибутів якості, функцій, структури ПЗ, з перевірки якості результатів проектування за допомогою метрик (функціональних, структурних та ін.) і методів моделювання та прототипування.

Нотації проектування дозволяють представити опис об'єкта (елемента) ПЗ і його структуру, а також поведінку системи за цим об'єктом. Існує два типи нотацій: структурна, поведінкова, множина їх різних представлень.

Структурні нотації – це структурне, блок-схемне або текстове подання аспектів проектування структури ПЗ з об'єктів, компонентів, їх інтерфейсів і взаємозв'язків. До нотацій відносять формальні мови специфікацій і проектування: ADL (Architecture Description Language), UML (Unified Modeling Language), ERD (Entity-Relation Diagrams), IDL (Interface Description Language) тощо. Нотації містять у собі мовний опис архітектури й інтерфейсу, діаграм класів і об'єктів, діаграм сутність-зв'язок, конфігурації компонентів, схем розгортання, а також структурні діаграми, що задають у наочному вигляді оператори циклу, розгалуження, вибору і послідовності.

Поведінкові нотації відображають динамічний аспект роботи системи та її компонентів. Ними можуть бути діаграми потоків даних (Data Flow), діяльності (Activity), кооперації (Colloboration), послідовності (Sequence), таблиці прийняття рішень (Decision Tables), передумови і постумови (Pre-Post Conditions), формальні мови специфікації (Z, VDM, RAISE) і проектування.

Стратегія і методи проектування ПЗ. До стратегій відносять: проектування вгору, вниз, абстрагування, використання каркасів та ін. Методи є функціонально-орієнтовані, структурні, які базуються на структурному аналізі, структурних картах, діаграмах потоків даних й ін. Вони орієнтовані на ідентифікацію функцій і їх

уточнення знизу-вгору, після цього уточнюються діаграми потоків даних і проводиться опис процесів.

В об'єктно-орієнтованому проектуванні ключову роль відіграє спадкування, поліморфізм й інкапсуляція, а також абстрактні структури даних і відображення об'єктів. Підходи, орієнтовані на структури даних, базуються на методі Джексона і використовуються для подання вхідних і вихідних даних структурними діаграмами. Методологія UML призначена для опису сценаріїв роботи проекту у наочному діаграмному вигляді. Компонентне проектування ґрунтується на використанні готових компонентів (reuse) з визначеними інтерфейсами і їх інтеграції в конфігурацію як основи розгортання компонентної системи для її функціонування в операційному середовищі.

Формальні методи опису програм ґрунтуються на специфікаціях, аксіомах, описах деяких попередніх умов, твердженнях і постумовах, що визначають заключну умову одержання програмою правильного результату. Специфікація функцій і даних, якими ці функції оперують, а також умови і твердження – основа доведення правильності програми.

2.3. Реалізація програмного забезпечення

Реалізація (конструювання) ПЗ – створення ПЗ з конструкцій (блоків, операторів, функцій) і його перевірка методами верифікації і тестування. До інструментів конструювання ПЗ віднесені мови конструювання, програмні методи й інструментальні системи (компілятори, СУБД, генератори звітів, системи керування версіями, конфігурацією, тестуванням та ін.). До формальних засобів опису процесу конструювання ПЗ, взаємозв'язків між людиною і комп'ютером з урахуванням середовища оточення віднесені структурні діаграми Джексона.

Область знань «Конструювання ПЗ (Software Construction)» містить у собі такі розділи:

1) зниження складності (Réduction in Complexity);

2) попередження відхилень від стилю (Anticipation of Diversity);

3) структуризація перевірок (Structuring for Validation);

4) використання стандартів (Use of External Standards).

Зниження складності – це мінімізація, зменшення і локалізація складності конструювання.

Мінімізація складності – це обмеження на обробку складних структур і великих обсягів інформації упродовж тривалого періоду. Вона досягається, зокрема, використанням у процесі конструювання простих елементів, а також рекомендацій стандартів.

Зменшення складності в конструюванні ПЗ досягається шляхом створення простого коду, що легко читається і спрощує тестування, підвищує продуктивність і впливає на досягнення інших характеристик і обмежень проекту. Зменшення складності спрощує процеси верифікації і тестування результатів конструювання елементів ПЗ.

Локалізація складності – це спосіб конструювання із застосуванням об'єктно-орієнтованого підходу, що лімітує інтерфейс об'єктів, спрощує їхню взаємодію, перевірку правильності самих об'єктів і зв'язків між ними. Локалізація призначена для внесення змін, пов'язаних із виявленими помилками в коді, або коли джерелом помилок є середовище, у якому виконується код.

Попередження відхилень від стилю. Для розв'язання різних задач конструювання застосовуються різні стилі конструювання (лінгвістичний, формальний, візуальний).

Лінгвістичний стиль заснований на використанні словесних інструкцій і виразів для подання окремих елементів (конструкцій) програм. Він призначений для конструювання нескладних конструкцій і приводиться до вигляду традиційних функцій і процедур або реалізується методами логічного і функціонального програмування й ін.

Формальний стиль використовується для точного й однозначного визначення компонентів системи, мінімальної кількості помилок, що можуть виникнути у зв'язку з неоднозначністю визначень або невдалих узагальнень об'єктів конструювання ПЗ.

Візуальний стиль – найбільш універсальний для конструювання прикладного ПЗ. Він дозволяє представляти елемент конструювання у наочному вигляді. Візуальна мова проектування

UML надає розробнику набір діаграм для подання статичної і динамічної структур ПЗ. При його застосуванні створюється текстовий і діаграмний опис конструктивних елементів ПЗ, який виводиться на екран дисплея для перегляду і коригування.

Структуризація перевірок припускає, що побудова ПЗ структурована таким чином, що спрощується пошук помилок, дефектів і різних збоїв у процесі перевірок як на стадії незалежного тестування, так і в процесі експлуатації. Структуризації перевірок сприяють огляд, інспектування, спільний перегляд, модульне тестування із застосуванням автоматизованих засобів тестування й ін.

Використання зовнішніх стандартів. Конструювання ПЗ залежить від застосовних зовнішніх стандартів, пов'язаних із мовами програмування, інструментальними засобами й інтерфейсами. При конструюванні має бути визначений достатній набір стандартів для керування і забезпечення координації між визначеними видами діяльності і групами операцій, мінімізації складності, внесення змін, аналізу ризиків тощо.

До таких стандартів відносять: мови програмування (Java, Ада 95, C++ тощо), інтерфейси мов програмування і прикладні інтерфейси платформ Windows (COM, DCOM), CORBA та ін. При конструюванні використовують стандарти мов опису даних (XML, SQL тощо), засобів комунікації (COM, CORBA та ін.), інтерфейсних мов (POSIX, IDL, APL), UML та ін.

Перелічені вище розділи області знань «Конструювання ПЗ» у ядрі знань SWEBOK об'єднуються в групу «Основи конструювання». Крім того, розглядаються групи розділів «Керування конструюванням» та «Практичні міркування».

Керування конструюванням – це керування процесом конструювання ПЗ, планування, оцінка виконання плану і розроблення заходів щодо внесення змін.

Моделі конструювання містять у собі набір операцій, послідовність дій і результатів. Види моделей визначаються стандартом ЖЦ, методологіями і практиками. Деякі стандарти ЖЦ за своєю природою орієнтовані на конструювання засобами екстремального програмування і раціонального уніфікованого процесу – RUP (Rational Unified Process).

Планування – це визначення порядку операцій, термінів і рівня виконання заданих умов у процесі конструкторської діяльності за моделлю ЖЦ, що містить у собі завдання і дії зі створення, перевірки й оцінки показників якості. Остаточний результат вимірюється за обсягом коду, ступенем повторного використання, кількістю помилок і дефектів, а також оцінюванням показників якості ПЗ.

Внесення змін пов'язане з помилками, виявленими при перевірці і тестуванні, проводиться з метою збереження функціональної цілісності системи. У випадку виявлення помилок на процесі супроводження приймається рішення про внесення змін або заміну коду у цілому.

2.4. Тестування програмного забезпечення

Тестування ПЗ – це процес перевірки готової програми в статиці (перегляди, інспекції, налагодження вихідного коду) і в динаміці (прогін на наборі тестових даних) з метою перевірки різних шляхів виконання програми і порівняння отриманих результатів із заздалегідь заданими.

Існує дві форми перевірки коду – модульна й інтеграційна. Спочатку використовують стандарти (IEEE 829:1996 і IEEE 1008:1987) з перевірки і тестування модулів. Потім проводиться інтеграційне тестування модулів системи і їх інтерфейсів у динаміці виконання. Під час різних видів перевірок збираються дані про помилки, дефекти, відмови тощо й оформляється відповідна документація (таблиці типів помилок, частоти і часу виявлення відмов та ін.). Зібрані дані використовуються при оцінюванні характеристик якості готового ПЗ, наприклад надійності.

Область знань «Тестування ПЗ (Software Testing)» містить у собі такі розділи:

1) основні концепції і визначення тестування (Testing Basic Concepts and definitions);
2) рівні тестування (Test Levels);
3) техніки тестування (Test Techniques);
4) метрики тестування (Test Related Measures);
5) керування процесом тестування (Managing the Test Process).

Ця область знань SWEBOK визначає методи перевірки правильності ПЗ: верифікація, валідація, тестування. Наводяться типи, рівні і техніки тестування ПЗ, методи планування процесу тестування, розроблення тестових наборів даних для прогону ПЗ у режимі випробування модулів або системи в цілому і наступної оцінки результатів тестування.

Основна концепція тестування – це базові терміни, ключові проблеми і їхній зв'язок з іншими областями знань. Тестування визначається як процес перевірки правильності програми в динаміці її виконання за тестовими даними. При тестуванні виявляються недоліки: відмови (faults) і дефекти (defects) як причини порушення роботи програми, збої (failures) як небажані ситуації, помилки (errors) як наслідки збоїв та ін. Базове поняття тестування – тест, що виконується в заданих умовах і за наборами даних. Тестування вважається успішним, якщо знайдено дефект або помилка, і вони відразу усуваються. Ступінь тестованості визначається критерієм покриття системи тестами, перевірки всіх можливих шляхів виконання програм та імовірності припущення стосовно того, що може з'явитися збій або помилкова ситуація в системі.

Рівні тестування:

– *тестування окремих елементів* – це перевірка окремих, ізольованих і незалежних одна від одної частин ПЗ;

– *інтеграційне тестування* орієнтоване на перевірку зв'язків і взаємодії компонентів (інтерфейсів), що можуть розміщуватися на різних архітектурних платформах розподіленого середовища;

– *тестування системи* – це перевірка правильності функціонування системи, пошук і виявлення відмов і дефектів у системі та їхнє усунення. При цьому контролюється виконання сформульованих нефункціональних вимог (безпека, надійність та ін.) у системі, правильність подання і здійснення зовнішніх інтерфейсів системи з зовнішнім середовищем.

На всіх рівнях тестування застосовуються методи:

– *функціонального тестування,* які забезпечують перевірку реалізації функцій, що визначені у вимогах, а також правильності їх виконання;

– *регресійного тестування,* що орієнтоване на повторне вибіркове тестування системи або її компонентів після внесення в них змін на тих самих тестах, що і до модифікації;

– *тестування ефективності* – це перевірка продуктивності, пропускної здатності, максимального обсягу даних і системних обмежень відповідно до вимог;

– *стрес-тестування* – це перевірка поведінки системи при максимально припустимому навантаженні або в разі його перевищення;

– *альфа- і бета-тестування* – це тестування системи (альфа) групою тестувальників організації-розробника і тестування системи «зовнішніми» користувачами (бета);

– *конфігураційного тестування* – перевірка структури й ідентифікації системи, а також роботи системи на різних конфігураціях апаратури й устаткування.

Тестуванню підлягають також перевірка реалізації вимог і забезпечення параметрів настроювання і розміщення компонентів ПЗ на заданій кількості і типах комп'ютерів і середовища.

Техніки тестування базуються на певних теоретичних і практичних положеннях щодо проектування (компонентного, об'єктно-орієнтованого, сервісного та ін.), а також на таких даних:

– інформація про структуру ПЗ або системи в документації («біла скринька»);

– підбір тестових наборів даних для перевірки правильності роботи компонентів і системи в цілому без знання їх структури («чорна скринька»);

– аналіз граничних значень, таблиць прийняття рішень, потоків даних, статистики відмов та ін.;

– блок-схеми побудови програм і складання наборів тестів для покриття системи цими тестами;

– виявлені й зафіксовані в таблицях системи дефекти, перед- і постумови виконання, структурні характеристики системи (кількість модулів, обсяг даних тощо).

Метрики тестування. Для вимірювання результатів тестування ПЗ й оцінки якості використовуються метрики. Вимір як частина планування і розробки тестів базується на розмірі

програм, їх структурі й кількості виявлених помилок і дефектів. Метрики тестування – це вимірювання процесу планування, проектування і тестування, а також результатів тестування на основі таксономії відмов і дефектів, покриття границь тестування, перевірки потоків даних та ін.

Процес тестування документується і, відповідно до стандарту IEEE 829:1995, містить у собі опис тестових документів, їх зв'язок між собою і завданнями тестування. Без документації на процес тестування неможливо провести сертифікацію продукту за моделями зрілості, зокрема моделлю CMM. Після завершення тестування оцінюється вартість і ризики ПЗ, викликані збоями або недостатньо надійною роботою системи. Вартість тестування – одне з обмежень, на основі якого приймається рішення про його припинення або продовження.

Керування процесом тестування:

– планування процесу тестування (складання планів, тестів, наборів даних) і оцінювання показників якості готового продукту;

– проведення тестування компонентів повторного використання і патернів як основних об'єктів складання ПЗ;

– генерація необхідних тестових сценаріїв, що відповідають середовищу виконання ПЗ;

– верифікація правильності реалізації системи і валідація реалізації вимог до ПЗ;

– збирання даних про відмови, помилки і виявлені непередбачені ситуації при виконанні програмного продукту;

– підготовка звітів за результатами тестування й оцінка характеристик системи.

Відповідно до стандарту ISO/IEC 12207 тестування ПЗ розглядається як невід'ємна частина ЖЦ.

2.5. Супровід програмного забезпечення

Супровід ПЗ – сукупність дій із забезпечення його роботи, внесення змін при виявленні помилок, адаптації ПЗ до нового середовища функціонування, а також підвищення продуктивності або поліпшення деяких характеристик ПЗ. У зв'язку з вирішенням так званої проблеми 2000 року (пов'язаної з кодуванням дат

у новому тисячолітті, зокрема, у двохсимвольному форматі) супроводження почало розглядатися як більш важливий процес, що здійснюють розробники. Після змін система має вирішувати ті самі задачі, а також мати план перенесення інформації в інші БД. Супровід, відповідно до стандартів ISO/IEC 12207 і ISO/IEC 14764, проводиться з метою виконання і модифікації програмного продукту в процесі експлуатації за умови збереження його цілісності.

Область знань «Супровід ПЗ (Software Maintenance)» містить:

1) основні концепції (Basic Concepts);

2) процес супроводження (Process Maintenance);

3) ключові питання супроводу ПЗ (key Issue in Software Maintenance);

4) техніки супроводу (Techniques for Maintenance).

Супровід розглядається з точки зору задоволення вимог споживача у готовому ПЗ, коректності його виконання, процесів навчання й оперативного обліку його процесу.

Основні концепції – це базові визначення і термінологія, підходи до еволюції і супроводу ПЗ, до оцінки вартості супроводу тощо. До основних концепцій можна віднести ЖЦ ПЗ (стандарт ISO/IEC 12207) і складання документації. Основне призначення цієї області знань полягає у виконанні готової програмної системи, фіксації помилок, що виникають при виконанні, дослідженні їх причин, аналізі необхідності модифікації системи з метою усунення помилок, оцінці вартості робіт із проведення змін функцій і системи в цілому. Розглядаються проблеми, пов'язані з ускладненістю продукту при великій кількості змін, і методи їх подолання.

Процес супроводження містить у собі моделі процесу супроводу і планування діяльності людей, які проводять запуск ПЗ, перевірку правильності його виконання і внесення в нього змін. Цей процес, згідно зі стандартом ISO/IEC 14764, проводиться шляхом:

– коригування, тобто зміни продукту для усунення виявлених помилок або нереалізованих завдань;

– адаптації, тобто настроювання продукту в умовах експлуатації, що змінилися, або в новому середовищі виконання;

– поліпшення, тобто еволюційної зміни продукту для підвищення продуктивності або рівня супроводу;

– перевірки ПЗ, пошуку і виправлення помилок при експлуатації системи.

Ключові питання супроводу – це управлінські, вимірювальні й вартісні.

Суть управлінських питань – контроль ПЗ при модифікації й удосконалюванні функцій і недопущення зниження продуктивності системи. Питання вимірювання пов'язане з оцінкою характеристик системи після її модифікації, а також повторного тестування для оцінки показників якості. Вартісні питання пов'язані з оцінкою витрат на супровід залежно від його типу, кваліфікації персоналу, платформи й ін.

Техніка супроводу або еволюція ПЗ. Прийнято розглядати супровід як еволюційну розробку програмних систем, оскільки здана в експлуатацію система не завжди цілком завершена, її треба змінювати впродовж терміну експлуатації. Унаслідок змін система стає більш складною і погано керованою. У зв'язку з цим виникає проблема зменшення її складності. До технологій еволюції ПЗ відносять реінженерію, реверсну інженерію і рефакторинг.

Реінженерія – це удосконалення застарілого ПЗ шляхом його реорганізації або реструктуризації, а також перепрограмування окремих елементів або настроювання параметрів на іншу платформу, середовище виконання зі збереженням зручності його супроводу.

Реверсна інженерія полягає у відновленні специфікації (графів викликів, потоків даних та ін.) за отриманим кодом системи для її аналізу на більш високому рівні. Відновлюється ідентифікація компонентів і зв'язків між ними для забезпечення перепрограмування системи на нову платформу. Найчастіше реверсна інженерія застосовується після того, як у код ПЗ було внесено багато змін і воно стало некерованим або змінилася платформа комп'ютера.

Рефакторинг – це реорганізація коду для поліпшення характеристик і показників якості об'єктно-орієнтованих і компонентних програм без зміни їх поведінки. Цей процес реалізується шляхом поступової зміни окремих операцій над текстами, інтерфейсами, середовищем програмування і виконання ПЗ, а також настроювання або внесення змін в інструментальні засоби підтримки ПЗ. Якщо при зміні зберігається формат існуючої системи, то рефакторинг – один з варіантів реверсної інженерії.

2.6. Інструменти інженерії ПЗ

Інструменти забезпечують автоматизовану підтримку процесів розроблення ПЗ і містять у собі множину інструментів, що охоплюють усі процеси ЖЦ.

Інструменти роботи з вимогами (Software Requirements Tools): 1) інструменти розробки (Requirement Development) і керування вимогами (Requirement Management), орієнтовані на аналіз, збирання, специфікування і перевірку вимог; 2) інструменти трасування вимог (Requirement traceability tools) є невід'ємною частиною роботи з вимогами, їх функціональний зміст залежить від складності проектів і рівня зрілості процесів.

Інструменти проектування (Software Design Tools) – інструменти для створення ПЗ із застосуванням базових нотацій (структурної SADT/IDEF, моделювання UML тощо).

Інструменти конструювання ПЗ (Software Construction Tools) – це інструменти для трансляції і об'єднання програм. До них належать: 1) редактори програм (program editors) і програми редагування загального призначення; 2) компілятори і генератори коду (compilers and code generators) як самостійні засоби об'єднання програмних компонентів в інтегрованому середовищі для одержання вихідного продукту з використанням препроцесорів, складальників, завантажників та ін.; 3) інтерпретатори (interpreters), які забезпечують контрольоване виконання програм за їх описом. Намітилася тенденція злиття інтерпретаторів і компіляторів (наприклад, Java, в .NET); 4) відлагоджувачі (debuggers), призначені для перевірки правильності опису вихідних програм і усунення помилок; 5) інтегроване середовище

розробки (IDE – integrated development environment) та бібліотеки компонентів (libraries components), що утворюють середовище виконання процесу розроблення ПЗ; 6) програмні платформи (Java, J2EE і Microsoft .NET) і платформи для розподілених обчислень (CORBA і WebServices тощо).

Інструменти тестування (Software Testing Tools): 1) генератори тестів (test generators), що допомагають у розробці сценаріїв тестування; 2) засоби виконання тестів (test execution frameworks), які забезпечують виконання тестових сценаріїв і відслідковують поведінку об'єктів тестування; 3) інструменти оцінки тестів (test evaluation tools), які підтримують оцінювання результатів виконання тестів і ступеня відповідності поведінки тестованого об'єкта очікуваній поведінці; 4) засоби керування тестами (test management tools), які забезпечують інженерне керування процесом тестування ПЗ; 5) інструменти аналізу продуктивності (performance analysis tools), кількісної її оцінки та оцінки поводження програм у процесі виконання.

Інструменти супроводу (Software Maintenance Tools) містять у собі: 1) інструменти полегшення розуміння (comprehension tools) програм, наприклад різні засоби візуалізації; 2) інструменти реінженерії (reengineering tools) підтримують діяльність з перетворення програм і зворотної інженерії (reverse engineering) для відновлення (артефактів, специфікації, архітектури) застарілого ПЗ або генерації нового продукту.

Інструменти конфігураційного керування (Software Configuration Management Tools): 1) інструменти відстеження (tracking) дефектів; 2) інструменти керування версіями; 3) інструменти керування складанням, випуском версії (конфігурації) продукту та його інсталяції.

Інструменти керування інженерною діяльністю (Software Engineering Management Tools) підрозділяються на: 1) інструменти планування і відстеження ходу проектів, кількісної оцінки зусиль і вартості робіт у проекті (наприклад, Microsoft Project 2003); 2) інструменти керування ризиками, які використовуються для ідентифікації, моніторингу ризиків і оцінки нанесеного ушкодження; 3) інструменти кількісної оцінки властивостей ПЗ шляхом вимірювань і розрахунків остаточного значення надійності та якості.

Інструменти підтримки процесів (Software Engineering Process Tools) розділені на: 1) інструменти моделювання та опису моделей ПЗ (наприклад, UML і його інструменти); 2) інструменти керування програмними проектами (наприклад, Microsoft Project); 3) інструменти керування конфігурацією для підтримки версій і всіх артефактів проекту.

Інструменти забезпечення якості (Software Quality Tools) діляться на дві категорії: 1) інструменти інспектування для підтримки перегляду (review) і аудиту; 2) інструменти статичного аналізу артефактів, даних, потоків робіт і перевірки їх властивостей на відповідність показникам.

Додаткові аспекти інструментального забезпечення (Miscellaneous Tool Issues) стосуються: 1) техніки інтеграції інструментів (платформ, представлень, процесів, даних) для їх природного сполучення в інтегрованому середовищі; 2) мети інструментів для генерації інших інструментів для генерації ПЗ; 3) оцінки інструментів при їх еволюції.

Контрольні запитання

1. Який зміст областей ядра знань SWEBOK на прикладі прототипу?
2. Яке призначення областей ядра знань SWEBOK на прикладі прототипу?
3. Яке місце компоненти Software requirements ядра знань документа SWEBOK у створенні інформаційної системи?
4. Які основні питання розробки вимог згідно з розділом Software requirements ядра знань документа SWEBOK на прикладі прототипу?

2.7. Лабораторна робота 2

Методологія формування вимог на прикладі ІС «Компетентність студента вишу»

Зміст

2.7.1. Мета, завдання та хід роботи.

2.7.2. Напрями формування вимог ІС «Компетентність студента вишу».

2.7.3. Опис ІС «Компетентність студента вишу».

2.7.4. Підсистема контролю успішності студента.

2.7.5. Підсистема професійних і психологічних тестів.

2.7.6. Підсистема обробки запитів, визначення категорій користувачів.

2.7.7. Підсистема експертних оцінок.

2.7.8. Структура бази даних системи «Компетентність студента вишу».

2.7.9. Службова інформація.

2.7.10. Функціональні характеристики системи.

2.7.11. Цілі і завдання системи.

2.7.12. Категорії користувачів системи та можливості, що їм надаються.

2.7.13. Контрольні запитання.

2.7.1. Мета роботи:

– ознайомитися з методологією формування вимог на прикладі ІС «Компетентність студента вишу»;

– розробити на основі сформованих вимог концептуальну модель БД ІС «Компетентність студента вишу».

Завдання

1. Ознайомитися з методологією формування вимог на прикладі ІС «Компетентність студента вишу».

2. Розробити інфологічну модель БД ІС «Компетентність студента вишу».

3. Розробити концептуальну модель БД ІС «Компетентність студента вишу».

4. Розробити рекомендації щодо повноти подання вимог ІС «Компетентність студента вишу».

Хід роботи

1. Вивчити особливості методології формування вимог на прикладі ІС «Компетентність студента вишу».

2. Розробити на основі сформованих вимог інфологічну модель БД ІС «Компетентність студента вишу».

3. Розробити на основі сформованих вимог концептуальну модель БД ІС «Компетентність студента вишу».

4. Розробити рекомендації щодо повноти подання вимог та технічного проекту ІС «Компетентність студента вишу».

5. Стисло відповісти на контрольні запитання.

Оформлення звіту

Звіт про виконану роботу оформляється відповідно до «Загальних рекомендацій до виконання лабораторних робіт та оформлення звіту» (додаток А, А. 1, А. 2).

2.7.2. Напрями формування вимог ІС «Компетентність студента вишу»

У загальному випадку під *вимогами* до програмних систем розуміють властивості, які вона повинна мати для виконання запропонованих замовником функцій. Прикладами таких функцій можуть бути бізнес-функції, документообіг, керування даними і структурою інформації, що необхідна для прийняття системних рішень та ін.

Предметна область – це частина реального світу, що розглядається в межах певного контексту. Під контекстом можна розуміти область дослідження чи область, яка є об'єктом певної діяльності.

Щодо предмета дослідження в методології розуміють виділену з об'єкта певну цілісність, яка має найбільш істотні з точки зору мети цього дослідження ознаки або властивості. Предмет дослідження є більш вузьким поняттям, ніж його об'єкт, оскільки він є лише частиною, елементом або певною стороною об'єкта.

Формування вимог при створенні інформаційної системи необхідно проводити відповідно до компоненти Software requirements ядра знань документа SWEBOK (Software Engineering Body of Knowledge).

Напрями формування вимог до системи «Компетентність студента вишу» такі:

– опис системи;
– підсистема контролю успішності студента;
– підсистема професійних і психологічних тестів;
– підсистема обробки запитів, визначення категорій користувачів;
– підсистема експертних оцінок;
– структура системи (модульна);
– інформаційні об'єкти системи;
– функціональні характеристики системи;
– цілі та завдання системи;
– категорії користувачів.

2.7.3. Опис ІС «Компетентність студента вишу»

ІС призначена для того, щоб допомогти студенту визначати свій рівень професійної компетентності і залежно від цього обирати індивідуальну траєкторію навчання у виші, а також осмислено здійснювати пошук місця роботи вже у процесі його навчання.

Подавши заяву в систему, студент стає її клієнтом і починає обслуговуватися впродовж усього навчання у виші. Заява є анкетою.

Система пропонує професійні (*засновані на дисциплінах, що вивчаються*), спеціальні (*засновані на професійних вміннях та навичках*) і психологічні тестування, що проводяться регулярно (*один раз на семестр, півроку*).

Особлива увага приділяється навчанню студента. За підсумками успішності й тестування, складаються експертні оцінки, які допомагають студенту визначати свій рівень професійної компетентності і залежно від цього обирати індивідуальну траєкторію навчання у виші.

На основі зібраної інформації складається резюме, що є достатньо інформативною характеристикою особи. Це резюме можна надсилати всім організаціям, що мають необхідні вакансії.

Основним призначенням системи є автоматизація ведення і зберігання звітних даних про студентів, складання характеристик і резюме, пошуку вакансій у фірмах.

Система дозволяє змінювати, доповнювати, вести пошук і перегляд інформації про студента, накладати обмеження доступу до системи у вигляді архіву, пов'язувати виш із фірмами (зацікавленими в пошуку співробітників), зберігати дані про студента, що закінчили навчання.

Система складається з чотирьох підсистем обробки інформації щодо:

– контролю успішності студента;

– професійних і психологічних тестів;

– визначення категорій повноважень користувача та обробки запитів;

– експертних оцінок.

2.7.4. Підсистема контролю успішності студента

Ця підсистема є частиною системи «Компетентність студента вишу», яка відповідає за статистичну звітність з успішності окремого студента, групи або цілого факультету, а також за зберігання і правильність її ведення.

Вхідними даними підсистеми є: набрані бали за видами навчальних завдань та іспитів за датою, прізвищем студента, номером групи, факультетом, предметом тощо. На виході підсистема видає оброблені дані: тенденцію успішності за видами контрольних заходів та робіт, середній бал студента, групи або факультету тощо.

Підсистема «Контроль успішності студента» включає такі функції:

–введення, виведення, редагування та збереження інформації щодо інформаційних об'єктів підсистеми;

– аналіз успішності студента в групі або на факультеті й виведення його у вигляді таблиць, графіків і діаграм;

– розрахунок середнього бала студента, групи або факультету;

– формування даних щодо студента, групи або факультету;

– виявлення тенденції успішності студента в групі або на факультеті;

– розрахунок якісної успішності студентів у групі або на факультеті;

– перевірку правильності ведення даних.

2.7.5. Підсистема професійних і психологічних тестів

Склад інформаційних сутностей та каталогів

Тест – {питання тесту; тип тесту (*професійно-психологічний, психологічний, спеціальний тощо*); відповіді на питання тесту з поміткою правильної відповіді; розробник; стандартизованість тесту …}.

Тестування – {ПІП студента; тип тесту; питання тесту; відповіді на питання тесту; дата тестування; мотив тестування …}.

Фірма – {тип та інші характеристики підприємства. У такій ситуації фірма виступає як роботодавець. Вона відправляє запит на фахівця}.

Клієнт – {особа, що втупила до вишу і користується послугами системи «Компетентність студента вишу»; клієнт отримує запит на проведення професійного, психологічного і спеціального тестів і питання тестів}.

Каталог «Архів» – {сховище, де зберігаються всі дані про клієнта, починаючи з моменту користування послугами системи «Компетентність студента вишу»}.

Каталог «Додаткові джерела» – {джерела, з яких надходять тести. Це INTERNET, журнали, спеціалізовані центри …}.

Призначення модулів підсистеми

Монітор – викликає модуль введення й одержує від нього дані для виконання подальших завдань; звертається до модуля тестування, передаючи йому початкові дані; після виконання свого завдання тестування передає йому свої результати.

Введення – одержує запит від монітора, передає йому знайдені дані.

Виведення – викликається монітором і одержує від нього результати, виводячи їх на дисплей.

Тестування – викликається монітором, одержує від нього по-

чаткові дані і залежно від них звертається до того або іншого модуля. Виконавши свої функції, модулі посилають тестуванню результати дій, а тестування вже передає отримані результати монітору.

Перевірка – одержує початкові дані від тестування і передає їх своїм підмодулям: реєстрація тестованих, визначення тестів. Підмодулі передають отримані результати модулю перевірки. Тестування (професійне, психологічне, спеціальне) одержує початкові дані від модуля тестування і передає їх своїм підмодулям: професійне, психологічне, спеціальне тестування.

Обробка – одержує дані від тестування, виконує свої завдання і повертає отримані результати модулю тестування.

Перегляд – одержує дані від модуля тестування, виконує свої завдання і посилає отримані результати модулю тестування.

2.7.6. Підсистема обробки запитів, визначення категорій користувачів

Ця підсистема призначена для визначення категорії, повноважень і обробки запитів користувачів ІС «Компетентність студента вишу».

Функції підсистеми:

- введення даних у БД системи про фірми;
- введення даних у БД системи про студентів;
- визначення прав доступу зареєстрованого користувача;
- обробка запитів;
- оновлення даних від фірм;
- оновлення даних від студента;
- оновлення даних від обслуговуючого персоналу;
- складання резюме;
- адміністрування БД системи.

Робота модулів підсистеми

Визначення категорії – модуль, що визначає категорію користувача.

Визначення повноважень – модуль, що визначає повноваження користувача.

Обробка запиту – модуль, призначений для обробки запитів користувача.

Виконання запиту – модуль, призначений для виконання запитів користувача.

Введення даних до БД системи – модуль, призначений для роботи з базою даних.

2.7.7. Підсистема експертних оцінок

Ця підсистема призначена для визначення і перегляду експертної оцінки. Вона дає коротку інформацію викладачу про студента. Студент може за її допомогою орієнтуватися в навчанні.

Крім того, підсистема може надати представнику фірми деяке уявлення про студента для розгляду його як претендента на роботу.

Визначення статусу клієнта. Клієнт входить у підсистему за ідентифікаційним номером, за яким йому присвоюється певний рівень доступу. Підсистема видає меню роботи відповідно до рівня доступу.

Обробка звітного завдання. Експерт посилає запит на надання інформації про студента (групу), результати тестування. Із БД надходять дані про успішність студента (групи), результати тестування. Робить запит про критерії (методологію) оцінювання. З БД надходять критерії для визначення експертної оцінки. Визначає експертну оцінку. Робить запит на перегляд експертної оцінки. Із БД надходить експертна оцінка.

Студент посилає запит на надання інформації про успішність, результати тестування. Із БД надходить інформація про успішність студента, результати тестування. Робить запит на перегляд експертної оцінки, яка надходить із БД.

Представник фірми робить запит на перегляд експертної оцінки, яка надходить із загальної БД.

Критерії експертних оцінок (наприклад, за 100-бальною системою):

– оцінити студента на початку, в середині, наприкінці семестру;

– оцінити ступінь запам'ятовування курсу студентом на початку, в середині, в кінці семестру;

– оцінити ступінь застосування знань студентом на початку, в середині, в кінці семестру.

2.7.8. Структура БД системи «Компетентність студента вишу»

База даних містить у собі таблиці даних:

1) з експертного оцінювання;

2) з успішності студента;

3) за результатами професійного і психологічного тестування.

2.7.9. Службова інформація

– запис про активне з'єднання;

– запис про користувальну транзакцію;

– рівень доступу в систему;

– обліковий запис повноважень користувача (для організації захисту від несанкціонованого доступу).

2.7.10. Функціональні характеристики системи

– первинне введення інформації в БД;

– супровід БД (введення нових даних, зміна існуючих даних, архівація даних);

– здійснення пошуку в БД за запитом користувача;

– віддалений доступ до системи;

– забезпечення захисту і безпеки даних, зокрема розподіл доступу до ресурсів сервера, контроль даних, що вводяться, забезпечення цілісності БД;

– виведення знайденої інформації.

2.7.11. Цілі й завдання системи

ІС «Компетентність студента вишу» забезпечуватиме зберігання, видачу й оновлення інформації освітнього процесу студента:

– забезпечувати інформаційну підтримку системи навчання студента;

– представляти й одержувати накопичену інформацію щодо конкретних об'єктів;

– представляти й одержувати інформацію від підсистеми обробки запитів користувачів;

– представляти й одержувати інформацію від підсистеми контролю студента за дисциплінами, що вивчаються на кафедрі;

– забезпечувати розмежування доступу до інформаційних ресурсів системи;

– забезпечувати моніторинг активних і пасивних користувачів і системних подій;

– забезпечувати користувачів можливістю інформаційного обміну;

– забезпечувати зв'язок між фірмами і виш;

– забезпечувати регулярне проходження студентами професійних і психологічних тестів;

– забезпечувати пошук даних за запитами.

2.7.12. Категорії користувачів ІС та можливості, що їм надаються

При роботі з ІС на стадіях наповнення під час експлуатації БД необхідна участь таких категорій користувачів: адміністратор БД та група експертів.

Адміністратор системи є внутрішній користувач і здійснює заповнення БД інформацією, підготовленою навчальною частиною, деканатом або групою експертів. Внесення змін у БД системи здійснюється лише адміністратором системи під керівництвом групи експертів. Викладачі і студенти є кінцевими користувачами, що працюють з ІС відповідно до прав доступу в інформаційно-пошуковому режимі.

Можливості, що надаються студенту:

– введення особистих анкетних даних;

– проходження психологічних і професійних тестів;

– отримання і здача контрольних завдань;

– перегляд експертних оцінок за звітними завданнями і результатами тестів за відповідними запитами;

– перегляд зведених таблиць і графіків за відповідними запитами;

– доступ до навчально-методичного забезпечення;

– перегляд повідомлень і внесення змін у повідомлення дошки оголошень за відповідними запитами;

– пошук вакансії в БД за запитом.

Можливості, що надаються експерту (викладачу):

– надання експертної оцінки, а також зміна її;

– перегляд інших оцінок за відповідними запитами;

– перегляд програми курсу і розробка рекомендацій на внесення змін;

– введення контрольних завдань і призначення їх студенту;

– реферування відповідей на завдання;

– доступ до інтелектуальних ресурсів;

– складання резюме (характеристик).

Можливості, що надаються керівництву навчальним процесом *(директору, декану, заступникам директора, заступникам декана):*

– перегляд програми курсу;

– перегляд динаміки успішності курсу, групи, окремого студента за відповідними запитами;

– перегляд зведених таблиць і графіків за відповідними запитами;

– перегляд експертних оцінок і характеристик викладачів за відповідними запитами.

Можливості, що надаються адміністратору:

– визначення прав доступу;

– введення і коректування системних даних;

– контроль роботи системи;

– здійснення контролю захисту системи від несанкціонованого доступу;

– зміна фізичної моделі даних системи.

Можливості, що надаються оператору:

– складання зведених таблиць і графіків;

– заповнення полів БД системи (введення інформації).

Можливості, що надаються фірмі:

– пошук у БД даних про студента за запитом;

– перегляд резюме студента.

Контрольні запитання

1. Яке призначення ІС «Компетентність студента вишу»?
2. Яке призначення підсистеми контролю успішності студента?

3. Яке призначення підсистеми професійних і психологічних тестів?

4. Яке призначення підсистеми обробки запитів?

5. Яке призначення підсистеми визначення категорій користувачів?

6. Яке призначення підсистеми експертних оцінок?

7. Яка структура БД системи «Компетентність студента вишу» ?

8. Які функціональні характеристики ІС?

9. Які цілі і завдання ІС?

2.8. Лабораторна робота 3

Тема. Аналіз предметної області та формування вимог до ІС індивідуального об'єкта дослідження

Мета роботи:

➢ *закріпити знання, уміння та навички* з інженерії програмного продукту як технології побудови програмних систем на інженерній основі за методами, засобами й інструментами програмування, сучасними стандартами процесів ЖЦ, менеджменту та керування якістю;

➢ здійснити опис предметної області індивідуального об'єкта дослідження (типовий приклад подано на рис. 1.2);

➢ сформувати вимоги до ІС для індивідуального об'єкта дослідження.

Завдання

1. Ознайомитись із особливостями реалізації ІС для базового прикладу та виробити стратегію реалізації власного проекту.

2. Ознайомитись із особливостями експлуатації ІС для базового прикладу та виробити стратегію експлуатації власного проекту.

3. Сформувати вимоги до ІС індивідуального об'єкта дослідження (додаток Г) щодо:

– функціональних характеристик;
– надійності;
– налаштованості ПЗ;
– умов експлуатації;

– складу і параметрів технічних засобів;
– інформаційної і програмної сумісності;
– програмної документації.

4. Здійснити опис предметної області «Індивідуального проекту», встановивши:
– призначення системи «Індивідуальний проект»;
– склад підсистем;
– цілі системи і підсистем;
– модульну структуру всієї системи;
– основні інформаційні об'єкти «Індивідуального проекту»;
– порядок занесення даних у БД;
– функціональні характеристики системи;
– дані, які надходять на вхід кожної з підсистем;
– категорії користувачів системи і надані їм можливості;
– призначення підсистеми обробки запитів та визначення категорій користувачів;
– критерії функціонування системи.

5. Розробити ескізи пропозицій у вигляді схем, нотацій, діаграм чи моделей щодо отриманих результатів у пп.1–3 завдання.

Хід роботи

1. Проведення аналізу індивідуального об'єкта дослідження.

2. Виконання роботи для індивідуального об'єкта дослідження здійснюється відповідно до завдання-прототипу.

3. Виконати всі пункти, які передбачені у завданні.

4. Враховуючи індивідуальні особливості об'єкта дослідження, допускається розширення і поглиблення робіт «Індивідуального проекту» (може ускладнитися процес отримання зворотної інформації про виконання індивідуальних завдань та їхнього корегування).

Оформлення звіту

Звіт про виконану роботу оформляється відповідно до «Загальних рекомендацій до виконання лабораторних робіт та оформлення звіту» (додаток А: А. 1, А. 2).

Контрольні запитання

1. Для чого призначена система «Індивідуальний проект»?

2. З яких підсистем складається система «Індивідуальний проект»?

3. Вказати цілі системи і підсистем.

4. Зобразити модульну структуру всієї системи.

5. Перерахувати основні інформаційні об'єкти «Індивідуального проекту».

6. Описати структуру та порядок занесення даних у БД.

7. Перерахувати функціональні характеристики системи.

8. Які дані надходять на вхід кожної з підсистем?

9. Назвати категорії користувачів системи і надані їм можливості.

10. Яке призначення підсистеми обробки запитів?

11. Дати визначення категорій користувачів.

12. Описати критерії функціонування системи.

РОЗДІЛ 3
МІЖДИСЦИПЛІНАРНЕ ІНФОРМАЦІЙНЕ МОДЕЛЮВАННЯ

3.1. Інформаційне моделювання

Сучасний аналіз являє собою сукупність методів і засобів, що дозволяють досліджувати властивості, структуру і функції об'єктів, явищ або процесів у цілому, представивши їх як системи з усіма складними міжелементними взаємозв'язками, взаємовпливом елементів на систему і на навколишнє середовище, а також впливом самої системи на її структурні елементи.

Невід'ємною частиною аналізу є моделювання – процес дослідження реальної системи, що включає побудову моделі, вивчення її властивостей і перенесення отриманих відомостей на модельовану систему. Загальними функціями моделювання є опис, пояснення і прогнозування поведінки реальної системи. Типовими цілями моделювання можуть бути пошук оптимальних або близьких до оптимальних рішень, оцінка ефективності рішень, визначення властивостей системи (чутливості до зміни значень характеристик та ін.), установлення взаємозв'язків між характеристиками системи. У найбільш загальному формулюванні модель – це об'єкт, що має подібність із прототипом і слугує засобом опису і/або пояснення, і/або прогнозування поведінки прототипу.

Зазначимо важливі властивості будь-якої моделі.

1. Спрощеність – модель відображає тільки істотні сторони об'єкта і, крім того, повинна бути проста для дослідження або відтворення.

2. Обмеженість – модель відображає оригінал лише в обмеженому числі його відносин і, крім того, ресурси моделювання також обмежені.

3. Наближеність – дійсність відображається моделлю грубо або наближено.

4. Адекватність – модель повинна успішно описувати модельовану систему.

5. Цілісність – модель реалізує деяку систему (тобто ціле).

6. Замкнутість – модель враховує і відображає замкнену систему необхідних основних гіпотез, зв'язків і відносин.

7. Керованість – модель повинна мати хоча б один параметр, змінами якого можна імітувати поведінку модельованої системи в різних умовах.

Складні системи характеризуються виконуваними процесами (функціями), структурою і поведінкою у часі. Для адекватного моделювання цих аспектів в автоматизованих інформаційних системах розрізняють функціональні, поведінкові моделі й інформаційні, що перетинаються між собою.

Функціональна модель системи описує сукупність виконуваних системою функцій, характеризує морфологію системи (її побудову) – склад функціональних підсистем, їхнього взаємозв'язку.

Поведінкова (подійна) модель описує інформаційні процеси (динаміку функціонування), у ній фігурують такі категорії, як стан системи, подія, перехід з одного стану в інший, умови переходу, послідовність подій.

Інформаційна модель відображає відносини між елементами системи у вигляді структур даних (склад і взаємозв'язки).

Можна виділити три основні області застосування моделей: навчання, наукові дослідження, управління. Під час навчання за допомогою моделей досягається висока наочність відображення різних об'єктів і полегшується передача знань про них. Це в основному моделі, що дозволяють описати й пояснити систему. У наукових дослідженнях моделі слугує засобом одержання, фіксування й упорядкування нової інформації, забезпечуючи розвиток теорії і практики. В управлінні моделі використовуються для обґрунтування рішень. Такі моделі повинні забезпечити як опис, так і пояснення і передбачати поведінку систем.

Інформаційні системи, будучи основним інструментом підвищення ефективності управлінських рішень, являють собою складні програмно-апаратні й телекомунікаційні комплекси, виступають як самостійний об'єкт досліджень. Питання архітектури таких систем і організації управління інформаційним процесом

представляють один із основних напрямів розглянутої області знань. Інформаційні системи покликані допомогти в рішенні інформаційних, логічних і розрахункових завдань. Теоретичні основи інформаційного, лінгвістичного, математичного, програмного й іншого видів забезпечення розподіленої обробки інформації, побудови баз даних, баз знань, аналітичних та інших систем визначають потенційні можливості й обмеження інформаційних систем у цілому.

Найпростішим і абстрактним рівнем опису системи є інформаційний підхід до моделювання, орієнтованого на використання даних, що характеризують властивості та стани системи, її взаємозв'язок із зовнішнім світом.

При інформаційному підході реальний об'єкт розглядається як «чорний ящик». У цьому випадку передбачається, що виділена система пов'язана із середовищем через сукупність входів і виходів. Виходи моделі описують результати діяльності системи, а входи – ресурси й обмеження. При цьому передбачається, що нічого не знаємо про внутрішній зміст системи. Модель у цьому випадку відображає дві важливі й істотні її властивості: цілісність і відособленість від середовища. Декомпозиція внутрішньої структури чорного ящика на більш дрібні складові (підсистеми, окремі елементи) дозволяє будувати моделі складу систем. Простота і доступність моделей «чорного ящика» і складу системи дозволяє вирішувати з їхнім використанням множину практичних завдань. Разом з тим для більш детального (глибокого) вивчення систем необхідно встановлювати в моделі сукупність відносин (зв'язку) між їх елементами.

3.2. Міждисциплінарне інформаційне моделювання у навчальному процесі

Сучасна освіта ставить своїм завданням формування особистості, готової до професійної, соціальної та інших видів діяльності, які передбачають системний світогляд, здатність вирішувати завдання в рамках навколишньої дійсності, із застосуванням знань, умінь та навичок, сформованих у різних предметних областях. Досягти цього можна за допомогою виявлення та реалізації

міждисциплінарних зв'язків під час навчального процесу. Успішне оволодіння новими знаннями з інформаційних технологій в умовах глобалізації та зростання обсягів інформації, які породжує науково-технічний прогрес, вимагає від учасників навчально-виховного процесу якісно нових загальних підходів та нового світобачення усіх процесів розвитку суспільства. Такому світобаченню сьогодні сприяє синергетичний підхід, який дозволяє досліджувати об'єкт у взаємозв'язку з суміжними галузями знань.

Система освіти з комп'ютерних наук є реальною за походженням, великою за кількістю складових елементів, відкритою за характером взаємодії із зовнішнім середовищем, динамічною за розвитком, цілеспрямованою за наявністю мети, складною за засобом детермінації. Вивчення існуючої різноманітності програмних продуктів значно полегшується і систематизується студентами, якщо у них сформовано єдиний підхід і вироблено загальну методологію, яка ґрунтується на об'єднавчих принципах.

Така діяльність має спрямовуватись на засвоєння студентами принципів синергетики у зв'язку із професійними знаннями конкретної дисципліни та з урахуванням взаємодоповнюваності знань з різних навчальних дисциплін.

У системі підготовки фахівців з комп'ютерних наук актуальним виступає так званий метод ІТ-проектів. Доцільність його застосування визначається необхідністю:

- не тільки передавати студентам сукупність знань, а і навчати їх мислити і формувати власну думку про суть проекту, отримувати ці знання самостійно завдяки конструктивній проблематизації, вміти користуватись отриманими знаннями для вирішення нових практичних завдань;
- напрацювання студентами комунікативних навичок та вмінь при роботі із різними технологіями та в різних групах, виконуючи різні ролі з реалізації всіх етапів життєвого циклу ІТ-проекту;
- розширення студентами людських контактів, сприйняття різних точок зору на поставлену проблему і розроблений ІТ-проект, а також розвитку умінь використовувати ними дослідницькі методи.

У зв'язку з цим потрібно зазначити, що при підготовці компетентних фахівців з комп'ютерних наук важливу роль слід відводити отриманню студентами необхідних теоретичних знань та формуванню у них практичних умінь і навичок з реалізації всіх етапів життєвого циклу ІТ-проектів (дослідження предметної області та формування вимог, розробка та затвердження технічного завдання, проектування або перепроектування, реалізація, супровід, зняття з експлуатації).

Враховуючи складність проблеми реалізації усіх складових проекту і значний обсяг робіт, створення повномасштабного ІТ-проекту та досягнення навчальної мети в межах однієї дисципліни є проблематичним. Тому виникає необхідність в організації інтегрованого викладання шляхом реалізації горизонтальних і вертикальних міждисциплінарних зв'язків у процесі навчання студента на різних курсах.

Одним із прикладів такого підходу є реалізація міждисциплінарного інформаційного моделювання на основі індивідуального наскрізного об'єкта дослідження з використанням різних методологій, інструментальних засобів і технологій у процесі викладання різних комп'ютерних дисциплін.

Впроваджені методичні розробки передбачають комплекси лабораторних робіт з кожної дисципліни, виконання яких здійснюється в два етапи: ознайомлення за необхідною методикою на прототипі; закріплення отриманих знань, умінь та навичок при реалізації «Індивідуального проекту» студента на основі індивідуального наскрізного об'єкта дослідження.

З вивченням нової дисципліни «Індивідуальний проект» кожен студент отримує подальший розвиток за рахунок удосконалення попередніх моделей та отриманих моделей у результаті освоєння нових методологій, інструментальних засобів і технологій. «Індивідуальний проект» трансформується в «Наскрізний індивідуальний проект».

З метою об'єктивного діагностування результатів професійної підготовки фахівця на науковому рівні розроблено системні, професійно орієнтовані критерії, що дозволяють оцінити рівень сформованості його професійної компетентності.

Передбачається 5 рівнів сформованості професійної компетентності фахівця: мінімальний, низький, середній, високий, найвищий.

1. Мінімальний рівень підготовки дозволяє фахівцю використовувати професійні знання для виконання лише конкретного завдання, проте він не завжди може застосувати їх для виконання іншого завдання; фахівець добре працює за алгоритмом і за чіткими інструкціями.

На реалізацію такого рівня компетентності орієнтований в основному перший етап виконання лабораторних робіт – ознайомлення з необхідною методологією на прототипі.

2. Низький рівень підготовки дозволяє фахівцю адаптувати професійні знання до особливостей завдання, він володіє вміннями використовувати їх для вирішення іншого класу аналогічних завдань, діяти відповідно до конкретних обставин.

На реалізацію цього рівня компетентності орієнтований другий етап виконання лабораторних робіт, тобто закріплення отриманих знань, умінь та навичок з освоєння такої методології при реалізації відповідного завдання (встановлення предмета дослідження) на основі індивідуального наскрізного об'єкта дослідження студента зі створення «Наскрізного індивідуального проекту».

3. Середній рівень підготовки дозволяє фахівцю самостійно моделювати систему професійних і спеціальних знань за певним класом завдань; він уміє вирішувати професійні завдання шляхом створення моделей, адекватних для цього класу завдань.

На реалізацію цього рівня компетентності орієнтовано виконання модульних контрольних робіт зі створення інформаційних моделей на основі нових завдань із суто економічним спрямуванням.

Можна віднести аналітико-синтетичну роботу студентів з текстовою інформацією: формування тексту на задану тему за певною структурою, реферування, рецензування, розробка тестів, створення мультимедійної презентації на теоретичний матеріал, доповідь на занятті у вигляді активного круглого столу.

4. Високий рівень підготовки дозволяє фахівцю моделювати алгоритм діяльності цього класу завдань і переносити його на вирішення нового, більш складного класу завдань; він володіє засобами моделювання в різних предметних областях, уміє розробляти моделі для нових класів завдань.

На реалізацію цього рівня орієнтовано виконання курсової роботи зі створення «Наскрізного індивідуального проекту» на основі індивідуального наскрізного об'єкта дослідження. Сюди можна віднести виконання підсумкових контрольних робіт зі створення інформаційних моделей на основі більш складного класу завдань із суто економічним спрямуванням.

5. Найвищий рівень підготовки передбачає володіння фахівцем системою загальнопрофесійних і спеціальних знань, умінь і здібностей, володіння творчим практичним мисленням, застосування системного підходу до вирішення різних завдань, а також їх творче вирішення.

На реалізацію цього рівня орієнтовано виконання курсової і дипломної робіт на довільну тему, погоджених з науковим керівником.

Впровадження міждисциплінарного інформаційного моделювання дозволило виявити освітньо-розвиваючий потенціал від інтеграції дисциплін у ході проектної діяльності, зокрема:

1) відповідність сучасному рівню наукових уявлень про світ;

2) можливість розгорнути перед студентами багатовимірну наукову картину світу в динаміці;

3) стимул до пошуку нових методичних видів взаємодії зі студентами, що відповідають принципам інтеграційного підходу;

4) злиття зусиль різних викладачів-предметників у вирішенні загальних проблем, можливість накопичення ціннісної орієнтації і мотивації студентів;

5) зменшення перевантаження в навчальному процесі, отримання якісно нового педагогічного результату;

6) усвідомлення викладачем нового бачення своєї дисципліни та співвідношення її з іншими науками.

Таким чином, актуалізація міждисциплінарних зв'язків у процесі міждисциплінарного інформаційного моделювання зі створення «Наскрізного індивідуального проекту» сприятиме більш міцному освоєнню пройденого матеріалу та виробленню інтегрованих умінь і навичок.

Розроблена педагогічна технологія зі створення студентами програмного продукту через наскрізний індивідуальний проект відповідає новим технологіям освіти і є перспективною. У результаті її впровадження досягаються для студентів виховна, освітня та розвиваюча цілі: 1) виховання дисциплінованості, організованості; 2) формування логічного, системного мислення, володіння інтелектуальними вміннями та розумовими операціями: аналізом, синтезом, узагальненням, порівнянням; 3) розвиток творчих здібностей, пізнавального інтересу і позитивної мотивації до навчання; 4) узагальнення знань та формування необхідних навичок, отриманих під час вивчення різних дисциплін; 5) сприяння в цілому формуванню та розвитку професійних компетентностей.

3.3. Архітектура моделей даних інформаційної системи

АІС підтримує різноманітні уявлення користувачів про предметну область. Для кінцевих користувачів АІС – сховище повідомлень за угодами і постачаннями, документами і цінниками тощо. Однак для внутрішніх користувачів (адміністраторів АІС і БД, системних і прикладних програмістів) та ж АІС подається у вигляді елементів даних, записів, сторінок і файлів. Зазначимо також, що уявлення внутрішніх користувачів АІС теж аж ніяк не ідентичні. Прикладний програміст оперує елементами даних, записами, ключами, структурними асоціаціями, але він, як правило, не має уявлення про фізичну організацію даних. Це вже прерогатива адміністратора і системних програмістів.

Як бачимо, у сучасній ІС підтримується кілька рівнів моделей подання даних. Їх різновиди будемо асоціювати з поняттям архітектури моделей подання даних (АМПД) ІС.

При розробці архітектури моделей подання даних ІС керуються положеннями Комітету планування стандарту і норм SPARK (Standards Planning and Requirements Committee) Національного інституту стандартизації США (American National Standard Institute – ANSI) та державного стандарту України ДСТУ 3329-96 щодо концепції для концептуальної схеми та інформаційної бази. Відповідно до них пропонується три рівні архітектури моделей подання даних: зовнішній, логічний і внутрішній (реалізації).

Враховуючи особливості їх організації, сприйняття й адміністрування, допускається поділ окремих рівнів на підрівні. Зовнішній представимо, власне, зовнішнім та інфологічним, а внутрішній, – власне, внутрішнім і фізичним, а тому розглядатимемо п'ять рівнів архітектури моделей подання даних: зовнішній, інфологічний, концептуальний, внутрішній і фізичний.

Початковий рівень моделі подання даних у такій архітектурі називається *зовнішнім* і відповідає локальним уявленням про предметну область кінцевих користувачів, а саме спеціалістів з предметної області.

Другий рівень називається *інфологічним,* він являє собою інтеграцію локальних уявлень про предметну область кінцевих користувачів, що відповідає «погляду» на предметну область її адміністратора (директора, міністра, ректора вишу тощо). Він вбачає усю множину інформаційних об'єктів, усі можливі асоціації між ними, водночас, як кожен кінцевий користувач, сприймає лише обмежений фрагмент предметної області (ПО). Зазначимо, що локальні уявлення користувача та інфологічне уявлення про предметну область існують поза будь-яким зв'язком із засобами реалізації бази даних ІС. Вони визначають лише інформаційні потреби розроблювальної системи, відображають особливості ПО, але не торкаються запитання, як відповідні дані будуть подані в пам'яті ЕОС.

Концептуальний рівень архітектури моделей подання даних відповідає уявленню про логічну організацію даних адміністратора БД. Цей рівень подання даних досить подібний до інфологічного, але його відмінність полягає в прив'язці до засобів реалі-

зації – СУБД. Опис БД на концептуальному рівні задається мовою опису даних використовуваної СУБД у термінах і обмеженнях, які прийняті у цій системі. Оскільки кожна СУБД на концептуальному рівні підтримує свою модель даних, перетворення інфологічного опису ПО в концептуальний опис для БД різних СУБД приводить до різних результатів.

Подання даних у пам'яті обчислювальних систем, їхню організацію збереження визначає *рівень реалізації* на основі мови опису, маніпулювання та збереження СУБД. Зазначимо, що параметри рівня реалізації подання БД впливають на ефективність інформаційної системи, наприклад на обсяг зовнішньої пам'яті, час реакції системи. У процесі функціонування АІС адміністратор може змінювати параметри реалізації БД, залишаючи без змін описи, що задаються на попередніх рівнях подання даних. Це дозволяє змінювати ефективність системи, не вносячи ніяких коригувань у розроблені додатки. Рівень реалізації АМПД представляється двома підрівнями: внутрішнім та фізичним. *Внутрішній рівень* АМПД – система позв'язаних електронних таблиць і програм, яка може бути представлена на пристроях видачі, формується засобами опису та маніпулювання даними СУБД. *Фізичний рівень* АМПД, як система файлів електронних таблиць даних і програм на носіях інформації, формується мовою збереження даних СУБД в елементах пам'яті обчислювальної системи.

На кожному рівні подання даних визначається своя модель ПО. Описи цих моделей можна називати схемами, тому говорять, що схеми локальних уявлень користувачів відображаються в інфологічну схему ПО, остання відображається в концептуальну схему БД тощо.

Структурний аналіз моделей даних ІС. Кожен рівень подання даних АІС можна описати через структурні компоненти і таким чином здійснити структурний аналіз моделей інформаційної системи шляхом декомпозиції кожного рівня подання даних від загального до найменшого складового елемента. Архітектуру моделей подання даних АІС та їх структурний аналіз представлено в таблиці 3.1.

На основі такого представлення АМПД можна відслід-

кувати відображення кожного структурного елемента на певному етапі декомпозиції рівнів архітектури подання даних АІС.

Наприклад, на 3-му розділі декомпозиції ***реквізит*** як певна властивість процесу або документа відображається в ***атрибут*** на концептуальному рівні. У свою чергу, атрибут відображається в ***поле*** на внутрішньому рівні подання даних.

Зазначимо, що усі відображення повинні бути гомоморфними. Вимога гомоморфізму означає, що в БД (на 4-му і 5-му рівні) повинні міститися дані, що відповідають усім характеристикам об'єкта предметної області (на 1-му рівні), з одного боку, а з іншого – будь-якому процесу обробки інформації ПО в автоматизованій інформаційній системі (на 1-му рівні) повинна відповідати деяка процедура обробки відповідних даних (на 4-му і 5-му рівні), що дозволяє отримати адекватний результат.

Слід зазначити, що в процесі оптимізації структури БД кожному ІНФОРМАЦІЙНОМУ_ОБ'ЄКТУ 1-го рівня відповідає одна або декілька СУТНОСТЕЙ 2-го рівня і кожній СУТНОСТІ 2-го рівня відповідає одна або декілька СТРУКТУРОВАНИХ_ТАБЛИЦЬ 3-го рівня АМПД. Сумарна кількість характеристик документів і процесів буде повністю відображена в реквізитах усіх сутностей 2-го рівня й атрибутах усіх структурованих таблиць 3-го рівня АМПД (за винятком надлишку атрибутів – зовнішніх ключів).

За складом характеристик, тотожними є СТРУКТУРОВАНІ_ТАБЛИЦІ 3-го рівня і ЕЛЕКТРОННІ_ТАБЛИЦІ 4-го рівня АМПД.

Таблиця 3.1

Архітектура та структурний аналіз моделей АІС

Рівень / Декомпозиція	*Зовнішній: предметна область (ПО) – предмет дослідження*	*Інфологічний: на основі знань спеціаліста предметника*	*Концептуальний:на основі моделі даних СУБД*	*Внутрішній: засобами опису та маніпулювання даними СУБД*	*Фізичний: засобами збереження даних СУБД у пам'яті ЕОС*
1	Система інформаційних об'єктів ПО (система документів	Інфологічна модель: діаграма «Сутність» / «Зв'язок»	Концептуальна модель: система пов'язаних таблиць за однаковими	База даних як система пов'язаних електронних таблиць і	Сукупність файлів таблиць даних і програм на носіях

	і процесів)	або ERD	значеннями атрибутів	програм	інформації
2	Інформаційний об'єкт	Сутність	Структурована таблиця	Запис	Фізичний запис
3	Властивість об'єкта	Реквізит	Атрибут	Поле	Сукупність комірок пам'яті ЕОС
4	Характеристика властивості	Значення реквізиту	Значення атрибуту	Значення поля	Стан комірок пам'яті ЕОС

3.4. Проектування баз даних інформаційних систем

Проектування структури БД – це визначення її складу, структури і технологічного процесу її створення та ведення, а також розробка організаційно-методичних та інструктивних матеріалів. Проектування БД виконується в кілька етапів відповідно до рівнів подання даних.

Проектування на зовнішньому рівні. Метою проектування на зовнішньому рівні є розробка інформаційного забезпечення, яке вміщує систему вхідної (первинної) документації, що характеризує певну ПО, систему класифікації та кодування інформації, а також перелік відповідних вихідних повідомлень, які потрібно формувати з використанням БД.

Проектування баз даних на зовнішньому рівні може проводитися з автоматизації вирішення задач для всієї предметної області (комплексний, предметний підхід) або з вирішення задач окремих підобластей ПО (процесний або функціональний підхід).

Комплексний підхід забезпечує об'єктивність, системність наповнення і стійкість інформаційної моделі за наявності значних ресурсів (фінансових, інформаційних, часових тощо) для розробки проекту.

Функціональний підхід орієнтований на створення БД для реалізації поточних вимог користувачів і прикладних програм. Це сприяє створенню системи з високими експлуатаційними характеристиками, але при цьому можуть виникнути складності в консолідації даних різних підсистем ПО.

Процес проектування БД на зовнішньому рівні складається з таких основних робіт: 1) визначення функціональних задач ПО, що підлягають автоматизованому розв'язанню; 2) вивчення

та аналіз оперативних первинних документів; 3) вивчення нормативно-довідкових документів; 4) вивчення правил (формул, алгоритмів) перетворення первинних характеристик документів і процесів (вхідних) у розрахункові (вихідні).

Результатом проектування на зовнішньому рівні буде перелік реквізитів оперативної інформації, які потрібно зберігати в БД, із зазначенням джерел їх отримання та форми подання. Однак цей перелік не виключає можливості існування в ньому надлишковості, дублювання, неузгодженості та інших недоліків. На цьому процес не закінчується, а здійснюється перехід до етапу інфологічного проектування.

Інфологічне проектування. Метою інфологічного проектування є створення структурованої інформаційної моделі ПО, для якої розроблятиметься БД. При проектуванні на інфологічному рівні створюється інформаційно-логічна модель у вигляді діаграми «Сутність»/Зв'язок», яка повинна адекватно відображати модельовану ПО. Інфологічні моделі дозволяють відобразити семантику модельованої предметної області на консолідованому рівні графічним способом (рис. 3.1).

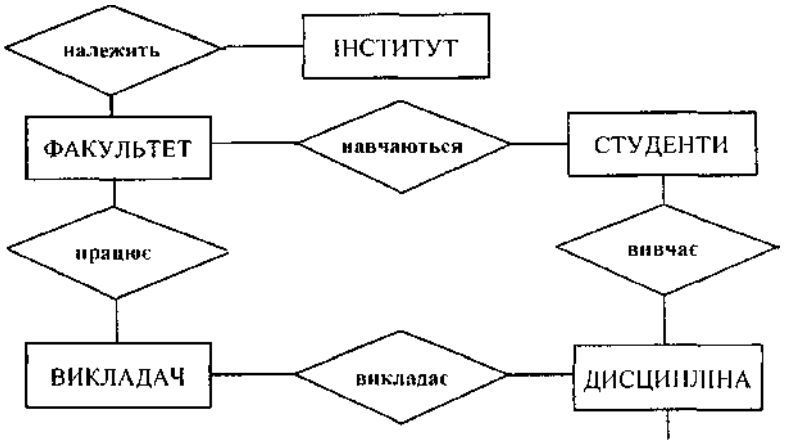


Рис. 3.1. ER-діаграма фрагмента предметної області «виш»

Основною складовою інфологічної моделі є реквізити, які потрібно проаналізувати і певним чином згрупувати в сутності для подальшого зберігання в БД. Таким чином, суть інфологічного моделювання полягає у виділенні сутностей на основі інформаційних об'єктів ПО, взаємозв'язків між ними і складу реквізитів.

Сутністю може виступати предмет, факт, дія, явище чи поняття, що є предметом пізнання людини чи результатом її діяльності, інформацію про які потрібно зберігати в БД. Розрізняють тип сутності та її екземпляр. Тип сутності описує абстрактні, а екземпляр – конкретні характеристики сутності. Наприклад, сутність СТУДЕНТИ (№ залікової книжки, ПІП, рік народження, курс, група) є типом, а СТУДЕНТИ (5393, Скорик, 1998, 2-й курс, КБ-16-1) – екземпляром сутності.

При інфологічному проектуванні виконуються такі дії.

1. Аналіз реквізитів на предмет вилучення синонімії і омонімії. Серед реквізитів можуть зустрітися такі, що мають однакові імена, але є різними за змістом, тобто омоніми. У цьому разі реквізитам-омонімам потрібно присвоїти різні унікальні імена. Прикладом реквізита-омоніма може бути реквізит з іменем *«залишок»*, який в одному випадку може характеризувати *залишок матеріалів*, а в інших – *залишок коштів, готової продукції.* Тому для однозначної ідентифікації кожному залишку необхідно присвоїти унікальне ім'я.

Крім того, є випадки синонімії, коли один і той самий реквізит у різних документах має різну назву. У цьому разі всім таким реквізитам присвоюється одне ім'я. Обираючи назву для синонімічної групи реквізитів, серед реквізитів-синонімів необхідно обрати ім'я, яке є семантичною домінантою для представленої групи, і залишити його в переліку реквізитів, а всі інші синонімічні його назви вилучити зі списку. Наприклад, реквізити *«верстат», «обладнання», «основний засіб», «пристрій»* є синонімічними, тому в переліку реквізитів замість групи потрібно залишати лише один реквізит «*обладнання*».

Складаючи список реквізитів, можливо, необхідно буде здійснити ще одну операцію, яка називається узагальненням. При цьому деяку сукупність реквізитів замінюють одним реквізитом, який узагальнює суть представленої групи. Наприклад, у переліку реквізитів є такі: *токар, фрезерувальник, слюсар* та ін. Цю групу можна замінити одним реквізитом «*спеціальність*». Отже, узагальненню підлягають реквізити, що мають однаковий формат і семантику.

2. Агрегація реквізитів і компонування їх у сутності. Процес агрегації реквізитів і визначення сутностей є ітеративним, тобто послідовним.

Для великих і складних систем з різними функціями і великою кількістю документації перелік реквізитів аналізують окремо за кожною функціональною задачею чи їх комплексом. В основі агрегації лежить аналіз типів співвідношень між реквізитами.

2.1. Спочатку серед реквізитів виділяються ті, між якими існує однозначний зв'язок в обох напрямах (1:1). Такі реквізити агрегуються в одну сутність, якій присвоюється унікальне ім'я. Наприклад, реквізити «*код матеріалу*», «*назва матеріалу*», «*одиниця вимірювання*» об'єднуються в одну сутність під назвою «*матеріал*». Важливим моментом є вибір імені сутності. Ідентифікація сутності якимось ім'ям залежить від того, якому об'єкту предметної області відповідає ця сутність. Виконуючи такий аналіз, іноді необхідно додатково уточнювати ім'я реквізита для однозначного його семантичного трактування.

При цьому слід пам'ятати, що в результаті виконання цих дій один і той самий реквізит може потрапити до кількох сутностей. Наприклад, код робітника може бути включений до сутності «ПРАЦІВНИК», яка містить усю довідкову інформацію про працюючих на підприємстві, а також до сутності «ВИРОБІТОК», що вміщує дані про щоденний виробіток кожного робітника.

2.2. Далі необхідно виконати перевірку на співвідношення 1:Б між рештою реквізитів і виділеними сутностями. При такому типі співвідношення в екземплярах сутності можуть дублюватися значення цього реквізита. Такі реквізити приєднуються до виділених сутностей.

2.3. Якщо після виконання описаного аналізу ще залишаться реквізити і серед них немає таких, які б були з виділеними сутностями у співвідношенні 1:1 чи 1:Б, то вирішують питання про створення з решти реквізитів окремих нових сутностей. Не виключена можливість, що при цьому може з'явитися сутність, яка містить лише один реквізит. Це свідчить про те, що існують недоліки в проектуванні на зовнішньому рівні. Тому потрібно

виконати дообстеження ПО з точки зору поповнення таких сутностей реквізитами.

Отримані сутності необхідно ідентифікувати унікальними іменами. Іменувати сутність краще одним словом і бажано, щоб це був іменник, наприклад «*ДЕТАЛЬ*», «*МАТЕРІАЛ*», «*ПОСТАЧАЛЬНИК*», або словосполучення з дієслова чи іменників (наприклад, «*ПРИБУТОК МАТЕРІАЛУ*», «*ВИДАТОК МАТЕРІАЛУ*»).

3. Вивчення реквізитів з неунікальними значеннями. Якщо ці реквізити представлені досить довгими текстами (текст_і, і = 1, 2…), то доцільно заміни їх на короткі кодові позначення. У цьому разі в інфологічну модель вноситься нова сутність, яку доречно назвати «ПЕРЕЛІК_текст_і». Вона складатиметься з коду реквізиту та його текстового значення. При цьому в усіх раніше виділених сутностях текстові значення замінюються на код.

4. Виділення запитів до БД і словесний їх опис. Щоб описати запити, необхідно дослідити процеси обробки інформації з кожної функціональної задачі модельованої ПО. Цей аналіз та формалізація процесу обробки інформації потрібні для визначення структурних зв'язків між сутностями, тобто вже на етапі інфологічного моделювання БД проектувальник має чітко уявляти алгоритм та логічну схему розв'язання кожної задачі. Запити виділяють опитуванням користувачів і з'ясуванням інформаційних потреб прикладних програм.

Концептуальне проектування. На цьому етапі здійснюється перехід від інфологічної моделі ПО до логічної концептуальної моделі на основі моделі даних, яка підтримується засобами конкретної СУБД. Перехід від інфологічної до концептуальної моделі називається відображенням.

Концептуальна модель являє собою базу даних, структуровану на логічному рівні й орієнтовану на конкретну СУБД. Перш ніж виконати концептуальне проектування, необхідно обрати СУБД. Кожна конкретна СУБД накладає ряд обмежень на побудову логічної моделі даних, тому насамперед необхідно вивчити специфіку й особливості СУБД, виявити всі фактори, які можуть вплинути на логічну модель БД.

Основними факторами, що впливають на концептуальне проектування з боку СУБД, є такі:

1. Тип логічної моделі, що його підтримує обрана СУБД. Нині найпоширенішими на ринку програмних засобів і в практиці автоматизації розрахунків є реляційні СУБД. Підтримування схеми бази даних реляційними СУБД здійснюється для забезпечення механізму посилкової цілісності бази даних.

2. Особливості фізичної організації даних у середовищі СУБД.

3. Кількісні обмеження з боку СУБД.

Усе ще не знайдено формалізованих методів, які б надавали змогу однозначно виконати концептуальне проектування. Тому його результат багато в чому залежить від уміння та рівня кваліфікації спеціалістів, які здійснюють проектні розробки.

У результаті концептуального проектування можна отримати кілька варіантів побудови логічної моделі даних. Тому важливим моментом є оцінка отриманих моделей і вибір найбільш оптимального варіанта.

Отриманий результат передусім потрібно оцінити з точки зору відповідності технологічних вимог. У разі невідповідності цим обмеженням потрібно здійснити перепроектування БД. Крім того, на отриманій моделі необхідно перевірити умови виконання всіх запитів користувачів і вимог прикладних програм, тобто умову адекватності логічної моделі інформаційній моделі предметної області.

Відображення інфологічної моделі бази даних на реляційну модель СУБД. У середовищі СУБД реляційного типу база даних зберігається у вигляді файлів, що називаються реляційними відношеннями. З іншого боку, вони представлятимуться у вигляді структурованих таблиць. Тому при відображенні інфологічної моделі на реляційну СУБД сутності потрібно трансформувати в реляційні відношення, врахувавши такий момент.

Якщо між сутностями існує зв'язок 1:1 і сутності семантично споріднені, то теоретично можливо об'єднати їх в одне реляційне відношення. Таке об'єднання може прискорити пошук при реалізації запитів.

При відображенні на реляційну модель перепроектовувати сутності, якщо не було допущено помилок на етапі інфологічного проектування, не є проблемним. Тобто сутності інфологічної моделі стають реляційними відношеннями. Необхідно лише перевірити виконання таких умов.

1. Усі атрибути відношень мають бути атомарними (неподільними).

2. Відношення не повинно мати дублюючих рядків і стовпчиків.

3. Усі атрибути у відношенні повинні мати унікальні імена. При відображенні на реляційну модель слід ураховувати особливість конкретної реляційної СУБД.

4. Обов'язковою умовою встановлення зв'язку між двома реляційними відношеннями є наявність у них хоча б одного спільного атрибута, що є ключем, за яким виконується зв'язок.

Проектування БД на прикладі. Проектування БД на інфологічному, концептуальному та внутрішньому рівні проводиться для об'єкта дослідження, який представлений вихідним документом і запитами (рис. 3.1).

Відомість *про викладачів і студентів вишу*

Освітній процес (суб'єкти)						
ПІП	*Викладач*			*Студент*		
	Вчений ступінь	*К-наукових праць*	*Заробітна плата*	*Стипендія*	*Середній бал*	*Форма навчання*

Запити до бази даних

(1 – визначити кількість викладачів із заробітною платою менше 4 500 грн; 2 – визначити студентів з денною формою навчання та із середнім балом менше 4.0; 3 – в якому журналі була надрукована перша стаття викладача, який має найбільшу кількість наукових праць)

Рис. 3.2. Документ і запити, на основі яких треба розробити проект БД

Для початку всі реквізити консолідуються в одну сутність СУБ'ЄКТ: проект 1в/1с/2р БД, де *в* – номер варіанта; *с* – номер

сутності; *ст* – номер структурованої таблиці; *т* – номер таблиці даних; *р* – рівень архітектури моделей подання даних (рис. 3.3).

№ з/п	Повна назва реквізиту	Ідентифікатор	Формат значення
1	ПІП викладача/студента	PIB	A(40)
2	Вчена ступінь викладача	ID_stupinj	X(20)
3	Кількість наукових праць викладача	ID_kil_pracj	9(4)
4	Заробітна плата викладача	ID_ZP	9(5).9(2)
5	Стипендія студента	ID_stypendy	9(4).9(2)
6	Середній бал студента	ID_Savg	9(3).9(2)
7	Форма навчання студента	ID_forma	A(20)

Рис. 3.3. Проект 1в/1с/2р БД: СУБ'ЄКТ

Для аналізу структури БД Суб'єкт розглянемо 4-й рівень подання даних – у середовищі СУБД (рис. 3.4).

№ з/п	PIB	ID_ Stupinj	ID_kil pracj	ID_ ZP	ID_ stypend	ID_S avg	ID_ forma
1	Скрипка	канд.техн.наук	45	5 000	–	–	–
2	Петренко	к.т.н.	34	3 900	–	–	–
3	Горобець	–	–	–	700	4.5	денна
4	Мокренко	–	–	–	870	3.8	денна

Рис. 3.4. Проект 1в/1т/4р БД: Таблиця даних (СУБ'ЄКТ)

Проект БД потребує доопрацювання, оскільки деякі поля є незаповненими – ділимо сутність СУБ'ЄКТ на сутності ВИКЛАДАЧ і СТУДЕНТ (рис. 3.5).



Рис. 3.5. Проект 2в/2р/1с-2с БД: Сутності (*Викладач, Студент*)

Проект БД містить 2 сутності, що пов'язані між собою. Побудуємо концептуальну модель БД, для чого кожній сутності ставимо у відповідність структуровану таблицю. При цьому доповнимо атрибутами зв'язку *Місце роботи* і *Місце навчання*, відповідно, обидві структуровані таблиці (рис.3.6).

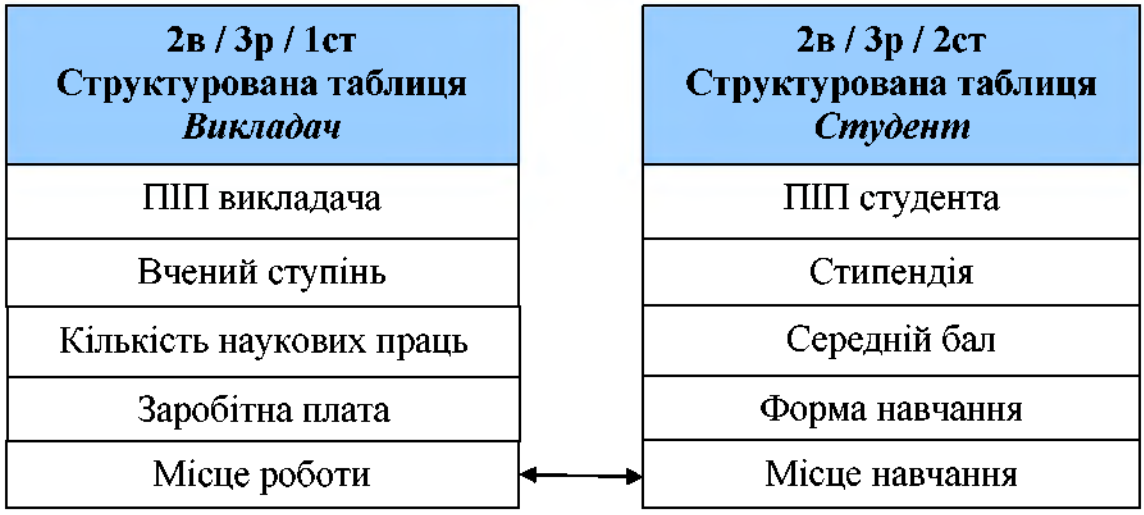


Рис. 3.6. Проект 2в/3р/1ст-2ст БД: ВИКЛАДАЧ, СТУДЕНТ

Структуровані таблиці ВИКЛАДАЧ і СТУДЕНТ нашого проекту БД містять текстові атрибути (*Вчений ступінь, Місце роботи, Форма навчання, Місце навчання*), кожен з яких доцільно замінити *Кодом_текстового_ атрибута* й організувати за ним зв'язок відповідно до *ТЕКСТОВИЙ АТРИБУТ {код текстового атрибута, назва текстового атрибута}*.

Структуровані таблиці ВИКЛАДАЧ і СТУДЕНТ містять також розрахункові атрибути, що не представляють актуальну характеристику об'єкта дослідження, а тому необхідно створити окремі структуровані таблиці з первинних даних, які дозволять їх встановити.

Достатньо актуальна ситуація в умовах системи освіти, коли викладач є слухачем в інститутах підвищення кваліфікації, а студенти працюють у статусі викладача. Текстові атрибути *ПІП_викладача* і *ПІП_студента* замінимо *Кодом_текстового_атрибуту,* за яким організуємо зв'язок зі ст. *ОСОБА {Ідентифікаційний_№_особи, ПІП_особи, інші дані особи}*.

У результаті проведення таких проекцій ми отримаємо концептуальний проект 3в/3р/1ст-8ст БД «Успішність» (рис.3.7).

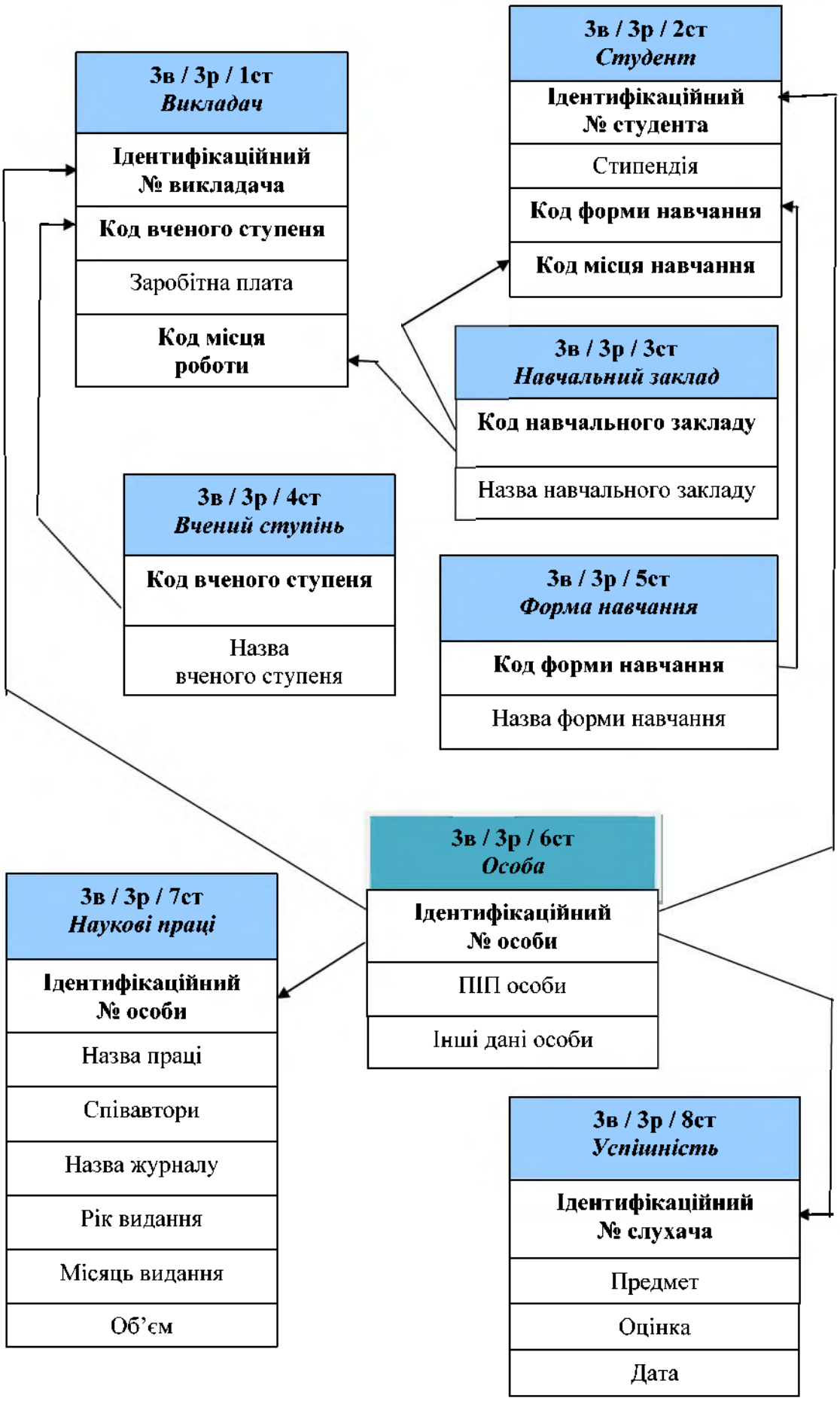


Рис. 3.7. Проект 3в/3р/1-8ст БД «Успішність»

3.5. Лабораторна робота 4

Тема. Створення бази даних, інтерфейсу доступу мовними та інструментальними засобами

Мета роботи:

✓ ознайомитися з особливостями розробки інфологічної та концептуальної моделей БД на прототипі;

✓ закріпити навички розробки інфологічної та концептуальної моделей БД відповідно до індивідуального завдання;

✓ ознайомитися з особливостями реалізації проекту БД у середовищі різних СУБД на основі розробленої концептуальної моделі БД;

✓ ознайомитися з особливостями розробки звіту на мовах запитів, відповідно до 1, 2 та 3-го запитів, і отримати позитивний результат;

✓ ознайомитися з особливостями створення інтерфейсу доступу до БД з використанням форм на основі таблиці та запиту для:

– введення даних у таблиці БД без прямого доступу до самих таблиць;

– виведення результатів роботи запитів у вигляді необхідних звітних документів.

Завдання

1. Ознайомитися на прототипі з особливостями розробки інфологічної та концептуальної моделей БД для документа «Освітній процес (суб'єкти)» та запитів.

2. Виробити навички розробки для досліджуваного об'єкта інфологічної та концептуальної моделей БД, яка дозволить отримати заданий документ і запити відповідно до індивідуального завдання.

3. Відповідно до розробленого проекту реалізувати структуру БД у СУБД Access (MS SQL Server) та порівняти отримані результати, щоб можна було надати позитивну відповідь на запити індивідуального завдання.

4. На основі створеної БД розробити звіти мовами запитів SQL і QBE, отримати на них позитивну відповідь.

5. Розробити інтерфейс доступу до БД з використанням форм на основі таблиці і запиту для:

– введення даних у таблиці БД без прямого доступу до самих таблиць;

– виведення результатів роботи запитів у вигляді необхідних звітних документів.

Хід роботи

1. Ознайомитися на прототипі з особливостями розробки інфологічної та концептуальної моделей БД для документа «Освітній процес (Суб'єкти)» та запитів.

2. Розробити інфологічну та концептуальну моделі БД, яка дозволить отримати заданий документ і запити відповідно до індивідуального завдання.

3. Відповідно до розробленого проекту реалізувати структуру БД в СУБД Access у власній папці з іменем «ПІП_студента_№групи» та наповнити її 10–15 записами такими даними, щоб можна було надати позитивну відповідь на запити індивідуального завдання.

4. На основі створеної БД розробити звіт мовою запитів QBE в середовищі СУБД, відповідно до 1, 2 та 3-го запитів, і отримати позитивний результат.

5. На основі створеної БД розробити звіт мовою запитів SQL за допомогою інструментальних засобів MSQRY32 або DataBase Tour Pro, відповідно до 1, 2 та 3-го запитів, і отримати позитивний результат.

6. Розробити інтерфейс доступу до БД з використанням форм на основі таблиці і запиту для: а) введення даних у таблиці БД без прямого доступу до самих таблиць; б) виведення результатів роботи запитів у вигляді необхідних звітних документів.

7. Відповідно до розробленого проекту реалізувати структуру БД у СУБД MS SQL Server і порівняти отримані результати.

8. Звіт про виконану роботу оформляється відповідно до «Загальних рекомендацій до виконання лабораторних робіт та оформлення звіту» (додаток А, А. 1, А. 2).

3.6. Робота в MS Access 2016 зі створення БД

Методичні вказівки для виконання завдання

Запустити СУБД. Після запуску MS Access головне вікно програми виглядатиме приблизно так, як показано на рис. 3.8.

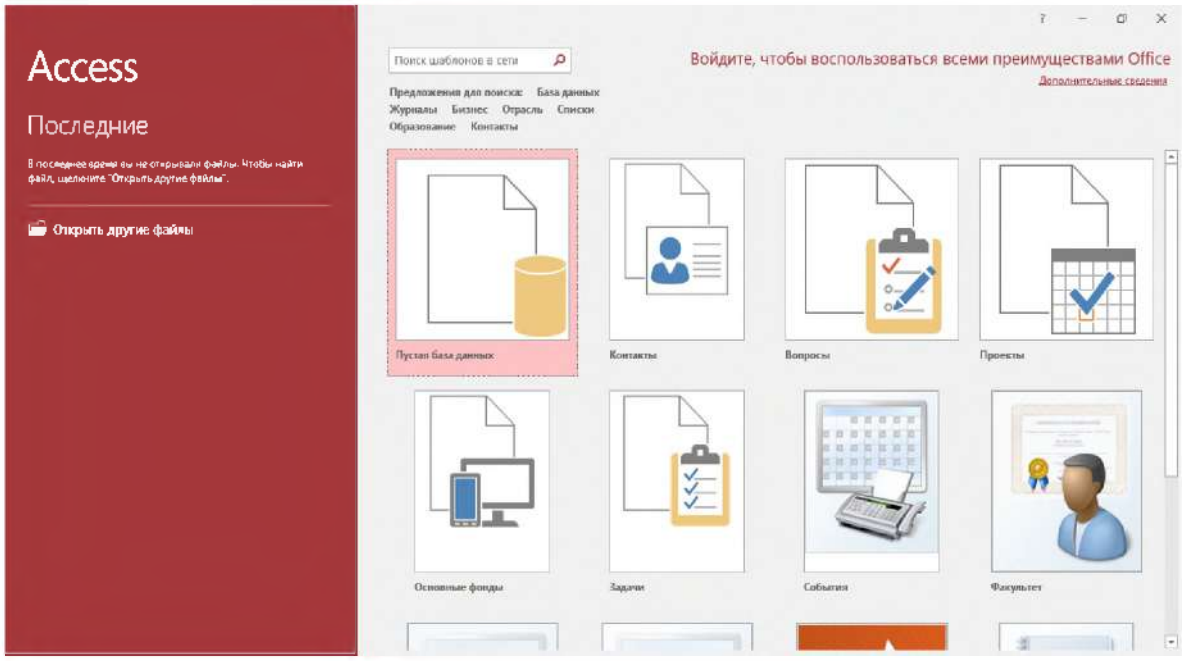


Рис. 3.8. Головне вікно СУБД MS Access 2016

На панелях інструментів можуть відображатися як загальні команди для всіх компонентів (робота з буфером обміну, друк та ін.), так і специфічні операції для певних умов.

Способи створення бази даних

Нову базу можна створити декількома способами:

✓ створення порожньої бази даних з подальшим ручним заповненням;

✓ створення бази шляхом копіювання даних з наявної бази;

✓ створення за допомогою шаблону.

Створення нової бази з існуючого файла. Це найпростіший спосіб створення БД. Для створення нової бази необхідно лише вказати ім'я існуючого файла. При цьому всі дані існуючої бази будуть просто скопійовані в нову базу і за необхідності перетворені в новий формат.

Створення нової бази за допомогою шаблону. У стандартному постачанні MS Access є одинадцять загальних шаблонів, призначених для створення найпоширеніших видів баз даних. Додаткові шаблони можна викачати з сайту фірми Microsoft прямо з програми за допомогою пошукового рядка ***Пошук шаблонів у мережі*** (рис. 3.9).

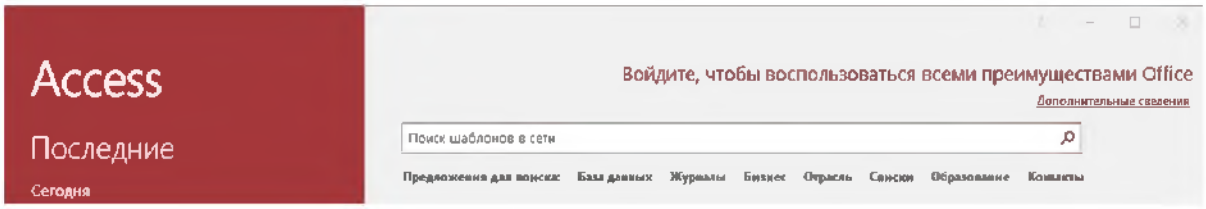


Рис. 3.9. Рядок пошуку шаблонів у мережі

Для прикладу оберемо шаблон Проекти. Після запиту на створення файла з'явиться вікно «***Проекти»,*** необхідно вказати ім'я БД і натиснути «Створити», з'явиться шаблонний проект бази даних.

Створення пустої бази даних. Для створення нової бази даних виберемо пункт ***Пуста база даних*** у головному вікні програми.

Для створення бази даних ручним способом розглянемо концептуальну модель БД **«Суб'єкти освітнього процесу»** (рис. 3.6). БД містить дві таблиці ***Викладач*** і ***Студент***.

Тепер можна приступити до створення таблиць, форм і звітів.

Створення таблиць. Доступ до таблиць здійснюється вибором розділу ***Таблиці*** на вкладці ***Створити***.

Нову таблицю можна створити двома способами: у режимі конструктора; шляхом введення даних.

Створення нової таблиці в режимі конструктора. Створення таблиці в режимі конструктора є, по суті, ручним введенням усіх даних про таблицю. Виберемо пункт ***Конструктор таблиць*** у вкладці ***Створити*** і запустимо його одинарним клацанням миші. На екрані з'явиться вікно конструктора.

Введемо поля таблиці ***Викладач*** у рядки вікна конструктора і вкажемо їх тип і опис. За умовчанням тип даних для нового поля – текстовий. Для його зміни досить обрати потрібний тип зі списку, що розкривається, в цьому осередку.

Для створення первинного ключа оберемо рядок з потрібним полем, натиснемо праву кнопку миші й оберемо пункт ***Ключове Поле*** з випадного меню. Зліва від рядка повинен з'явитися значок ключа. Саме так позначаються первинні ключі в MS Access.

Нижче таблиці полів у вікні конструктора знаходиться панель властивостей поточного поля. Більшість властивостей зрозуміла з назви. Введемо в осередок ***Підпис*** докладнішу назву кожного поля. Тепер ці дані відображатимуться у формах і звітах замість імен полів.

Після введення всіх даних закриємо вікно конструктора і на запит про збереження відповімо позитивно. У вікні введення імені таблиці, що з'явилося, наберемо ***Викладач***. Створена таблиця повинна з'явитися в поточному розділі (рис. 3.10).

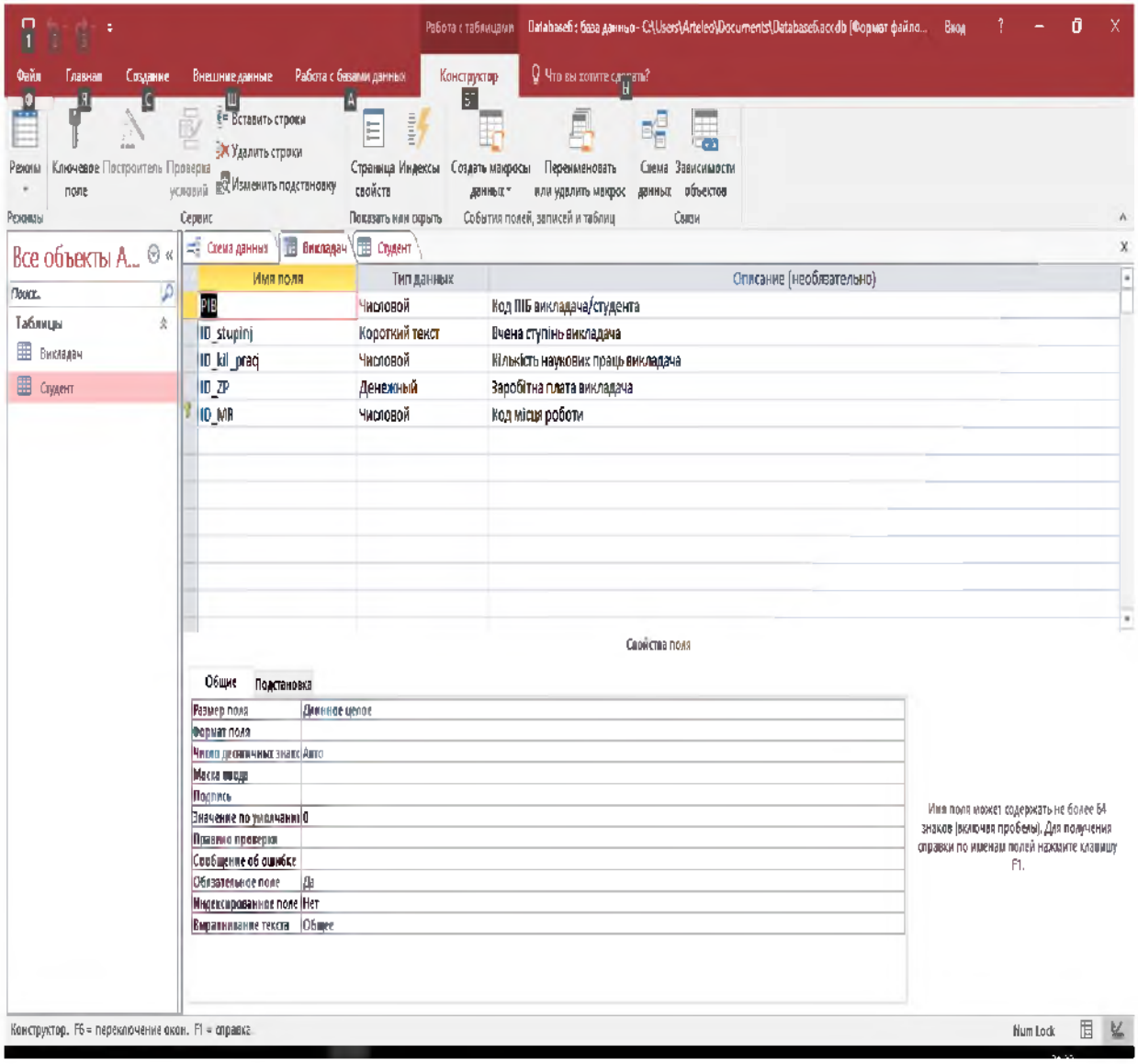


Рис. 3.10. Структура таблиці *Викладач* у СУБД MS Access 2016

Аналогічно створимо таблицю ***Студент*** (рис. 3.11).

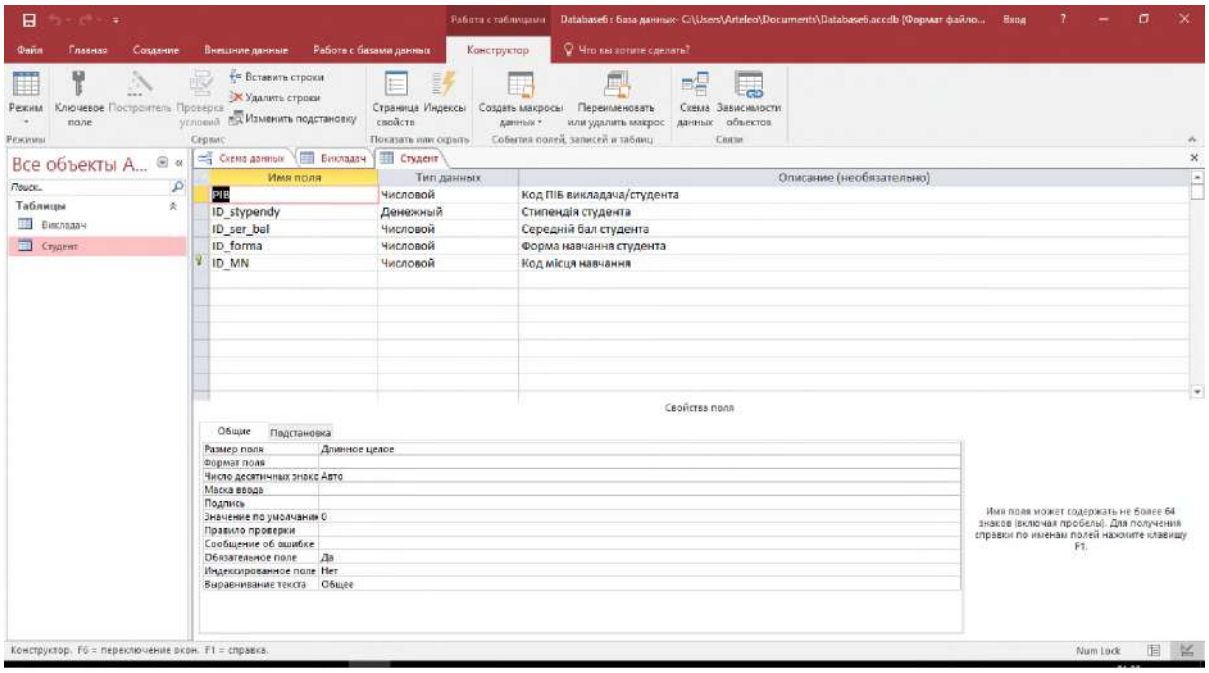


Рис. 3.11. Структура таблиці *Студент* в СУБД MS Access 2016

Створення зв'язків між таблицями. Access володіє гнучким і зручним візуальним засобом створення і редагування зв'язків між таблицями. Для його активації оберемо ***Конструктор | Схема Даних***. Додамо у вікно, що відкрилося, всі наявні таблиці.

Для організації зв'язку між таблицями ***Викладач*** і ***Студент*** клацнемо мишею на полі ***Id_MR*** таблиці ***Викладач,*** перетягнемо його на поле з назвою ***Id_MN*** таблиці ***Студент***. З'явиться вікно зміни зв'язків. У центральній таблиці вказані поля головної і зв'язаної таблиць. Встановимо прапорець ***Забезпечення цілісності даних***, що означає жорсткий зв'язок між таблицями, і прапорці каскадного видалення й оновлення даних.

Аналогічно можна зв'язати решту таблиць бази даних. У результаті повинна появитися схема бази даних, яка представлена на рис. 3.12.

Закриємо вікно і збережемо всі зміни. Створені зв'язки можна перевірити тестовим введенням даних. При введенні перших даних у головну таблицю зліва з'явиться поле з хрестиком у центрі. Натиснувши на нього, можна вводити дані в підлеглу таблицю. При видаленні запису в головній таблиці видаляться і всі пов'язані записи підлеглих таблиць.

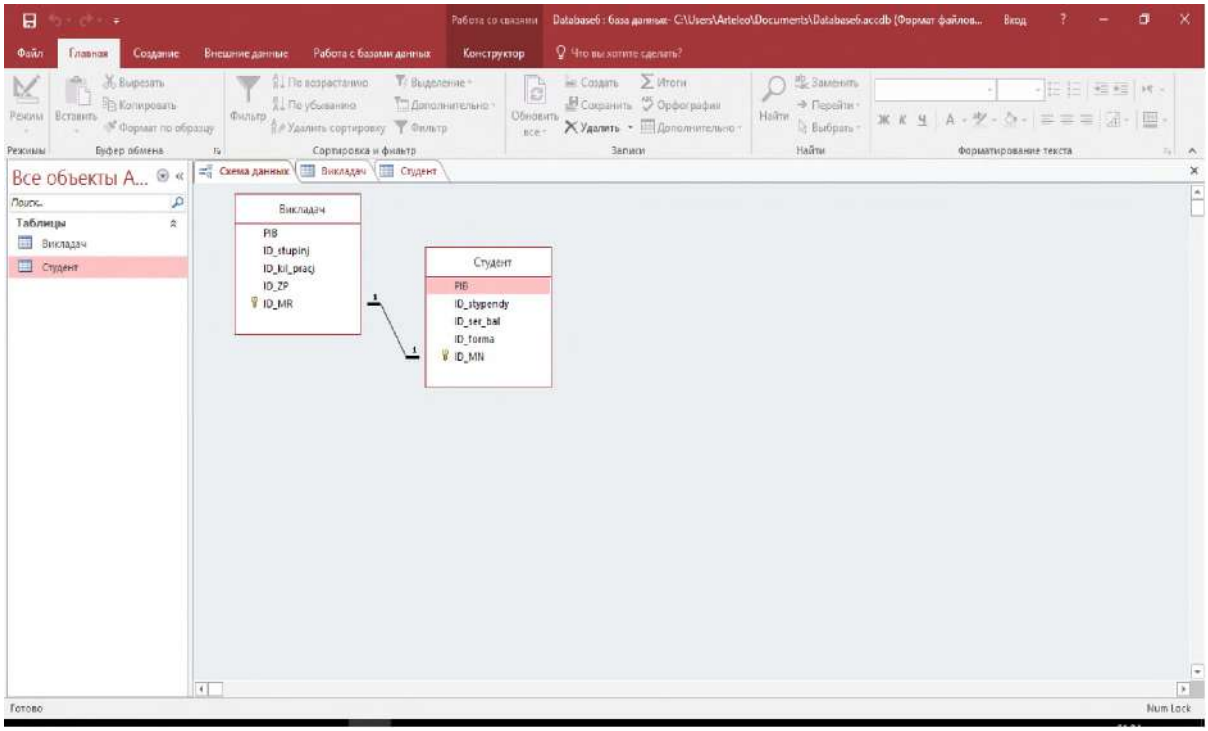


Рис. 3.12. Схема бази даних «Суб'єкти освітнього процесу»

Створення запитів. Запити є основним інструментом управління базами даних. У Access для створення запитів зручно використовувати візуальні засоби. Робота в них здійснюється завданням зразків значень у шаблоні запиту, що передбачає той тип доступу до бази даних, який потрібен у цей момент.

Для відображення візуальних засобів створення потрібно перейти в розділ ***Створення*** і запустити ***Конструктор запитів***. З'явиться вікно конструктора (рис. 3.13) з вікном додавання таблиць і запитів.

Поля запиту. Таблиця полів запиту потрібна для введення полів, умов відбору, групових і агрегатних функцій, способу сортування.

Будь-яке поле в рядку ***Поле*** містить список доступних полів усіх таблиць, що беруть участь у запиті.

Поля в рядку ***Ім'я таблиці*** також мають список, що розкривається і містить імена доступних таблиць.

Поля в рядку ***Сортування*** призначені для вибору способу сортування записів у запиті. Виведення поля на екран можна виключити, виключивши значок ***Виведення на екран***.

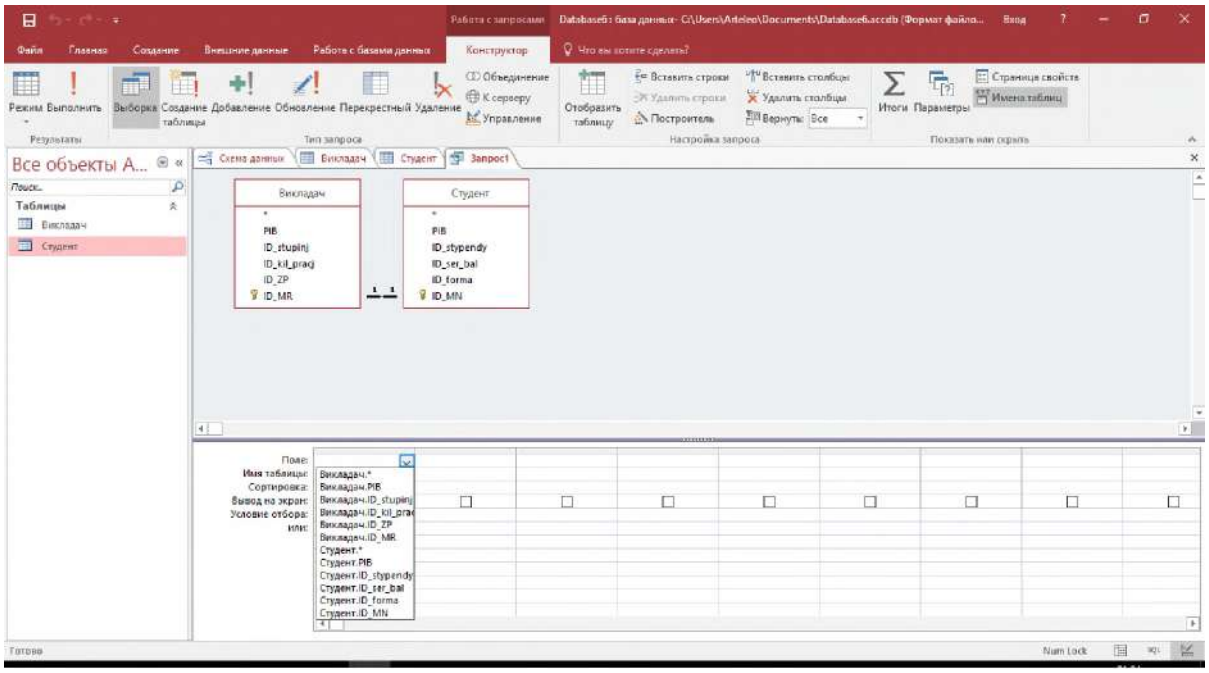


Рис. 3.13. Конструктор запитів

У полі ***Умова відбору*** можна ввести будь-яку умову, що стосується цього поля.

Наприклад: ***«>10», «BETWEEN 1 AND 10».***

Якщо умова є рівністю, то знак *«=»* можна не вводити, залишивши лише параметр для порівняння.

З випадного меню на полях запиту доступний пункт ***Групові операції***, при виборі якого в таблиці полів з'являється ще один рядок: ***Групова операція***. У полях цього рядка можна обрати угрупування полів таблиці, додаткові умови відбору, вирази (наприклад, вкладені команди вибору) й агрегатні функції.

Приклад створення однотабличних запитів. Нехай потрібно:

«вивести на екран список студентів з денною формою навчання, середній бал яких вищий від 3-х балів; середній бал студента включити у звіт».

Допустимо, що код денної форми навчання дорівнює 1. Для формування списку здійснимо його сортування.

Запустимо конструктор запитів і додамо в нього таблицю *Студент*.

Таблиця повинна з'явитися у вигляді віконця зі списком полів. Перенесемо поля ***PIB, ID_ser_bal*** і ***ID_forma*** в таблицю

полів. Поставимо для поля ***ID_ser_bal*** умову ***>3***, що означає *середній бал* вище 3; умова для коду форми навчання студента буде *ID_forma=1*, а також передбачимо сортування.

Для створення ***SQL***-аналога запиту необхідно перейти в ***Конструктор | Режим | Режим SQL***. SQL запит виглядатиме таким чином:

SELECT Студент.PIB, Студент.ID_ser_bal
FROM Студент
WHERE Студент. ID_ser_bal>3 and ID_forma=1
ORDER BY Студент.PIB;

Закриємо і збережемо запит. Тепер при його відкритті (рис. 3.14) можна не тільки переглядати дані таблиці, але і вводити їх. Ця функція доступна для всіх однотабличних запитів, в яких присутні всі обов'язкові поля.

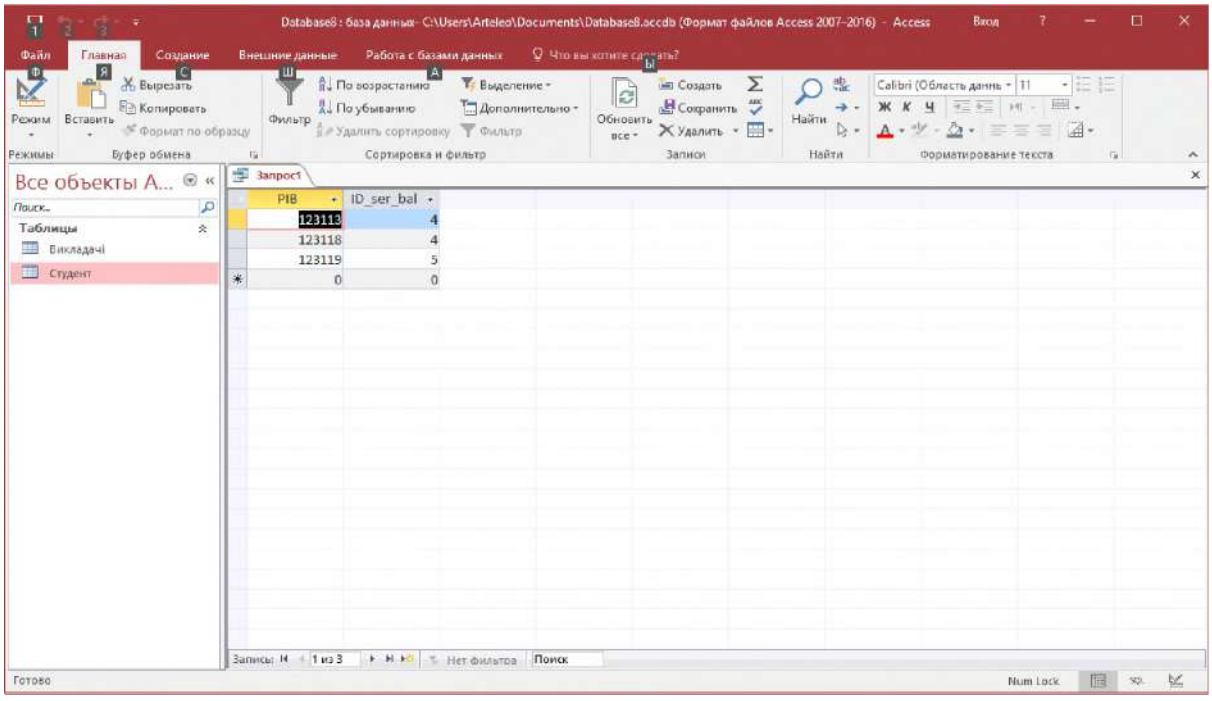


Рис. 3.14. Результат виконання однотабличного запиту

Створення запитів на зміну даних. Запити на оновлення даних служать для створення, зміни, видалення груп записів з однієї або декількох таблиць.

Існує чотири типи запиту на оновлення даних.

1. Створення таблиці. *Створення нової таблиці з даними, одержаними з інших таблиць.*

2. Оновлення. *Зміна груп записів в одній або декількох таблицях.*

3. Додавання. *Додавання груп записів у таблицю з інших таблиць.*

4. Видалення. *Видалення записів з однієї або декількох таблиць.*

Єдиною відмінністю запиту на *створення таблиць* від запиту на *вибірку* полягає в тому, що результат вибірки буде збережений у заздалегідь створеній таблиці, ім'я якої буде запрошено при завданні типу запиту.

При запиті на *оновлення даних* з'явиться нове поле з назвою ***Оновлення***, в яке слід поміщати дані для заміни.

При запиті на *додавання* необхідно вказати назви полів, в які поміщатимуться результати вибірки.

Запит на *видалення* передбачає видалення записів за заданої умови з однієї таблиці або з декількох, якщо при цьому у зв'язках між таблицями встановлено каскадне видалення.

При роботі в конструкторі створення запитів на зміну даних проводиться в спеціальному режимі.

Для цього необхідно вибрати стрілку в пункті ***Тип запиту*** випадного меню і вказати на «*оновлення*».

Слід звернути увагу на те, що при цьому в таблиці полів зникли ***Угрупування, Виведення на екран*** і ***Сортування.***

Проте додався новий рядок – ***Оновлення***.

Додавши до полів запиту поле ***Назва***, введемо в поле ***Оновлення*** рядок ***Миша***, а в полі ***Умова відбору – Маніпулятор***. Після збереження запиту при його запуску всі знайдені записи ***Миша*** в таблиці будуть замінені на ***Маніпулятор***.

Створення форм

Форми в Access слугують для зручного відображення даних з однієї чи декількох таблиць і введення даних користувачем. Можна одержати інформацію про форми, що містяться в базі даних, натиснувши кнопку з однойменною назвою у вікні бази даних. Створити нову форму можна всього двома способами: за допомогою конструктора і майстра.

Створення форм за допомогою майстра є найпоширенішим

способом створення форм завдяки простоті й зручності використання.

Створення звітів

Організація процесу створення звітів є одним із основних переваг Access. Широкі можливості зі зміни властивостей дозволяють створювати найрізноманітніші звіти. У новій версії з'явилася можливість створення безпаперових звітів.

У Access всі звіти поділяються на шість категорій:

1. Одностовпчикові звіти. *У одному довгому стовпчику перераховуються значення всіх полів таблиці або запиту.*

2. Стрічкові звіти. *Значення кожного запису поміщаються в окремий рядок.*

3. Багатостовпчикові звіти. *Вони створюються з одностовпчикових звітів. Інформація, що не поміщається в перший стовпець, переходить у верхню частину другого стовпця та ін.*

4. Звіти з угрупуванням даних. *Дозволяють обчислювати підсумкові значення для груп записів і подають інформацію в зручному для використання вигляді.*

5. Поштові наклейки. *Використовуються для друку імен і адрес.*

6. Вільні звіти. *Містять підлеглі звіти, при цьому кожен підлеглий звіт створюється на основі незалежних джерел даних.*

Контрольні запитання

1. Які є способи створення бази даних?
2. Як відбувається створення нової бази з існуючого файла?
3. Як відбувається створення нової бази за допомогою шаблону?
4. Як відбувається створення нової таблиці в режимі конструктора?
5. Як відбувається створення таблиць за допомогою майстра?
6. Як відбувається створення зв'язків між таблицями?
7. Як відбувається створення запитів на зміну даних?
8. Як відбувається створення форм?
9. Які бувають звіти?
10. Як відбувається створення звітів залежно від їх категорії?

Оформлення звіту

Звіт про виконану роботу оформляється відповідно до «Загальних рекомендацій до виконання лабораторних робіт та оформлення звіту» (додаток А).

Завдання для моделювання

1. Для досліджуваного об'єкта розробити проект БД, який дозволить отримати відомість:

ВІДОМІСТЬ 1

розрахунку відрядної індивідуальної заробітної плати

за цехом _____ за _____ місяць 201__ р.

код та назва цеху

Таб номер	Прізвище	Код деталі	Розряд	Професія		Кількість виготовлених деталей	Розцінка за деталь	Відроблений час	Сума заробітної плати
				Код	Назва				

ВІДОМІСТЬ 2

потреби в матеріалах на річну виробничу програму

за підприємством на 201__ р.

Код матеріалу	Назва матеріалу	Одиниці вимірювання	Код виробу	Назва виробу	Програма випуску виробів на рік	Норми витрат матеріалу на виріб	Потреба матеріалу на річну програму

ВІДОМІСТЬ 3

обліку руху матеріалів на складі ___ за ___ місяць

код та назва складу

Матеріал			Залишок Вхідний		Надходження		Видатки		Залишок вихідний	
Код	Назва	Одиниця виміру	Кількість	Вартість	Кількість	Вартість	Кількість	Вартість	Кількість	Вартість

ВІДОМІСТЬ 4

обліку якості готової продукції за цехом ___ за __ місяць

код та назва цеху

Номер дільниці	Деталь		Операція		Вид браку		Причина (винуватець)		Кількість деталей	
	Код	Назва	Код	Назва	Код	Назва	Код	Назва	Годних	Бракованих

ВІДОМІСТЬ 5

відвантаження готової продукції за _____ місяць

Код покупця	Назва покупця	Код продукції	Назва продукції	№ угоди на відвантаження	Кількість за угодою	Фактично відвантажено	Відхилення

2. Створення SQL-запитів

для доступу до БД «Студент» (*Додаток В*)

1. Створити запит, результатом якого є список студентів, що успішно навчаються та в яких помилково перед прізвищем або іменем і по батькові можуть бути пробіли; запит має повертати такі поля: прізвище, ім'я, по батькові.

2. Створити запит, результатом якого є список студентів IV курсу навчально-наукового інституту ІТтаМ, які були присутніми на останній модульній контрольній роботі; запит має повертати такі поля: прізвище, ім'я, по батькові й групу.

3. Створити запит, який повертає середній бал оцінки на всіх контрольних роботах студентів IV курсу інституту ІТтаМ.

4. Створити запит, результатом якого є список студентів IV курсу навчально-наукового інституту ІТтаМ, які одержали на підсумковій контрольній роботі оцінку, яка вище середньої оцінки успішності по інституту; запит має повертати такі поля: прізвище, ім'я, по батькові й оцінку.

5. Створити запит, результатом якого є список студентів IV курсу інституту ІТтаМ, які одержали на всіх контрольних роботах оцінку 5 балів; запит має повертати такі поля: прізвище, ім'я, по батькові і групу.

6. Створити запит, результатом якого є список першокурсників, які успішно навчаються і вік яких не перевищує 18 років; список має бути упорядкований в алфавітному порядку; запит має повертати такі поля: прізвище, ім'я, по батькові, групу і вік студента.

7. Створити запит, який повертає середній бал оцінки за модульну контрольну роботу та кількість відсутніх на ній студентів у групах.

8. Створити запит, результатом якого є список студентів III курсу навчально-наукового інституту ІТтаМ, які або вступили до університету через 1 рік після закінчення школи, або брали академічну відпустку; запит має повертати такі поля: прізвище, ім'я, по батькові і вік студента.

ЧАСТИНА 2
МЕТОДОЛОГІЇ СТРУКТУРНОГО АНАЛІЗУ І ПРОЕКТУВАННЯ

РОЗДІЛ 4
МОДЕЛЮВАННЯ ЗА ДОПОМОГОЮ CASE-ЗАСОБУ ALLFUSSION PROCESS MODELLER ПАКЕТА ALLFUSSION MODELING SUITE COMPUTER ASSOCIATES

4.1.Теоретичні аспекти методології IDEF0

Опис системи за допомогою IDEF0 називається функціональною моделлю. Функціональна модель призначена для опису існуючих бізнес-процесів, у якому використовуються як природні, так і графічні мови. Для передачі інформації про конкретну систему джерелом графічної мови є сама методологія IDEF0.

Методологія IDEF0 пропонує побудову ієрархічної системи діаграм – одиничних описів фрагментів системи. Спочатку проводиться опис системи в цілому і її взаємодії з навколишнім світом (*контекстна діаграма*), після чого проводиться функціональна декомпозиція – система розбивається на підсистеми і кожна підсистема описується окремо (*діаграми декомпозиції*). Потім кожна підсистема розбивається на більш дрібні і так далі до досягнення потрібного рівня деталізації.

Кожна IDEF0-діаграма містить функціональні блоки та інтерфейсні дуги. Блоки зображують функції модельованої системи. Дуги зв'язують блоки і відображають взаємодії та взаємозв'язки між ними.

Функціональні блоки (*роботи*) на діаграмах зображуються прямокутниками, що означають пойменовані процеси, функції або завдання, які реалізуються впродовж певного часу і відображають відповідні результати. Ім'я роботи повинно бути виражене віддієслівним іменником, що означає дію.

IDEF0 вимагає, щоб у діаграмі було не менше трьох і не більше шести блоків. Ці обмеження підтримують складність діаграм і моделі на рівні, доступному для читання, сприйняття і використання.

Кожна сторона блоку має особливе цілком певне призначення:

- ✓ліва сторона блоку призначена для входів;
- ✓верхня – для управління;
- ✓права – для виходів;
- ✓нижня – для механізмів.

Таке позначення відображає певні системні принципи: входи перетворяться у виходи, управління обмежує або пропонує умови виконання перетворень, механізми показують, що і як виконує функція.

Блоки в IDEF0 розміщаються за ступенем важливості, як її розуміє автор діаграми. Цей відносний порядок називається домінуванням. Домінування сприймається як вплив, який один блок робить на інші блоки діаграми. Наприклад, домінуючим блоком діаграми може бути або перший з необхідної послідовності функцій, або плануюча чи контролююча функція, що впливає на всі інші.

Найбільш домінуючий блок зазвичай розміщається у верхньому лівому куті діаграми, а найменш домінуючий – у правому куті.

Розташування блоків на сторінці відображає авторське визначення домінування. Таким чином, топологія діаграми показує, які функції впливають на інші. Щоб підкреслити це, аналітик може перенумерувати блоки відповідно до порядку їхнього домінування. Порядок домінування може позначатися цифрою, розміщеною в правому нижньому куті кожного прямокутника і вказуватиме на найбільше домінування або на наступне.

Взаємодія робіт із зовнішнім світом і між собою відображається за допомогою інтерфейсних дуг. Інтерфейсна дуга (потік, стрілка, Arrow) показує елемент системи, який обробляється функціональним блоком або здійснює деякий вплив на функцію, що подана таким функціональним блоком. Графічним відображенням

інтерфейсної дуги є однонаправлена стрілка. Кожна інтерфейсна дуга повинна мати своє унікальне ім'я (Arrow Label).

За допомогою інтерфейсних дуг передають такі об'єкти:

1) матеріальні потоки (*деталі, товари, сировина …*);

2) фінансові потоки (*готівкові, безготівкові, інвестиції…*);

3) інформаційні потоки (*інформація, дані про наміри, усні розпорядження …*);

4) ресурси (*співробітники, інструментальні засоби, станки, пристрої…*).

У різних випадках як інтерфейсні дуги відображаються:

- ✓ вхідними і вихідними дугами – всі види об'єктів;
- ✓ управляючими дугами – лише потоки документів та інформації;
- ✓ механізмами – лише ресурси.

В IDEF0 розрізняють п'ять типів стрілок: вхід, управління, вихід, механізм, виклик.

Вхід – об'єкти, що використовуються й перетворюються під час роботи для одержання результату (виходу). Допускається, що робота може не мати ні однієї стрілки входу. Стрілка входу малюється як вхідна в ліву грань роботи.

Управління – інформація, що управляє діями роботи. Управляючі стрілки несуть інформацію, яка вказує, що повинна виконувати робота. Кожна робота повинна мати хоча б одну стрілку управління, що зображується як вхідна у верхню грань роботи.

Вихід об'єкти, у які перетворяться входи. Кожна робота повинна мати хоча б одну стрілку виходу, що малюється як вихідна із правої грані роботи.

Механізм – ресурси, що виконують роботу. Стрілка механізму малюється як вхідна в нижню грань роботи. З рішенням аналітика стрілки механізму можуть не зображуватися на моделі.

Виклик – спеціальна стрілка, що вказує на іншу модель роботи (рис. 4.1).

Стрілка виклику малюється як вихідна з нижньої частини роботи і використовується для відображення того, що деяка робота виконується за межами модельованої системи.

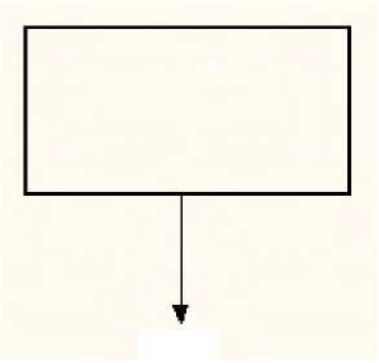


Рис. 4.1. Стрілка виклику

У методології IDEF0 потрібно тільки п'ять типів взаємодій між блоками для опису їхніх відношень: *управління, вхід, зворотний зв'язок з управління, зворотний зв'язок з входу, вихід-механізм.*

Зв'язки із входу й управління є найпростішими, оскільки вони відображають прямі впливи, які інтуїтивно зрозумілі і досить прості (рис. 4.2, 4.3).

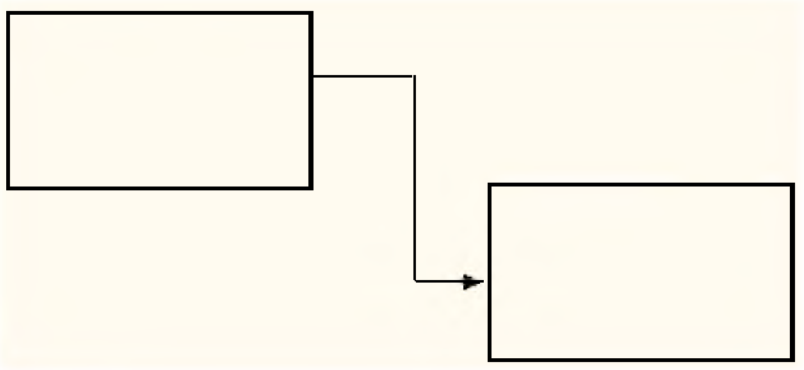


Рис. 4.2. Зв'язок з виходу

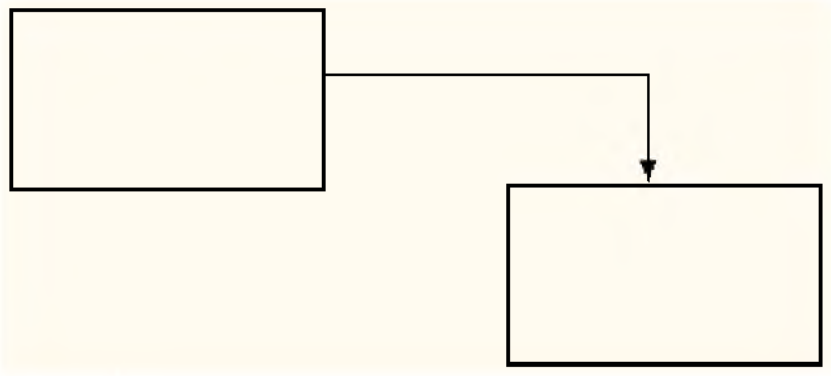


Рис. 4.3. Зв'язок з управління

Зв'язок з управління виникає тоді, коли вихід одного блоку безпосередньо впливає на блок з меншим домінуванням.

Зворотний зв'язок із входу і зворотний зв'язок з управління є більш складними, оскільки являють собою ітерацію або рекурсію (рис. 4.4, 4.5).

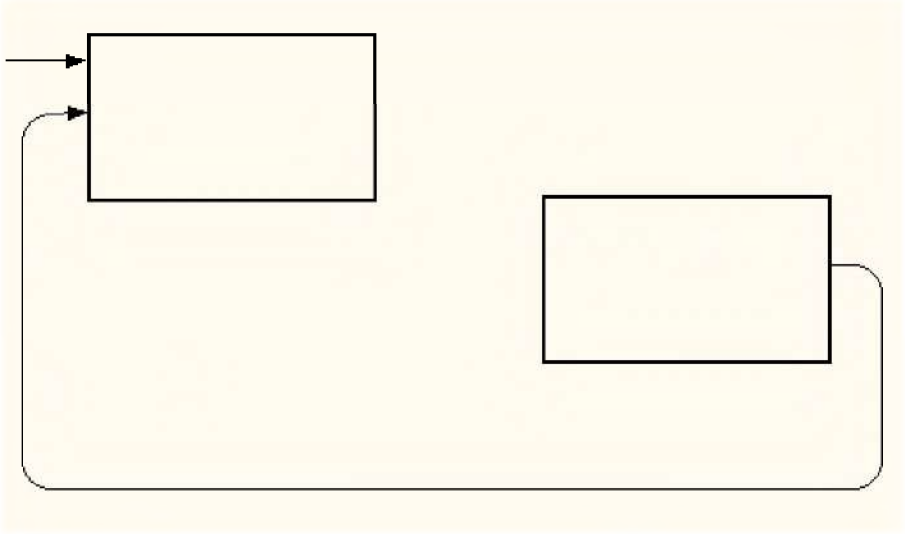


Рис. 4.4. Зворотний зв'язок із входу

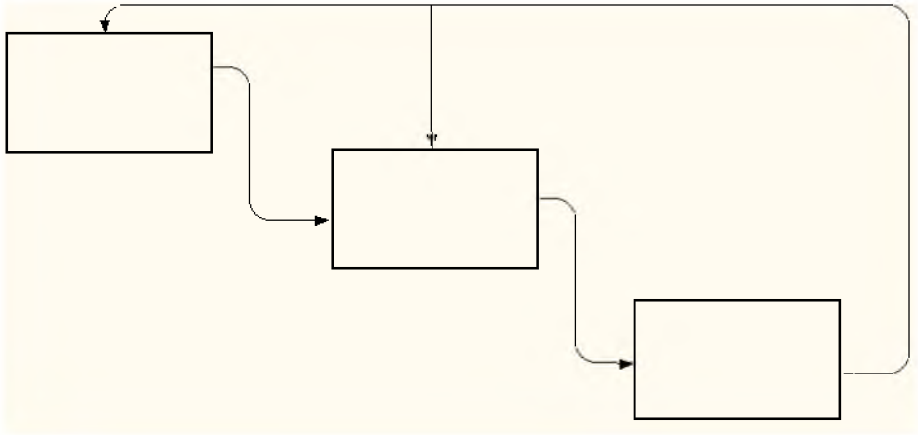


Рис. 4.5. Зворотний зв'язок з управління

Виходи з однієї роботи впливають на майбутнє виконання інших робіт, що згодом вплине на вихідну роботу. Зворотний зв'язок з управління виникає тоді, коли вихід деякого блоку впливає на блок з більшим домінуванням.

Зв'язки «*вихід-механізм*» зустрічаються нечасто (рис. 4.6).

Вони відображають ситуацію, при якій вихід однієї функції стає засобом досягнення мети для іншої.

Зв'язки «*вихід-механізм*» характерні при розподілі джерел ресурсів (наприклад, *необхідні інструменти, обслуговуючий персонал, фізичний простір, устаткування, фінансування, матеріали*).

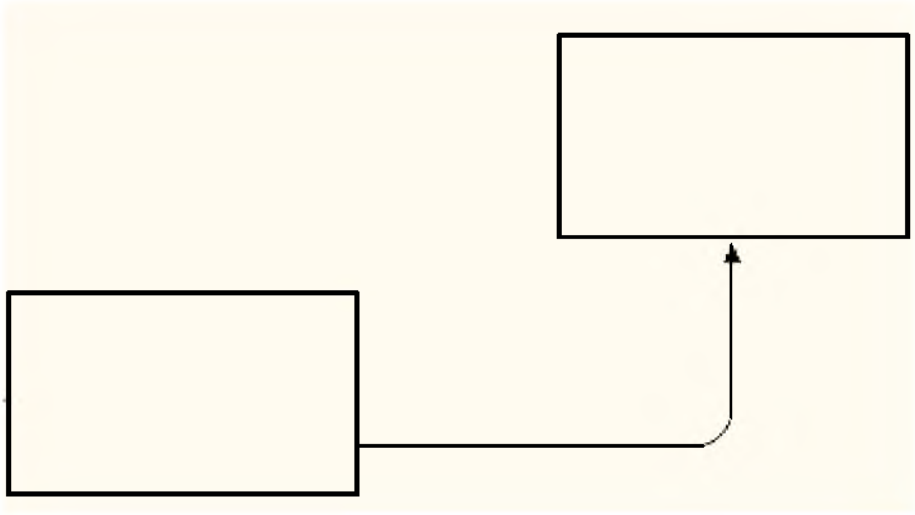


Рис. 4.6. Зв'язок вихід-механізм

В IDEF0 дуга рідко зображує один об'єкт. Зазвичай вона символізує набір об'єктів. Оскільки дуги представляють набори об'єктів, вони можуть мати багато початкових точок (*джерел*) і кінцевих точок (*призначень*).

Тому дуги можуть розгалужуватися і з'єднуватися різними способами. Уся дуга або її частина може виходити з одного або декількох блоків і закінчуватися в одному або декількох блоках.

Методика кількісного аналізу діаграм. Для проведення кількісного аналізу діаграм введемо показники моделі:

- кількість блоків на діаграмі – N;
- рівень декомпозиції діаграми – L;
- збалансованість діаграми – B;
- число стрілок, що з'єднуються з блоком, – A.

Цей набір факторів належить до кожної діаграми моделі. Необхідно прагнути до того, щоб кількість блоків на діаграмах нижніх рівнів була менша кількості блоків на батьківських діаграмах, тобто зі збільшенням рівня декомпозиції *убував би коефіцієнт N/L*. Таким чином, убування цього коефіцієнта говорить про те, що в міру декомпозиції моделі функції повинні спрощуватися, отже, кількість блоків повинна убувати.

Діаграми повинні бути збалансовані. Це означає, що в межах однієї діаграми не повинно спостерігатися ситуації, зображеної на рис. 4.7, *для роботи 1 вхідних стрілок і стрілок управління значно більше, ніж вихідних.*

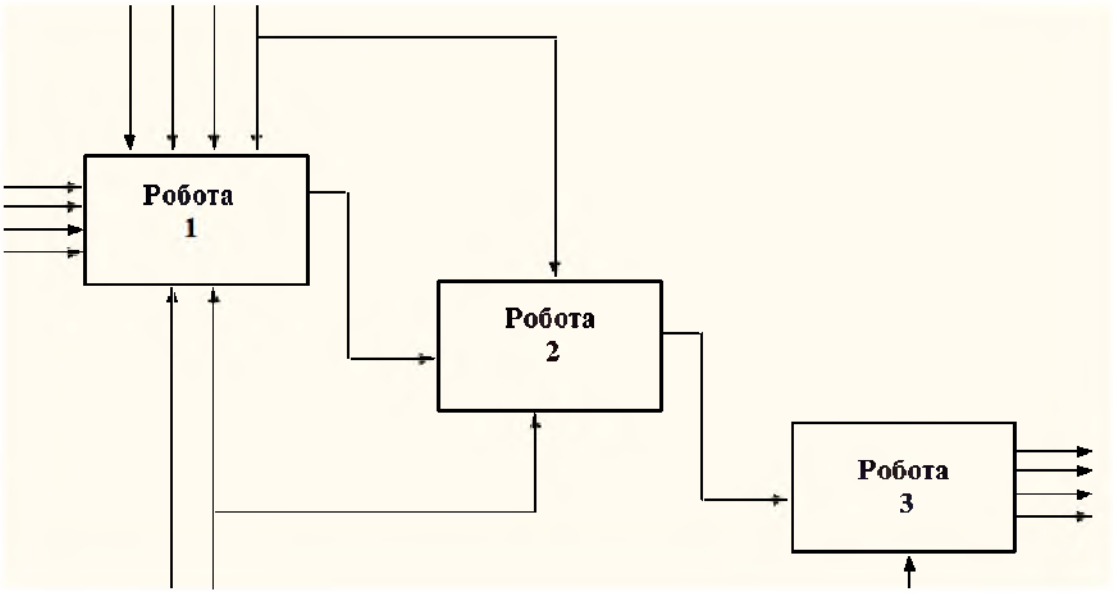


Рис. 4.7. Приклад незбалансованої діаграми

Слід зазначити, що ця рекомендація може не виконуватися в моделях, що описують виробничі процеси.

Наприклад, при описі процедури складання в блок може входити багато стрілок, що описують компоненти виробу, а виходити одна стрілка – готовий виріб.

Введемо коефіцієнт збалансованості діаграми:

$$K_b = \left| \frac{\Sigma_{i=1}^{N} A_i}{N} - \max_{i=1}^{N} \left(A_i \right) \right|$$

Необхідно досягати того, щоб ***Kb*** був мінімальний для діаграми. Крім аналізу графічних елементів діаграми, необхідно розглядати найменування блоків.

Для оцінки імен складається словник елементарних (тривіальних) функцій модельованої системи. Фактично в цей словник повинні потрапити функції нижнього рівня декомпозиції діаграм.

Наприклад, для моделі БД елементарними можуть бути функції «*знайти запис*», «*додати запис у БД*», водночас як функція «*реєстрація користувача*» вимагає подальшого опису.

Після формування словника і складання пакета діаграм системи необхідно розглянути нижній рівень моделі. Якщо на ньому виявляться збіг назв блоків діаграм і слів зі словника, то це говорить, що достатній рівень декомпозиції досягнутий.

Коефіцієнт, що кількісно відображає цей критерій, можна записати як L*C – добуток рівня моделі на число збігу імен блоків зі словами зі словника.

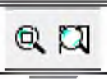
Чим нижчий рівень моделі (більше L), тим кращий збіг.

Інструментарій AllFussion Process Modeller. Під час запуску AllFussion Process Modeller за умовчанням з'являється основна панель інструментів, палітра інструментів і досліджувач моделей Model Explorer.

Функціональність панелі інструментів доступна з основного меню AllFussion Process Modeller (табл. 4.1).

Таблиця 4.1

Елементи управління основної панелі інструментів AllFussion Process Modeller

Елементи управління	Опис	Відповідний пункт меню
	Створити нову модель	File->New
	Відкрити модель	File->Open
	Зберегти модель	File->Save
	Друкувати модель	File->Print
	Вибір масштабу	View->Zoom
	Масштабування	View->Zoom
	Правопис	Tools->Spelling
	Включення та виключення Навігатора моделі Model Explorer	View->Model Explorer
	Включення та виключення Додаткової панелі інструменти праці	Model Manager

При створенні нової моделі виникає діалог, у якому варто вказати опцію моделі (створена заново; відкрита з репозиторію), внести ім'я моделі і обрати методологію, за якою буде побудована модель (рис. 4.8).

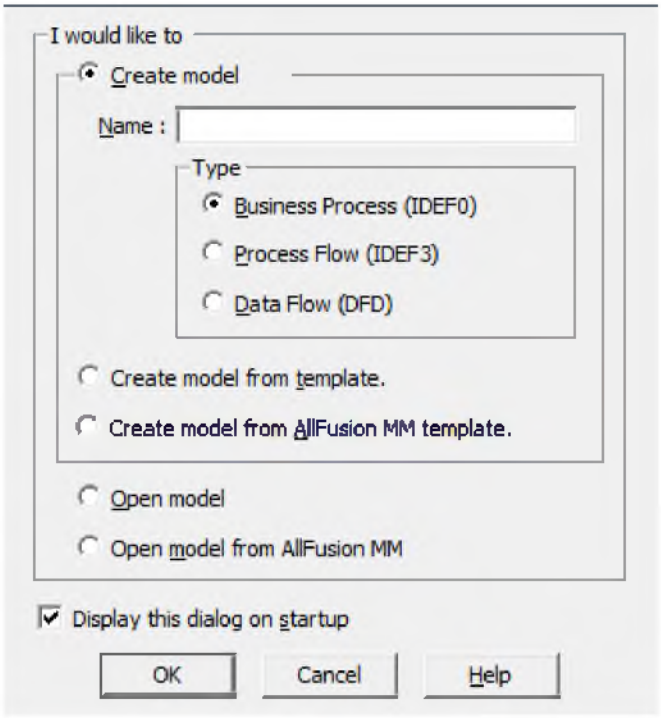


Рис. 4.8. Діалогове вікно створення моделі

CASE-засіб AllFussion Process Modeller підтримує три методології – IDEF0, IDEF3 і DFD. Крім того, можлива побудова змішаних моделей, тобто модель може містити одночасно як діаграми IDEF0, так і IDEF3 і DFD.

Склад палітри інструментів змінюється автоматично, коли відбувається перемикання з однієї нотації на іншу.

Модель в AllFussion Process Modeller розглядається як сукупність робіт, кожна з яких оперує з деяким набором даних. Якщо клацнути по будь-якому об'єкті моделі лівою кнопкою миші, то з'являється спливаюче контекстне меню, кожний пункт якого відповідає редактору властивості об'єкта.

Вибір методології проектування зі створення чи редагування моделі проводиться в діалоговому вікні, що подано на рис. 4.8.

Для побудови діаграм в AllFussion Process Modeller використовується три панелі інструментів для кожного типу діаграм зі своїми можливостями: 1) IDEF0 (додавання роботи на діаграму, створення нового зв'язку, інструмент редагування об'єктів, декомпозиція діаграми); 2) DFD (додавання в діаграму зовнішнього посилання, додавання стрілки потоку даних у діаграму, декомпозиція діаграми, додавання сховища даних, посилання на іншу

сторінку з направленням стрілки на будь-яку діаграму); 3) IDEF3 («старший» зв'язок, перехрестя, об'єкт посилання).

4.2. Технологія побудови моделі за допомогою CASE-засобу AllFussion Process Modeller пакета AllFussion Modeling Suite на прикладі системи «Компетентність студента вишу»

Побудова моделі системи повинна починатися з вивчення всіх документів, що описують її функціональні можливості. Одним з таких документів є технічне завдання, а саме розділи «Призначення розробки», «Мета і завдання системи» і «Функціональні характеристики системи».

Після вивчення вхідних документів та опитування замовників і користувачів системи необхідно сформулювати мету моделювання і визначити точку зору на модель. Розглянемо технологію її побудови на прикладі системи «Компетентність студента вишу», основні можливості якої були описані в Лабораторній роботі 2.

Сформулюємо мету моделювання: описати функціонування системи, яке було б зрозуміло її користувачеві, не вдаючись у подробиці, що пов'язані з реалізацією. Модель будуватимемо з погляду користувачів (*студент, викладач, адміністратор, деканат, фірма*).

Побудова контекстної IDEF0-діаграми. Відповідно до опису системи основною функцією є обслуговування її клієнтів за допомогою обробки запитів, які від них надходять. Таким чином, визначимо єдину роботу контекстної діаграми як «*Обслуговування клієнта системи*».

Далі визначимо вхідні й вихідні дані, а також механізми й управління.

Щоб здійснити обслуговування клієнта, необхідно зареєструвати його в системі, відкрити доступ до БД і обробити його запит.

Як вхідні дані, використовуватимуться: «*ім'я клієнта*», «*пароль клієнта*», «*вхідна БД*», «*запит клієнта*». Виконання запиту веде або до одержання інформації від системи, або до зміни вмісту БД (наприклад, при складанні експертних оцінок), тому вихідними даними будуть «*звіти*» і «*змінена БД*». Процес обробки за-

питів виконуватиметься монітором системи під контролем адміністратора. Таким чином, визначимо контекстну діаграму системи (рис. 4.9).

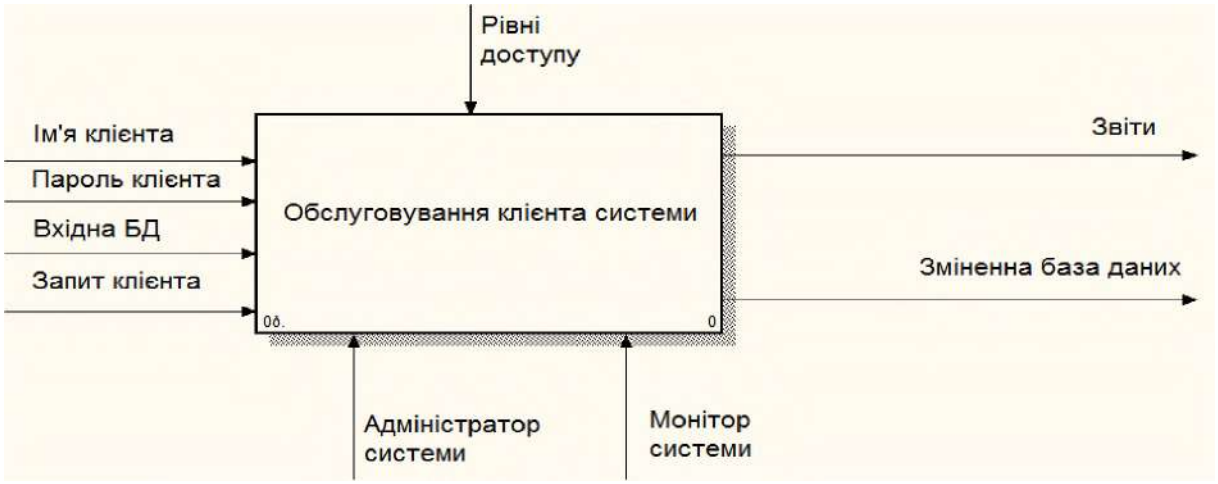


Рис. 4.9. Контекстна діаграма системи

Проведемо декомпозицію контекстної діаграми, описавши послідовність обслуговування клієнта:

1. Визначення рівня доступу в систему.
2. Вибір підсистеми.
3. Звернення до підсистеми.
4. Зміна БД (*за потреби*).
5. Обробка запиту клієнта.

У результаті чого отримаємо діаграму, зображену на рис. 4.10.

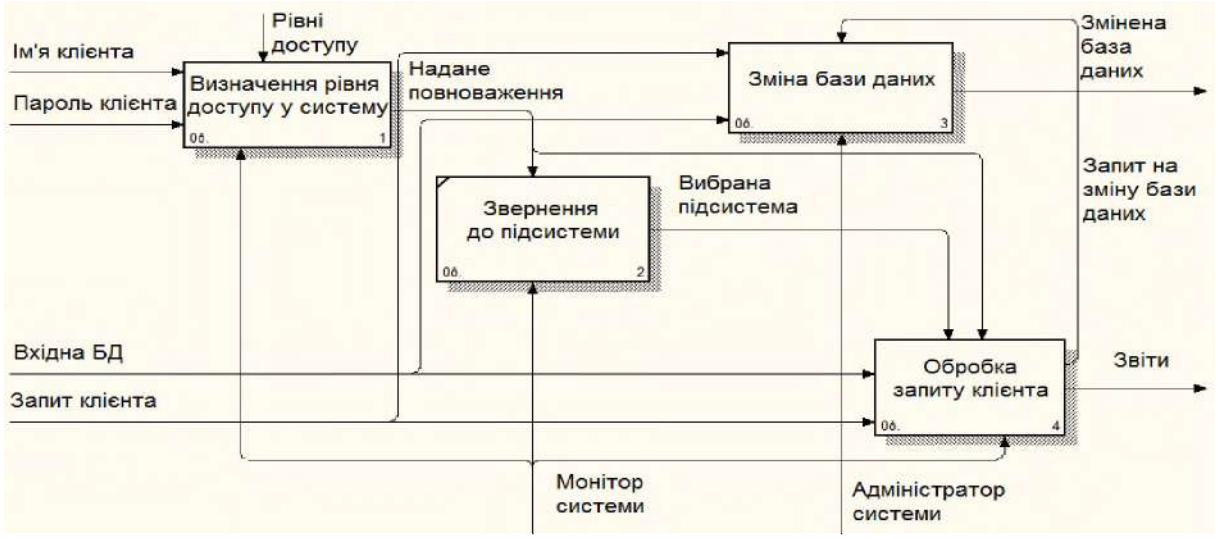


Рис. 4.10. Декомпозиція роботи «Обслуговування клієнта системи»

Закінчивши декомпозицію контекстної діаграми, переходять до декомпозиції діаграми наступного рівня.

Декомпозиція роботи «Визначення рівня доступу в систему». Зауважимо, що при розгляді третього і більш нижніх рівнів моделі вертаються до батьківських діаграм і корегують їх.

Виконаємо декомпозицію послідовно всіх блоків отриманої діаграми, що зображена на рис. 4.10. Першим етапом при визначенні рівня доступу в систему є визначення категорії користувача.

За іменем клієнта здійснюється пошук у базі користувачів, визначаючи його категорію. Відповідно до певної категорії з'ясовуються повноваження, що надані користувачеві системи.

Далі проводиться процедура доступу в систему, перевіряючи ім'я і пароль доступу. Аналізуючи інформацію про повноваження і рівень доступу в систему, для користувача формується набір дозволених дій. Таким чином, визначення рівня доступу в систему буде виглядати, як показано на рис. 4.11.

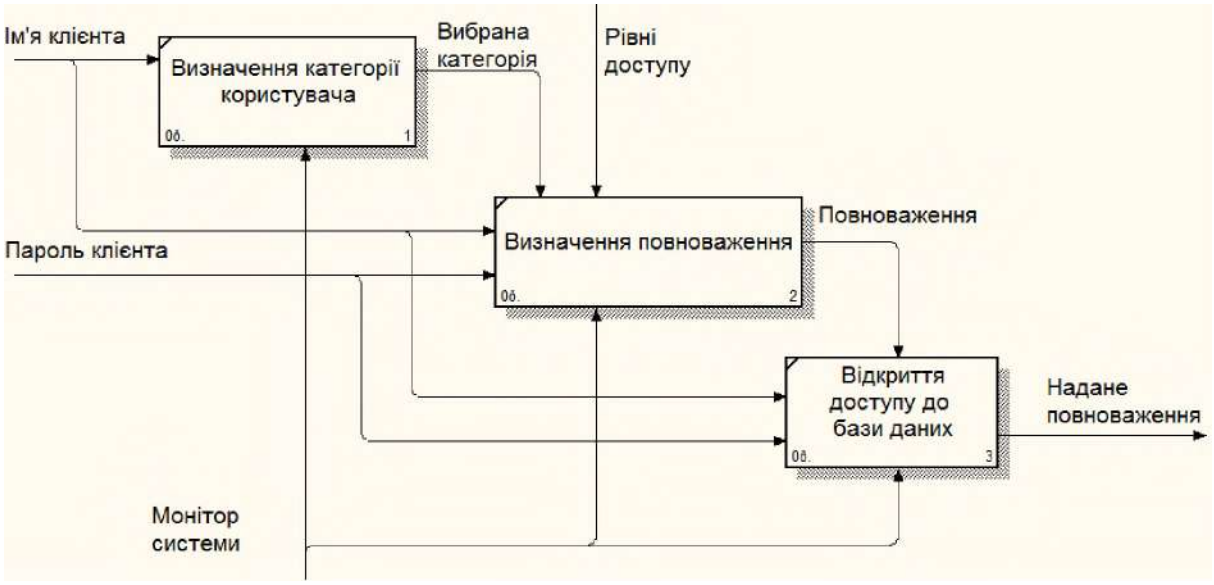


Рис. 4.11. Декомпозиція роботи «*Визначення рівня доступу в систему*»

Декомпозиція роботи «*Звернення до підсистеми*» не відповідає меті і точці зору моделі. Користувача системи не цікавлять внутрішні алгоритми її роботи. У цьому випадку йому важливо, що вибір підсистеми буде зроблений автоматично, без його втручання, тому декомпозиція «*Звернення до підсистеми*» тільки ускладнить модель.

Декомпозиція роботи «Обробка запиту клієнта». Виконаємо декомпозицію роботи «*Обробка запиту клієнта*», що здійснюється підсистемою обробки запитів на основі встановлених категорії і повноваження користувача для дозволу на відкриття БД. У загальному випадку БД може перебувати на віддаленому сервері, тому може знадобитися встановлення з'єднання з ним. Визначимо послідовність робіт: 1) відкриття БД; 2) виконання запиту; 3) генерація звітів. Після відкриття БД необхідно повідомити систему про встановлення з'єднання із БД, потім виконати запит і згенерувати звіти для користувача (рис. 4.12). На етапі виконання запиту може знадобитися зміна вмісту БД, наприклад при складанні експертних оцінок. Тому на діаграмі (рис. 4.12) необхідно передбачити таку можливість.

З аналізу отриманої діаграми виникає запитання, за якими правилами відбувається генерація звітів? Необхідна наявність заздалегідь сформованих шаблонів, за якими проводитиметься вибірка із БД, причому ці шаблони повинні відповідати запитам і повинні бути заздалегідь визначені. Крім того, клієнтові повинна бути надана можливість вибору форми звіту.

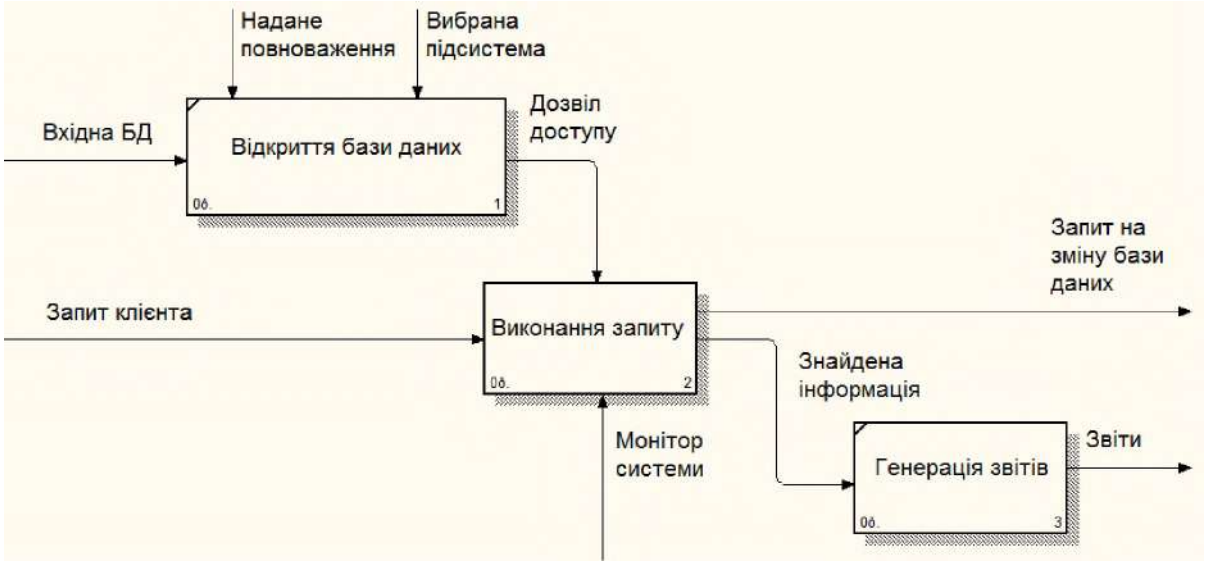


Рис. 4.12. Декомпозиція роботи «Обробка запиту клієнта»

Скорегуємо діаграму (рис. 4.12), додавши в неї стрілки «*Шаблони звітів*» і тунельну стрілку «*Клієнт системи*». На рис. 4.13 подана скорегована діаграма декомпозиції роботи «Обробка запиту клієнта» (*варіант 2*).

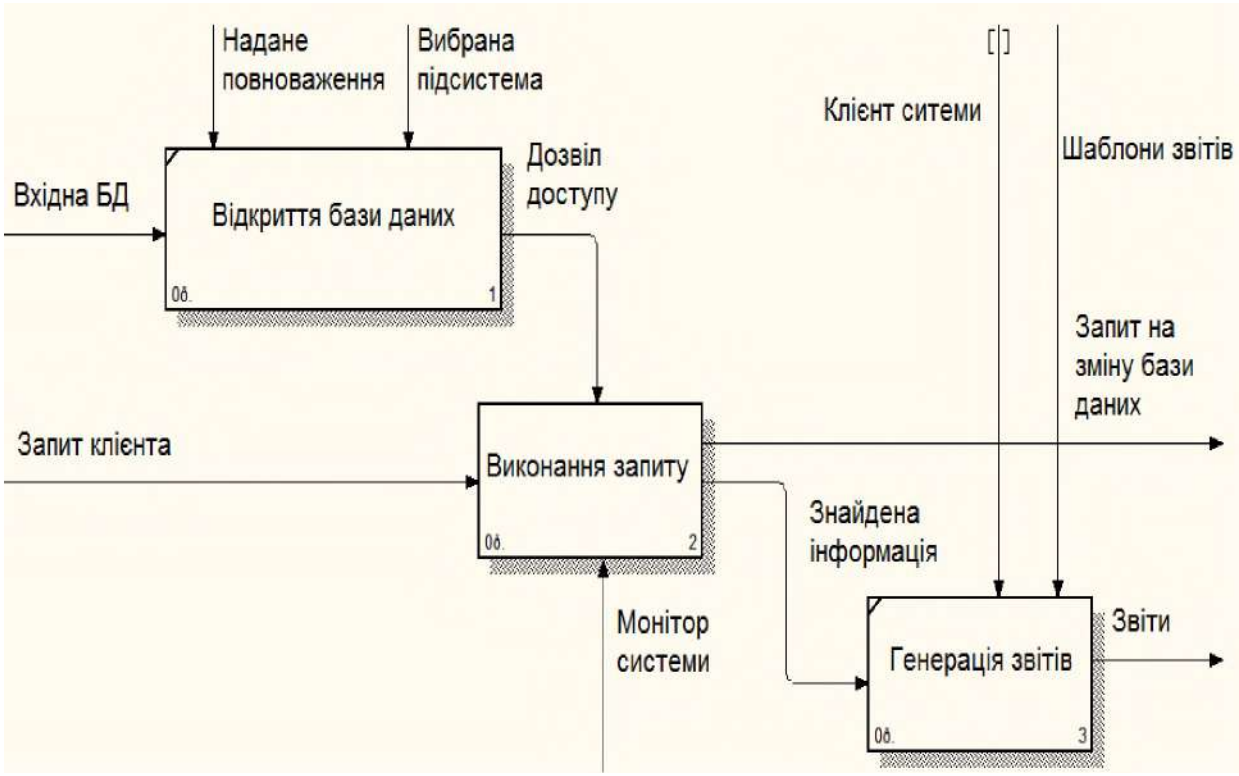


Рис. 4.13. Декомпозиція роботи «Обробка запиту клієнта» (*варіант 2*)

Тут тунелювання «*Клієнт системи*» застосоване для того, щоб не виносити стрілку на діаграму верхнього рівня, тому що функція вибору форми звіту не є досить важливою для відображення її на батьківській діаграмі. Внесені зміни до діаграми «*Обробка запиту клієнта*» приводять до корегування всіх батьківських діаграм, поданих на рисунках 4.9–4.10. У результаті отримаємо кореговані діаграми, які представлені на рисунках 4.14–4.15.

Декомпозицію роботи «*Виконання запиту*» (рис. 4.13) доцільно провести за допомогою діаграми DFD, що краще відображає процеси обробки інформації.

Декомпозиція роботи «Зміна БД». Виконаємо декомпозицію блоку 3 «*Зміна БД*» (рис. 4.10), після чого він матиме такий вигляд (рис. 4.16).

Дані системи «Компетентність студента вишу» містяться в БД, яка складається із семи таблиць: користувач; студент; вакансія; успішність; тест; експертна оцінка; резюме.

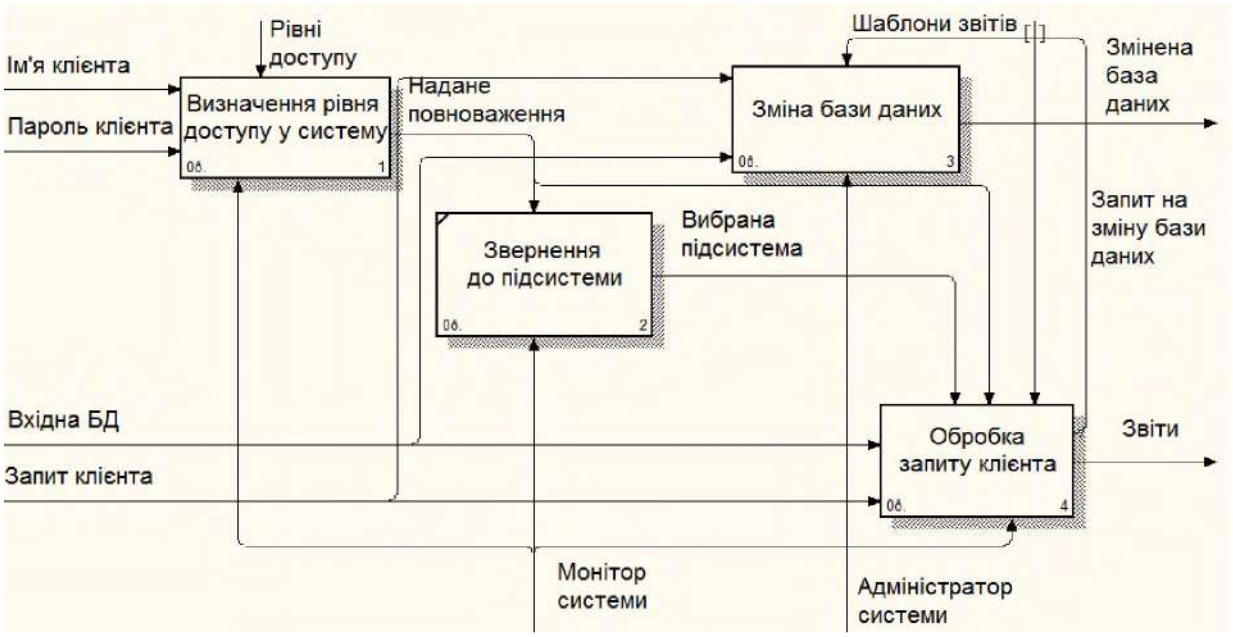


Рис. 4.14. Декомпозиція роботи «Обслуговування клієнта системи» ***(в-2)***

Відповідно до мети моделювання клієнтові важливо розуміти, що дані, які надійшли, не одразу обновляються в системі, а проходять додатковий етап обробки і контролю.

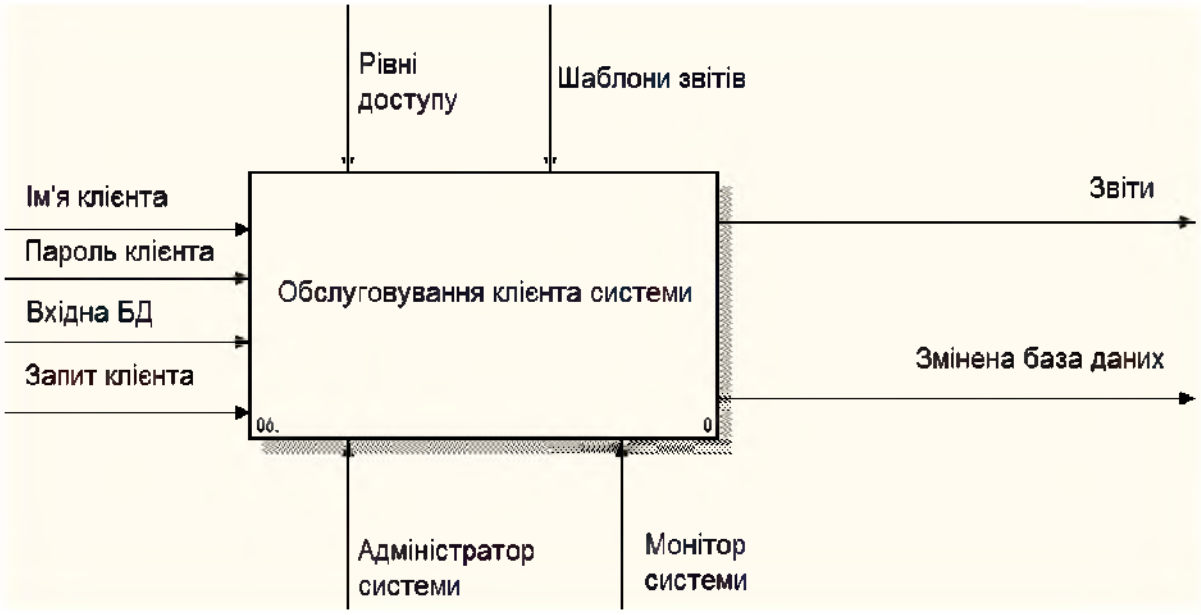


Рис. 4.15. Контекстна діаграма системи ***(варіант 2)***

Алгоритм модифікації БД можна сформулювати в такій послідовності: 1) визначається таблиця БД, у якій змінюватимуться дані; 2) оператором формується тимчасовий набір даних і надається адміністраторові; 3) адміністратор здійснює контроль даних і вносить їх у таблиці БД.

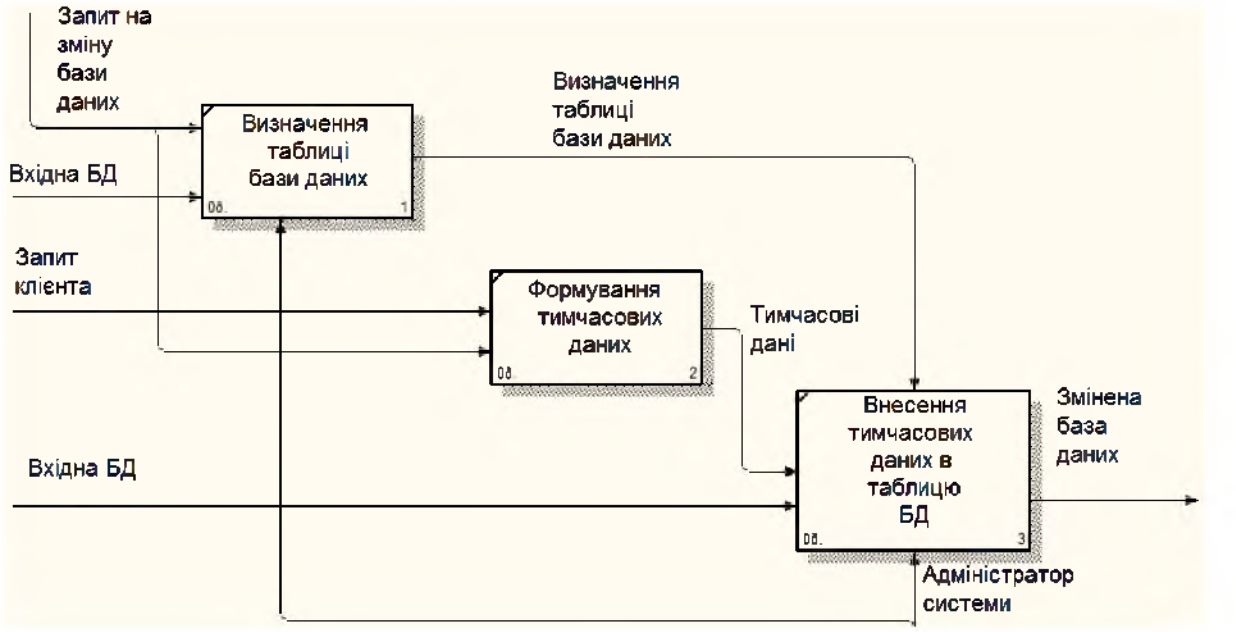


Рис. 4.16. Декомпозиція роботи «Зміна БД»

Розглянутий алгоритм модифікації БД можна реалізувати іншим способом, надавши можливість оновлення БД безпосередньо після запитів, минаючи процес контролю даних. У цьому випадку необхідно забезпечити контроль цілісності БД для запобігання її ушкодження. Діаграма «*Зміна БД*» б виглядатиме таким чином (рис. 4.17).

Проведення подальшої декомпозиції діаграми «*Зміна БД*» поясненням, як здійснюється зміна фізичного рівня БД у системі, ускладнюватиме модель. При цьому користувач не одержить ніякої додаткової інформації про роботу системи «Компетентність студента вишу». Декомпозицію цієї роботи доцільно проводити в процесі проектування БД системи на етапі створення логічної моделі БД.

Кількісний аналіз «Обробка запиту клієнта» (первинної та вдосконаленої). Проведемо кількісний аналіз первинної та вдосконаленої моделей роботи «Обробка запиту клієнта», зображених на рис. 4.12 і 4.13, відповідно до вищеописаної методики. Розглянемо коефіцієнт *N/L* цих моделей. У батьківської діаграми «*Обробка запиту клієнта*» (рис. 4.10) коефіцієнт N/L дорівнює 4/2 = 2, а діаграми декомпозиції коефіцієнт N/L становить 3/3 = 1. Значення коефіцієнта убуває, що говорить про спрощення опису функцій зі зниженням рівня моделі.

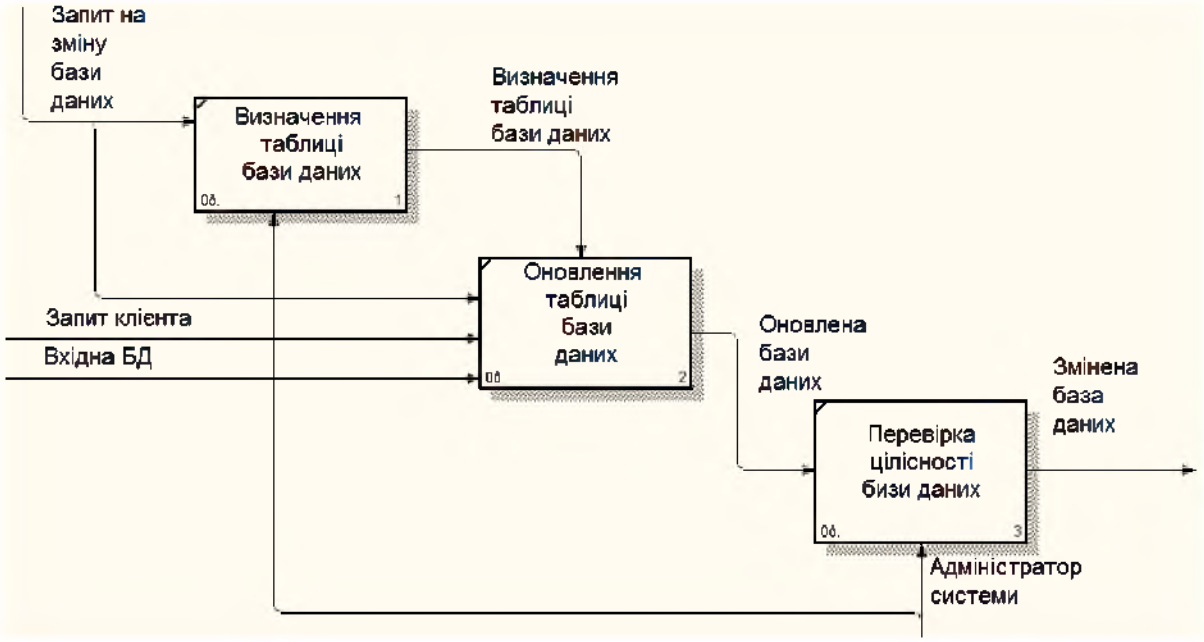


Рис. 4.17. Декомпозиція роботи «Зміна БД» ***(варіант 2)***

Розглянемо зміну коефіцієнта **Kb** в обох варіантах моделей.

Для першого варіанта, зображеного на рис. 4.12, ***Kb = |11/3 - 5| = 1,34,*** для другого варіанта (рис. 4.13) ***Kb = |13/3 - 5| = 0,66.***

Коефіцієнт ***Kb*** зменшує своє значення, отже, збалансованість діаграми покращується. Вважатимемо, що рівень декомпозиції розглянутих діаграм достатній для реалізації мети моделювання і на діаграмах нижнього рівня як найменування робіт використовуються елементарні функції (*з погляду користувача системи*).

Підводячи підсумки розглянутого прикладу, необхідно відзазначити важливість розгляду декількох варіантів діаграм при моделюванні системи. Такі варіанти можуть виникати при коригуванні діаграм, як це було зроблено з «*Обробкою запиту клієнта*» або при створенні альтернативних реалізацій функцій системи (*декомпозиція роботи «Зміна БД»*).

Розгляд альтернативних варіантів дозволяє обрати найкращий і включити його в пакет діаграм для подальшого розгляду.

Контрольні запитання

1. Поясніть принцип іменування розгалужуючих і зливаючих стрілок.

2. Які методології підтримуються CASE-засобом AllFussion PM?

3. Перелічіть основні елементи головного вікна AllFussion PM.

4. Опишіть процес створення нової моделі в AllFussion PM.

5. Як провести зв'язок між роботами?

6. Як задати ім'я роботи?

7. Опишіть процес декомпозиції роботи.

8. Як додати роботу на діаграму?

9. Як дозволити тунелювання стрілки?

10. Чи може модель AllFussion PM містити діаграми декількох методологій?

4.3. Лабораторна робота 5

Тема. Методологія IDEF0 зі створення функціональної моделі бізнес-процесів індивідуального об'єкта дослідження та розвиток «Індивідуального проекту»

Мета роботи:

✓провести аналіз діяльності організації (фірми, відомства), що відповідає індивідуальному об'єкту дослідження з формуванням семантики процесів;

✓закріпити навики створення і редагування функціональних моделей в AllFusion PM;

✓розвинути «Наскрізний індивідуальний проект» у результаті створення функціональної моделі IDEF0 індивідуального об'єкта дослідження за допомогою AllFusion PM.

Завдання

1. Виділити основні процеси в діяльності організації (фірми, відомства), що відповідає індивідуальному об'єкту дослідження.

2. Описати фактори (стрілки), які забезпечують функціонування організації (фірми, відомства), – вхід, вихід, управління, механізми.

3. Створити контекстну діаграму на основі семантики процесів та стрілок для представлення діяльності організації (фірми, відомства), *що відповідає індивідуальному об'єкту дослідження.*

4. Сформувати визначення, статус і джерело для кожної роботи (процесу).

5. Описати стрілки, які забезпечують виконання кожної роботи (процесу).

6. Створити діаграму декомпозиції на основі обраного числа та опису процесів і стрілок для представлення діяльності організації (фірми, відомства), *що відповідає індивідуальному об'єкту дослідження.*

7. Детально описати операції функціонування кожної роботи (процесу).

8. Описати стрілки, які забезпечують виконання операцій кожної роботи (процесу).

9. Створити діаграми декомпозиції наступного рівня на основі більш детального опису представницького процесу та стрілок для представлення діяльності організації (фірми, відомства), *що відповідає індивідуальному об'єкту дослідження.*

Хід роботи

1. Виділити основні процеси в діяльності організації (фірми, відомства).

2. Описати фактори (стрілки), які забезпечують функціонування організації (фірми, відомства), – вхід, вихід, управління, механізми, – відповідно до табл. 4.2.

Таблиця 4.2

Стрілки контекстної діаграми

Назва стрілки	Визначення стрілки	Тип стрілки

3. Створити контекстну діаграму на основі семантики процесів та стрілок для представлення діяльності організації (фірми, відомства).

4. Встановити визначення, статус і джерела для *кожної роботи* (процесу) відповідно до табл. 4.3.

Таблиця 4.3

Роботи діаграми декомпозиції А0

Назва роботи	Визначення

5. Описати стрілки, які забезпечують виконання *кожної роботи* (процесу), відповідно до табл.4.4.

Таблиця 4.4

Стрілки діаграми декомпозиції А0

Назва стрілки	Джерело стрілки	Тип джерела стрілки	Призначення стрілки	Тип призначення стрілки

6. Створити діаграму декомпозиції **А0** на основі вибраного числа та опису процесів і стрілок.

7. Детально *описати операції* функціонування кожної роботи (процесу) відповідно до табл. 4.5.

Таблиця 4.5

Роботи діаграми декомпозиції А2

Назва роботи	Визначення

8. Описати стрілки, які забезпечують *виконання операцій кожної роботи* (процесу), відповідно до табл. 4.6.

Таблиця 4.6

Стрілки діаграми декомпозиції А2

Назва стрілки	Джерело стрілки	Тип джерела стрілки	Призначення стрілки	Тип призначення стрілки

9. Створити діаграму декомпозиції наступного рівня **А2** на основі більш детального опису представницького процесу та стрілок.

Оформлення звіту

Звіт про виконану роботу оформляється згідно із «Загальними рекомендаціями до виконання лабораторних робіт та оформлення звіту» (додаток А, А.1, А.2).

Завдання для моделювання

1. Формування семантики процесів за аналізом діяльності організації, що відповідає об'єкту дослідження, БД якого дозволить отримати відомість.

2. Створення ФМ на основі семантики процесів діяльності організації:

ВІДОМІСТЬ 1

розрахунку відрядної індивідуальної заробітної плати

за цехом _____ за _____ місяць 201__ р.

код та назва цеху

Табномер	Прізвище	Код деталі	Розряд	Професія		Кількість виготовлених деталей	Розцінка за деталь	Відроблений час	Сума заробітної плати
				Код	Назва				

ВІДОМІСТЬ 2

потреби в матеріалах на річну виробничу програму

за підприємством на 201__ р.

Код матеріалу	Назва матеріалу	Одиниці вимірювання	Код виробу	Назва виробу	Програма випуску виробів на рік	Норми витрат матеріалу на виріб	Потреба матеріалу на річну програму

ВІДОМІСТЬ 3

обліку руху матеріалів на складі ___ за ___ місяць

код та назва складу

Матеріал			Залишок вхідний		Надходження		Видатки		Залишок вихідний	
Код	Назва	Одиниця виміру	Кількість	Вартість	Кількість	Вартість	Кількість	Вартість	Кількість	Вартість

ВІДОМІСТЬ 4

обліку якості готової продукції за цехом ___ за __ місяць

код та назва цеху

Номер дільниці	Деталь		Операція		Вид браку		Причина (винуватець)		Кількість деталей	
	Код	Назва	Код	Назва	Код	Назва	Код	Назва	Придатних	Бракованих

ВІДОМІСТЬ 5

відвантаження готової продукції за _____ місяць

Код покупця	Назва покупця	Код продукції	Назва продукції	№ угоди на відвантаження	Кількість за угодою	Фактично відвантажено	Відхилення

4.4. Теоретичні аспекти методології діаграм потоків даних (DFD)

DFD-методологія розроблена незалежно від IDEF0 і супутніх їй методологій. Автори цієї методики – Крис Гейн і Триш Сарсон (Chris Gane and Trish Sarson), США. В її основу, так само, як і в IDEF0, належать функції, що представлені блоками-прямокутниками і стрілками-зв'язками.

Склад компонентів, що формують DFD-моделі, такі: Activities – *процеси, функції*, Arrows – *зв'язки*, External Reference (ER) – *зовнішні посилання (сутності)*, Data Store (DS) – *накопичувачі (сховища) даних.*

DFD-моделі у вигляді діаграм потоків даних представляють мережу пов'язаних між собою робіт. Їх зручно використовувати для опису документообігу й обробки інформації в інформаційних системах. Таким чином, діаграма потоків даних DFD може описувати:

✓ функції обробки інформації (роботи, Activities);

✓ потоки даних (стрілки, arrow), документи об'єкти, співробітники або відділи, які беруть участь в обробці інформації;

✓ зовнішні посилання (external reference), які забезпечують інтерфейс із зовнішніми об'єктами, що перебувають за межами модельованої системи;

✓ таблиці для зберігання інформації, документів (сховища даних, data store).

Компоненти діаграм потоків даних. Для побудови діаграм DFD в CASE-засобі AllFussion PM використовуються, згідно з нотацією Гейна-Сарсона, такі компоненти: потоки даних, процеси, сховище, зовнішня сутність.

Потоки даних є механізмами, що використовуються для моделювання передачі інформації (або фізичних компонентів) з однієї частини системи в іншу. Потоки зображуються на діаграмі іменованими стрілками, орієнтація яких указує напрям руху інформації. Стрілки можуть підходити до будь-якої грані прямокутника роботи (процесу, функції) і можуть бути двонаправленими для опису взаємодії типу «команда–відповідь».

Призначення процесу (роботи, функції) полягає в продукуванні вихідних потоків із вхідних відповідно до дії, що задається ім'ям процесу. Кожний процес повинен мати унікальний номер для посилань на нього усередині діаграми. Цей номер може використовуватися разом з номером діаграми для одержання унікального індексу процесу у всій моделі.

Сховище даних дозволяє на певних етапах визначати дані, які зберігатимуться в пам'яті між процесами. Фактично сховище представляє «зрізи» потоків даних у часі. Інформація, що воно містить, може використовуватися в будь-який час після її визначення, при цьому дані можуть обиратися в будь-якому порядку. Ім'я сховища повинно ідентифікувати його вміст. У випадку, коли потік даних входить у сховище або виходить із нього і його структура відповідає структурі сховища, він повинен мати те саме ім'я, яке немає необхідності зображати на діаграмі.

Зовнішня сутність представляє сутність поза контекстом системи, що є джерелом або приймачем даних системи. Передбачається, що об'єкти, представлені такими вузлами, не повинні брати участь ні в якій обробці. Блоки зовнішніх сутностей зображуються у вигляді прямокутника з тінню (*справа і знизу*) і розташовуються по краях діаграми. Одна зовнішня сутність може бути використана багаторазово на одній або декількох діаграмах (рис. 4.18).

Для доповнення моделі IDEF0 діаграмою DFD потрібно в процесі декомпозиції у вікні діалогу Activity Box Count указати тип діаграми DFD (рис. 4.19).

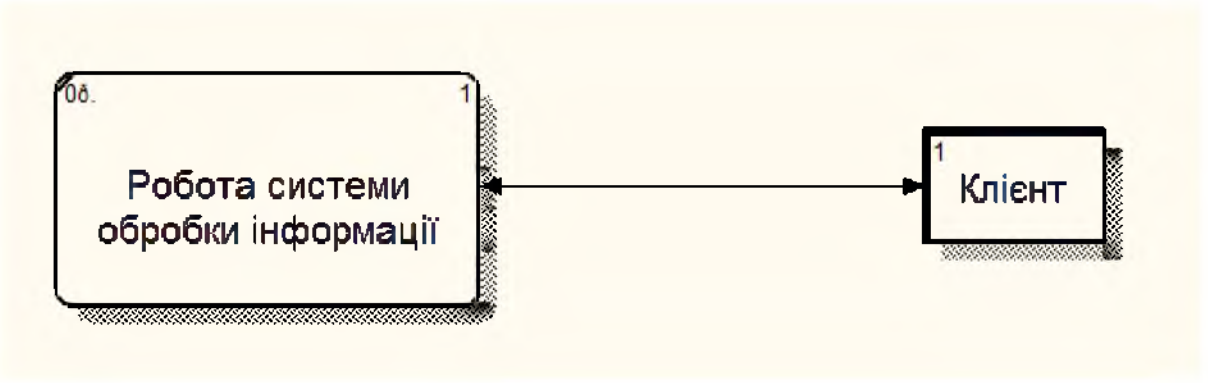


Рис. 4.18. Зовнішня сутність

На відміну від методології IDEF0, яка є основою для моделювання матеріально-інформаційних систем, DFD-методологія підтримує моделювання виключно систем (підсистем) інформаційних.



Рис. 4.19. Вікно діалогу Activity Box Count для вибору нотації DFD

Порівняльна характеристика DFD і IDEF0 методологій. В узагальненому викладі особливості DFD-методики, порівняно з IDEF0, полягають у такому:

1. Синтаксис DFD не підтримує предметної спеціалізації стрілок на 4 групи, підтримуючи лише стрілки (*і відповідно зв'язки*) входу і виходу.

2. Відсутні будь-які обмеження на топологію зв'язків між компонентами діаграми: Activities і граничних зв'язків.

3. Як зовнішні посилання (external reference, ER), так і сховища даних (data store, DS) можуть бути уведені на TOP (верхню)-діаграму.

4. На кожній з DFD-діаграмі одноіменні ER і DS можуть бути використані неодноразово.

5. Не регламентується прив'язка стрілок входів і виходів до тих або інших граней блоку.

6. Виходячи з умови коректності опису блоку Activity зв'язками діаграма DFD визнає наявність одного входу й одного виходу.

7. Не відображається домінування одних блоків Activity над іншими.

8. Зображення DFD-блоків відрізняється тим, що їхні кути мають округлення.

9. Методологія DFD передбачає можливість вказівки механізмів реалізації функцій у нижньому полі блоку, розділеного по вертикалі на два інформаційні поля. Проте засіб AllFusion PM не підтримує цієї можливості.

DFD-діаграма функціонування довільної ІС одержує на вході інформацію, що перетвориться у вихідну інформацію, що згенерована системою за запитом. Очевидно, що вхід «*запит*» відіграє роль управління роботою ІС. Відповідно до визначення IDEF0 *управління* виражає «*умови, що необхідні для формування коректного виходу*». Ту ж роль відіграє в ІС *запит*. Так, якщо на вхід ІС будуть подані цифри 2 і 3, то на виході одержимо: 5 – при запиті «*скласти*» і 6 – при запиті «*помножити*».

Таким чином, на відміну від думки про відсутність у DFD-моделях *механізмів* і *управління*, насправді, як і IDEF0, DFD структурується фактично теж на 5 груп операторів: *функцію, вхід, вихід, управління-запит, механізм*. Особливості полягають лише в способі подання *управління* і *механізмів* на DFD-діаграмах.

Варто додати, що механізми ІС мають як інформаційну, так і матеріальну природу, а в більшості випадків ще і «людську» (Human) для інтерактивних систем. Їхні матеріальні механізми представлені засобами обчислювальної техніки, що підтримують функціонування ІС.

Управління ІС мають ту ж природу, що і всі DFD-зв'язки, винятково інформаційну. Вони можуть бути представлені стрілками входу.

DFD-діаграма може бути декомпозицією IDEF0. Як і в IDEF0, може бути поведена декомпозиція блоку Activity. При цьому зв'язок між батьківськими і дочірніми діаграмами DFD-моделі підтримується завдяки міграції граничних стрілок TOP-діаграми і всіх стрілок декомпонованої Activity батьківської діаграми другого і більш низького ієрархічного рівня.

Крім декомпозиції DFD Activity, без зміни методики моделювання, так само як і в IDEF0, можлива декомпозиція з переходом в IDEF3-представлення. Однак декомпозиція DFD в IDEF0 некоректна і засобом AllFusion Process Modeler блокується.

4.5. Технологія побудови моделі на основі DFD за допомогою AllFussion Process Modeller на прикладі системи «Компетентність студента вишу»

Використання діаграм DFD. Діаграми DFD можна використовувати як доповнення до діаграм IDEF0 для опису документообігу й обробки інформації.

Розглянемо процес «*Обробка запиту клієнта*», що описаний у п. 4.2 на рисунку 4.13 (третій рівень декомпозиції). Запити до ІС «Компетентність студента вишу» надходять від користувачів, тому запити від кожної категорії мають певні особливості і оброблятимуться окремо.

Виділимо зовнішні сутності діаграми відповідно до кожної категорії користувачів, визначаючи потоки даних, якими вони обмінюються із ІС. Одержимо діаграму, зображену на рис. 4.20.

Відповідно до опису системи проведемо декомпозицію поданих на рис. 4.20 блоків. Усі процеси обробки запитів контролюються і виконуються монітором системи, тому стрілка-механізм «*Монітор системи*» буде повторюватися на декомпозованих діаграмах. Точка зору моделі, що визначена в лабораторній роботі з вивчення IDEF0-методології, не вимагає розгляду внутрішніх особливостей функціонування системи, тому затунелюємо стрілку «*Монітор системи*» для того, щоб не переносити її на діаграми нижнього рівня. Декомпозиція запиту до БД подана на рисунках 4.21–4.24.

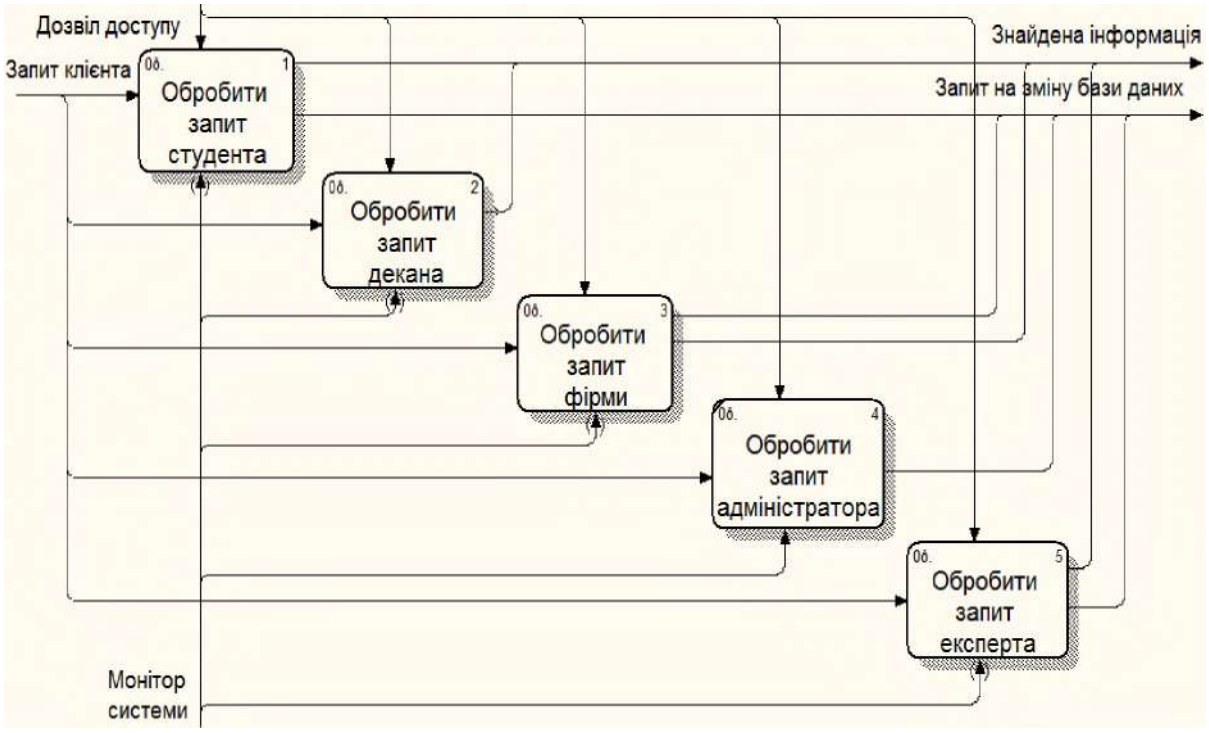


Рис. 4.20. DFD-декомпозиція роботи «Виконання запиту»

Аналіз створених діаграм DFD. Проведемо аналіз отриманих діаграм. Розглядаючи діаграму, зображену на рис. 4.20, необхідно відзначити наявність у ній зайвого блоку «*Обробити запит адміністратора*».



Рис. 4.21. Декомпозиція процесу «Обробити запит студента»

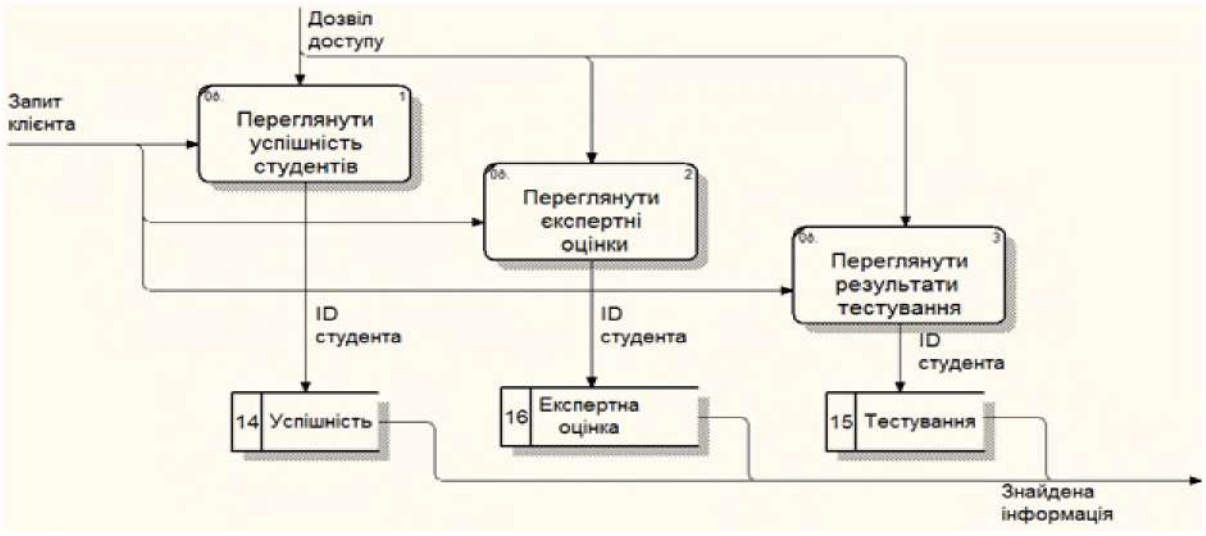


Рис. 4.22. Декомпозиція процесу «Обробити запит декана»



Рис. 4.23. Декомпозиція процесу «Обробити запит фірми»



Рис. 4.24. Декомпозиція процесу «Обробити запит експерта»

Перелік «*процесів*» при складанні першого варіанта діаграми був визначений виходячи з категорій користувачів. Це призвело до виникнення конфлікту з функціями системи і з точкою зору на модель. Адміністратор не обслуговується системою як звичайний клієнт, він забезпечує її моніторинг. Адміністратор може додавати паролі, змінювати рівень доступу в систему, додавати нового користувача тощо.

З погляду клієнта системи, діяльність адміністратора є другорядною, тому на всіх діаграмах відсутній опис функцій адміністратора, представляючи його вплив як «*механізм*» для інших функцій. Тому можна видалити *процес* «*Обробити запит адміністратора*» й отримаємо DFD-декомпозицію *процесу* «*Виконання запиту*» (*варіант 2*), що показаний на рис. 4.25.

Перейдемо до аналізу діаграми декомпозиції *процесу* «*Обробити запит студента*». Згідно з рис. 4.14, що описує процес обробки запиту, після виконання запиту відбувається генерація звіту за обраною користувачем формою і лише потім здійснюється перегляд отриманих даних.

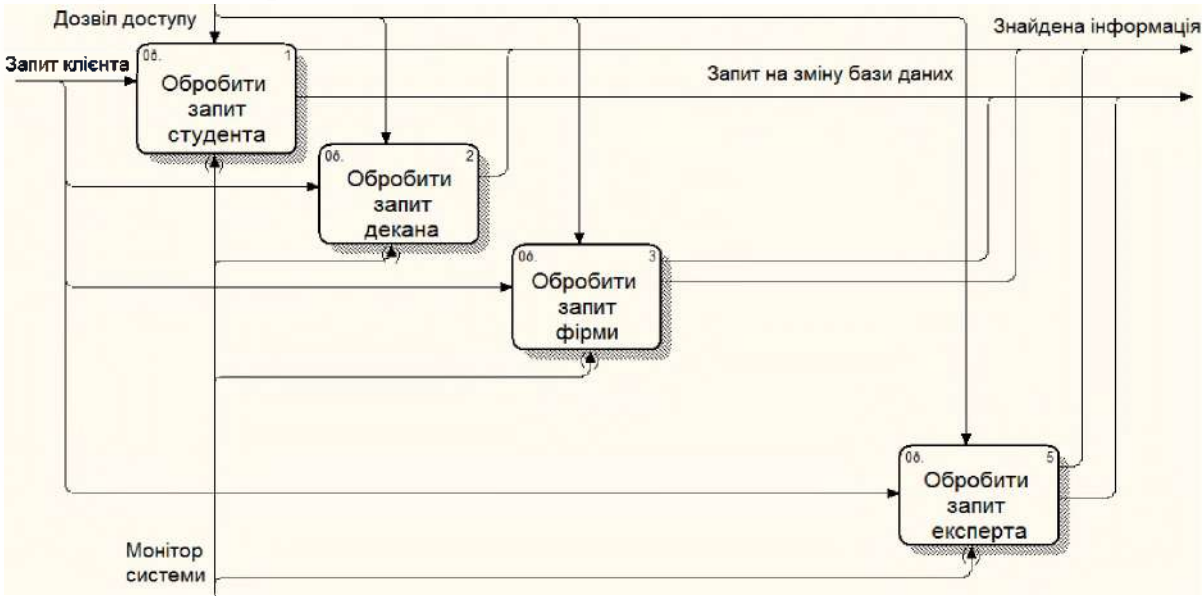


Рис. 4.25. DFD-декомпозиція *процесу* «Виконання запиту» (*варіант 2*)

На діаграмі «*Обробити запит студента*» (рис. 4.21) процес починається зі слова «*переглянути*». Виникає протиріччя. У процесі обробки запиту студента дані повинні бути знайдені в таблицях БД і передані іншому модулю, що генерує звіти.

Змінимо назву *процесу* на діаграмі й одержимо другий варіант діаграми (рис. 4.26).

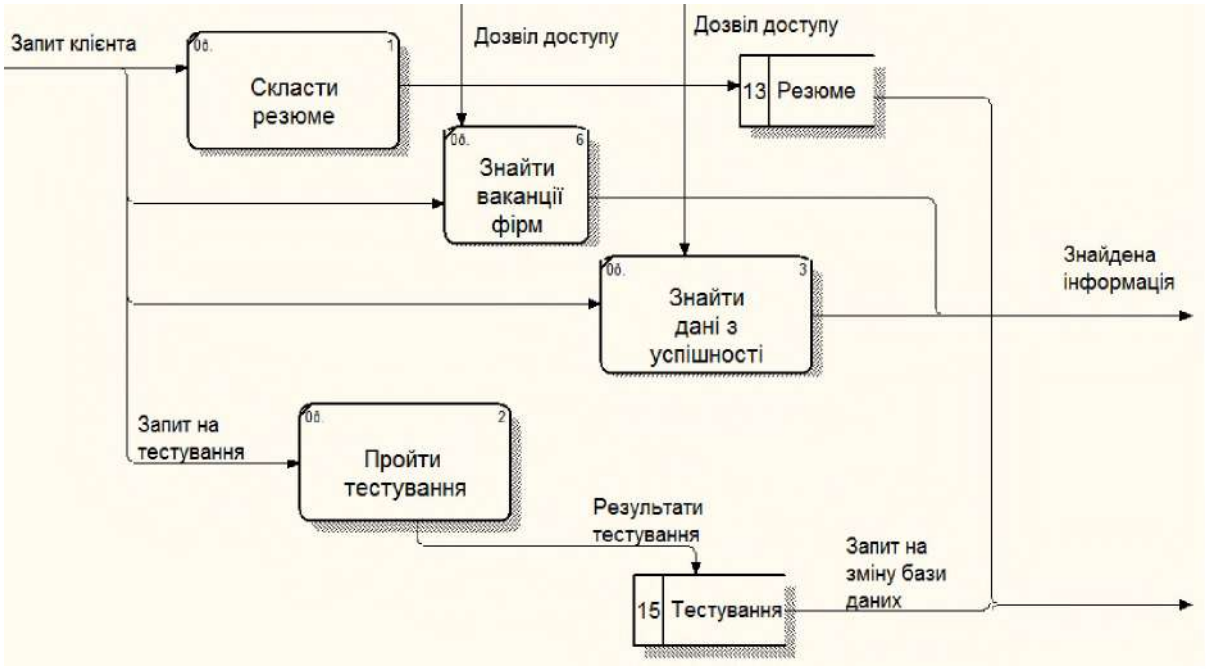


Рис. 4.26. Декомпозиція *процесу* «Обробити запит студента» (варіант 2)

Аналогічне протиріччя спостерігається й у діаграмі на рис. 4.22. Крім того, зі сховищ, що зображують таблиці БД, виходить стрілка «*Знайдена інформація*», у той час як ця стрілка повинна виходити прямо з *процесу*, передаючи дані для генерації звіту. Звертання до таблиць БД потрібно у випадку, коли інформація необхідна для виконання функції усередині *процесу*. Скорегуємо діаграму, одержавши новий варіант (рис. 4.27).

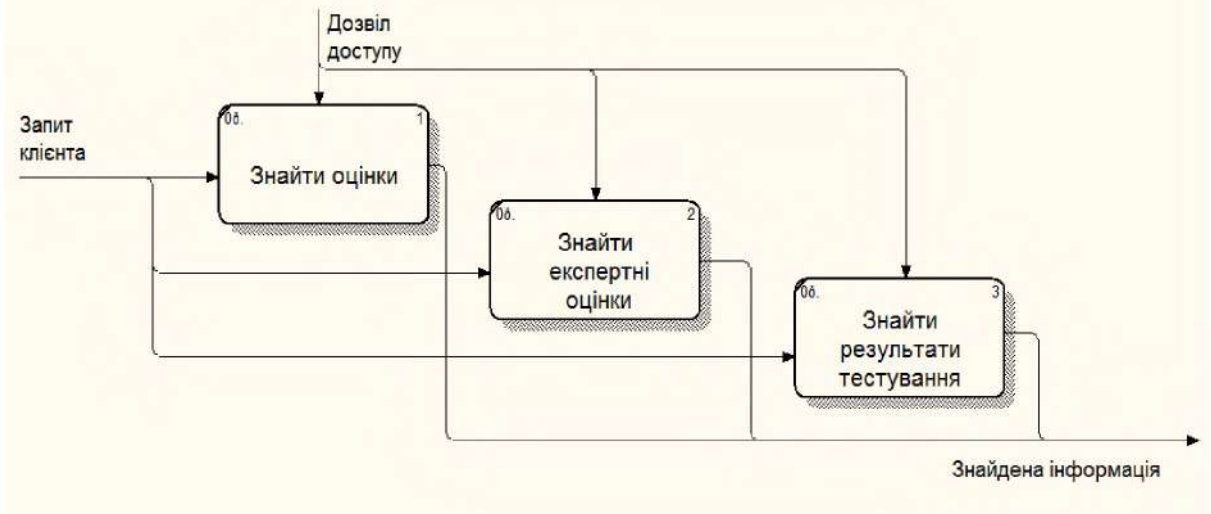


Рис. 4.27. Декомпозиція *процесу* «Обробити запит декана» (варіант 2)

Перейдемо до наступної діаграми – «*Обробити запит експерта*» (рис. 4.24). Корегувань ця діаграма не вимагає, крім уточнення назви *процесів*.

Перед проведенням експертної оцінки експерт повинен *обрати студента.* Термін «*визначити*» може мати на увазі рішення експерта, при якому він на підставі будь-яких переваг обирає студента для проведення експертної оцінки. Насправді студент може просто обиратись із загального списку. Причому якщо такі оцінки студентів не знайдені, то необхідно генерувати запит для експерта на їх складання.

Завдяки необхідності генерування запиту для експерта на складання експертної оцінки з компетентності студентів отримаємо DFD-декомпозицію *процесу* «Виконання запиту» (*варіант 3*), що показана на рисунку 4.28.

Ця ситуація повинна відображатися на діаграмі *«Обробити запит фірми»*, тобто необхідно внести додатковий *процес* «*Знайти експертні оцінки студента*».

Крім зазначених недоліків, діаграма «*Обробити запит фірми*» має ще декілька. Можна об'єднати роботи «*Скласти запрошення*» і «*Відправити студенту*» в одну роботу «*Скласти і відіслати запрошення*», спростивши діаграму.

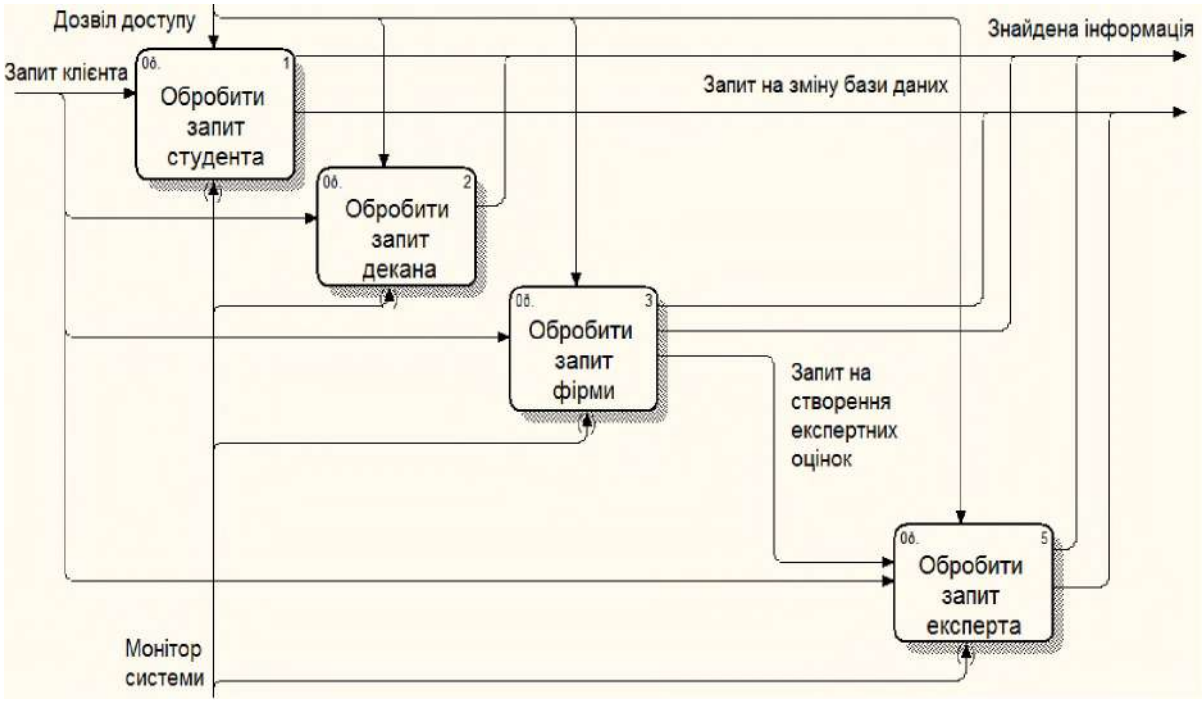


Рис. 4.28. DFD-декомпозиція *процесу* «Виконання запиту» (*варіант 3*)

Може існувати вимога фірми на складання експертної оцінки для студента.

Тоді декомпозиція *процесу* «Обробити запит фірми» буде наступною (рис. 4.29).

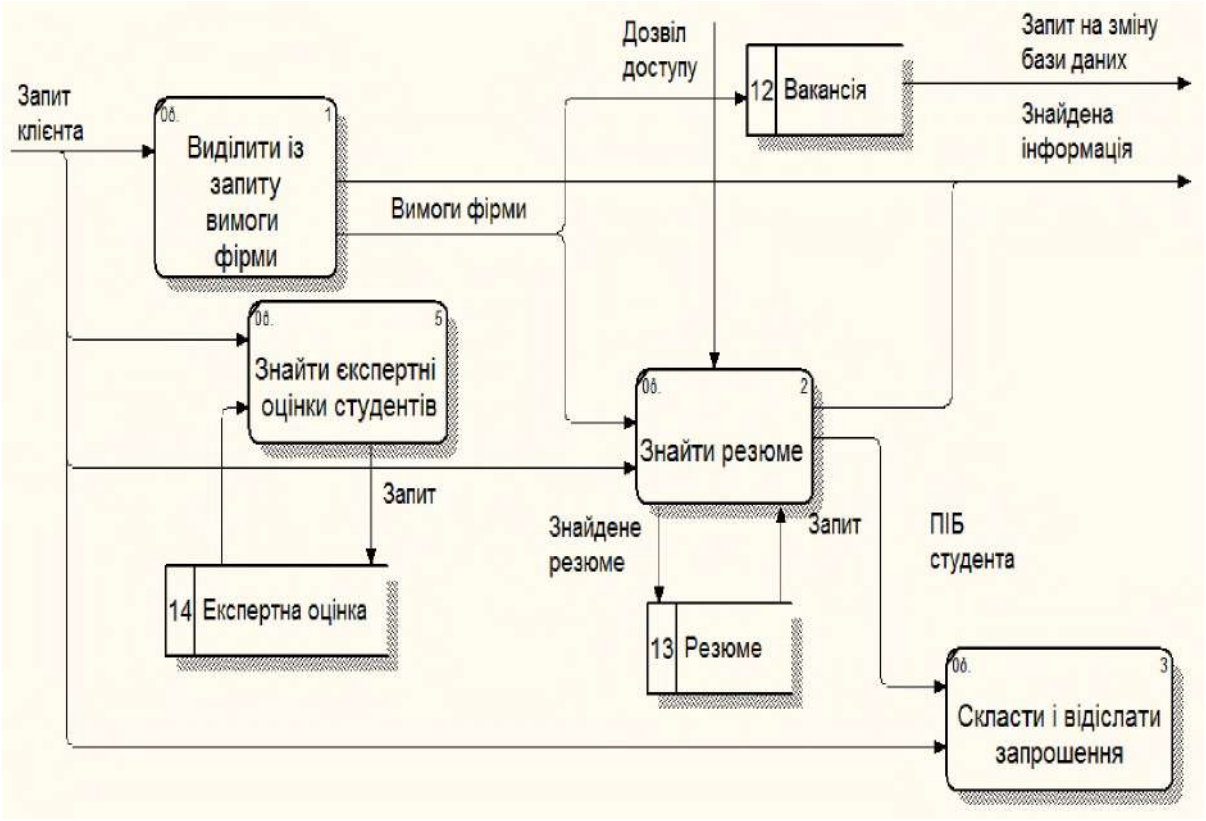


Рис. 4.29. Декомпозиція *процесу* «Обробити запит фірми» (*варіант 2*)

Проведемо аналіз варіантів діаграми «*Виконання запиту*», зображених на рисунках 4.20 і 4.25, за методикою, описаною в *попередній роботі*.

У другому варіанті скоротилося число блоків на тому самому рівні декомпозиції, спростивши діаграму.

Таким чином, зменшився коефіцієнт ***N/L***.

Розрахуємо і порівняємо коефіцієнти збалансованості діаграм.

Для 1-го варіанта Kb = |(5 + 4 + 5 + 4 + 5)/ 5 – 5|=0,4.

Для 2-го – Kb = |(5 + 4 + 5 + 5)/ 5 – 5| = 0,25.

Зменшення коефіцієнта Kb свідчить про більш рівномірний розподіл потоків даних між блоками моделі.

4.6. Лабораторна робота 6

Тема. Методологія DFD зі створення діаграм потоків даних індивідуального об'єкта дослідження та розвиток системи «Індивідуальний проект»

Мета роботи:

✓закріплення навичок з реалізації основних принципів методології DFD;

✓створення нового проекту в CASE-засобі AllFussion PM;

✓доповнення діаграмами потоків даних ФМ представницьких бізнес-процесів індивідуального об'єкта, розроблених у лабораторній роботі 5.

Завдання

1. Удосконалити «Індивідуальний проект» доповненими діаграмами потоків даних (DFD) за допомогою CASE-засобу AllFussion PM:

✓доповнити створену раніше діаграму IDEF0 діаграмою DFD;

✓додати на діаграму DFD зовнішню сутність і сховище даних;

✓пов'язати діаграму і зовнішню сутність;

✓пов'язати діаграму і сховище;

✓визначити ім'я зв'язку із зовнішньою сутністю.

Хід роботи

1. Проведення аналізу індивідуального об'єкта дослідження.

2. Виконання роботи для індивідуального об'єкта дослідження здійснюється відповідно до завдання-прототипу.

3. Виконати всі пункти, які передбачені у Завданні.

4. Враховуючи індивідуальні особливості об'єкта дослідження, допускається розширення і поглиблення робіт «Індивідуального проекту» (може ускладнитися процес отримання зворотної інформації про виконання індивідуальних завдань та їхнього корегування).

Завдання для моделювання

1. Створення діаграми потоків даних на основі аналізу діяльності організації, що відповідає об'єкту дослідження, БД якого дозволить отримати відомість.

ВІДОМІСТЬ 1

розрахунку відрядної індивідуальної заробітної плати

за цехом _____ за _____ місяць 201__ р.

код та назва цеху

Таб номер	Прізвище	Код деталі	Розряд	Професія		Кількість виготовлених деталей	Розцінка за деталь	Відроблений час	Сума заробітної плати
				Код	Назва				

ВІДОМІСТЬ 2

потреби в матеріалах на річну виробничу програму

за підприємством на 201__ р.

Код матеріалу	Назва матеріалу	Одиниці вимірювання	Код виробу	Назва виробу	Програма випуску виробів на рік	Норми витрат матеріалу на виріб	Потреба матеріалу на річну програму

ВІДОМІСТЬ 3

обліку руху матеріалів на складі ___ за ___ місяць

код та назва складу

Матеріал			Залишок Вхідний		Надходження		Видатки		Залишок вихідний	
Код	Назва	Одиниця виміру	Кількість	Вартість	Кількість	Вартість	Кількість	Вартість	Кількість	Вартість

ВІДОМІСТЬ 4

обліку якості готової продукції за цехом ___ за __ місяць

код та назва цеху

Номер дільниці	Деталь		Операція		Вид браку		Причина (винуватець)		Кількість деталей	
	Код	Назва	Код	Назва	Код	Назва	Код	Назва	Придатних	Бракованих

ВІДОМІСТЬ 5

відвантаження готової продукції за _____ місяць

Код покупця	Назва покупця	Код продукції	Назва продукції	№ угоди на відвантаження	Кількість за угодою	Фактично відвантажено	Відхилення

Оформлення звіту

Звіт про виконану роботу оформляється згідно із «Загальними рекомендаціями до виконання лабораторних робіт та оформлення звіту» (додаток А: А.1, А.2).

Контрольні запитання

1. Що описує діаграма DFD?
2. Що описують сховища?
3. Що описують зовнішні сутності?
4. Яка нотація використовується в CASE-засобі AllFussion PM для побудови діаграм DFD?
5. Перерахувати складові діаграми DFD.
6. У чому сутність призначення процесу?
7. Що називають зовнішньою сутністю?
8. Пояснити механізм доповнення діаграми IDEF0 діаграмою DFD.

4.7. Теоретичні аспекти методології IDEF3

Діаграми ***IDEF3*** також називають ***WorkFlow diagramming*** – методологією моделювання, що використовує графічний опис інформаційних потоків, відношень між процесами обробки інформації й об'єктів, що є частиною цих процесів.

Діаграми ***WorkFlow*** використовуються для аналізу процедур обробки інформації.

Мета – надати аналітикам опис послідовності виконання процесів, а також об'єктів, що беруть участь спільно в одному процесі.

Методологія ***IDEF3*** може бути також використана як метод створення процесів. Методологія IDEF3 доповнює методологію

IDEF0 і містить усе необхідне для побудови моделей, які можуть бути використані для імітаційного моделювання.

Структурні компоненти IDEF3 моделі-діаграми та одиниці роботи. *Діаграма* є основною одиницею опису в ***IDEF3***-моделі. Організація діаграм в ***IDEF3*** є найбільш важливою, якщо модель редагується кількома людьми. У цьому випадку розробляювач повинен визначати, яка інформація входитиме в ту або іншу модель.

Одиниці роботи – ***Unit of Work*** (*UOW*), також можуть іменуватися *роботами*, є центральними компонентами моделі.

В ***IDEF3*** *роботи* зображуються прямокутниками і мають ім'я, що позначає процес дії і номер (ідентифікатор). В ім'я, звичайно, включається основний результат *роботи* (наприклад, постановлення завдань бізнесу).

UOW *(Unit of Work)* – термін, що використовується в BPwin для позначення функції, регулярно переводиться в текстах як «*одиниця роботи*». У дійсності слово *work*, крім його загальновживаного перекладу як «*робота*», має ще цілий ряд варіантів перекладу. У тексті ***IDEF3*** для позначення цього оператора використовується інший термін – ***UOB*** *(Unit of Behavior)* – «*одиниця поведінки*», що відповідає суті *IDEF3*-методики, що описує послідовність подій, що відбуваються при роботі модельованої системи, тобто системи, що описує події.

Свідченням того, що два ці терміни позначають те саме, є також зауваження в тексті AllFusion Process Modeler. Так що при використанні терміна Unit of Work правильно було б використовувати його переклад як «*одиниця поведінки, одиниця подій, подія*».

Структурні компоненти IDEF3 моделі-зв'язки. Зв'язки показують взаємовідношення робіт *(подій)*. Усі зв'язки в IDEF3 є односпрямованими.

Старша (Precedendce) лінія – суцільна лінія (→), зв'язуюча одиниця робіт *(подій)*. Рисується зліва на право або зверху до низу. Показує, що *робота-джерело* повинна закінчитися перш ніж *робота-мета* почнеться.

Лінія відношення (Relation Link) – пунктирна лінія (→), що використовується для зображення зв'язків між одиницями робіт, а також між одиницями робіт і об'єктами посилань.

Потоки об'єктів (Object Flow) – стрілка із двома наконечниками (–>>), застосовується для опису застосування об'єкта у двох або більше одиницях роботи, наприклад коли об'єкт породжується в одній роботі й використовується в іншій.

Перехрестя (Junction) – використовуються для відображення логіки взаємодії стрілок при злитті і розгалуженні або для відображення безлічі подій, які можуть або повинні бути завершені перед початком наступної роботи (табл. 4.7).

Таблиця 4.7

Типи перехресть

Позначення	Найменування	Зміст у випадку злиття стрілок	Зміст у випадку розгалуження стрілок
	*Asynchronous AND*	Усі попередні процеси повинні бути завершені	Усі наступні процеси повинні бути запущені
	*Synchronous AND*	Усі попередні процеси завершені одночасно	Усі наступні процеси запускаються одночасно
	*Asynchronous OR*	Один або кілька попередніх процесів повинні бути завершені	Один або кілька наступних процесів повинні бути запущені
	*Synchronous OR*	Один або кілька попередніх процесів завершені одночасно	Один або кілька наступних процесів запускаються одночасно
	*XOR (Exclusive OR)*	Тільки один процес завершений	Тільки один наступний процес запускається

Розрізняють перехрестя для злиття (*Fan-in Junction*) і розгалуження (*Fan-out Junction*) стрілок. Перехрестя не може використовуватися одночасно для злиття і для розгалуження. Для внесення перехрестя слугує кнопка.

Об'єкти-посилання є спеціальними символами, які посилаються на зовнішні частини опису процесу (табл. 4.8).

Таблиця 4.8

Типи об'єктів-посилань

Тип об'єкта	Мета опису
OBJECT	Описує участь важливого об'єкта в роботі *(події)*
GOTO	Інструмент циклічного переходу (у повторюваній послідовності робіт), можливо на поточній діаграмі, але не обов'язково. Якщо всі роботи *(події)* циклу присутні на поточній діаграмі, цикл може також зображуватися стрілкою, що вертається на стартову роботу *(подію)*. GOTO може посилатися на перехрестя
UOB Unit of behavior	Застосовується, коли необхідно підкреслити багатократне використання будь-якої роботи, але без циклу. Наприклад, робота «Контроль якості» може бути використана у процесі «Виготовлення виробу» кілька разів після кожної одиничної операції. Звичайно, цей тип посилання не використовується для моделювання робіт, що запускаються автоматично
NOTE	Використовується для документування важливої інформації, що належить до будь-яких графічних об'єктів на діаграмі. NOTE є альтернативою внесенню текстового об'єкта в діаграму
ELAB Elaboration	Використовується для вдосконалення графіків або їх більш детального опису. Звичайно вживається для детального опису розгалуження й злиття стрілок на перехрестях

Вони додаються на діаграму для того, щоб звернути увагу редактора на будь-що важливе, що неможливо пов'язати зі стрілкою, роботою або перехрестям.

Для внесення об'єкта-посилання слугує кнопка R. Об'єкт-посилання відображається у вигляді прямокутника. Об'єкти-посилання повинні бути пов'язані з одиницями робіт або перехрестями пунктирними лініями. При внесенні об'єктів-посилань необхідно вказати їхній тип.

Порівняльна характеристика IDEF0, DFD і IDEF3 методологій. Process flow diagram (***діаграма потоку процесів***) – IDEF3 або діаграма потоку робіт – це метод підтримки графічного опису того, що роблять система або бізнеси.

IDEF3 – метод процесного опису, основна мета якого полягає в тому, щоб забезпечити структурний метод (моделювання), яким експерт (предметної) області може описати ситуацію як установлену послідовність подій, а також описати об'єкти, що беруть участь у цьому.

IDEF3 моделює передування і причинність відношень між ситуаціями і подіями у формі, звичною для експерта предметної області, забезпечуючи структурний метод знанням того, як система, процес або організація функціонують. Функціональне моделювання в IDEF3-методиці значно відрізняється від функціонального моделювання в IDEF0 і DFD як за складом операторів, так і за прийнятими нотаціями.

Особливості цієї методики визначаються, в першу чергу, тим, що вона відображає функції системи в причинній послідовності їх здійснення. Розглядаються паралельні або альтернативні процеси.

Ця відмінність викликає періодичну необхідність рішення питань про вибір можливих варіантів послідовності дій або координації цих дій. Така необхідність виникає щоразу, коли зв'язки, що символізують в IDEF3 умови початку здійснення функції, розгалужуються або зливаються, що приводить до використання при моделюванні в IDEF3 нового оператора – «*перехрестя*» (Junction). Введення в IDEF3 перехрестя є обов'язковою умовою кожного злиття і кожного розгалуження стрілок (*на відміну від IDEF0 і DFD не допускається розгалуження і злиття стрілок інакше, ніж через перехрестя).*

Подібно DFD IDEF3 не має будь-якої регламентації функціонального призначення стрілок-зв'язків, крім «вхід» і «вихід». Відсутні також будь-які топологічні обмеження, крім вимоги розгалуження і злиття всіх зв'язків IDEF3 тільки через перехрестя. Допускається ведення паралельних зв'язків між роботами-подіями.

4.8. Технологія реалізації методології IDEF3 на прикладі інформаційної системи «Компетентність студента вишу»

За допомогою діаграм IDEF3 моделюють послідовності *робіт*, що мають технологічні і часові зв'язки. До таких моделей можна віднести проект розробки ІС «Компетентність студента вишу». Перед початком моделювання необхідно створити ієрархічну структуру *робіт*, що описує процес створення ІС, яку можна представити у вигляді трьох блоків (формування вимог та технічного завдання; створення технічного проекту; реалізація та впровадження), склад яких подано нижче.

1. Формування вимог та технічного завдання

1.1. Постановка бізнес-задачі

✓ *оцінка майбутнього програмного продукту на основі ключових факторів успіху організації;*

✓ *проведення аналізу реалізованості проекту.*

1.2. Виявлення вимог

✓ *аналіз предметної області замовника;*

✓ *спілкування з носіями знань і фахівцями в предметній області замовника;*

✓ *визначення високорівневих вимог.*

1.3. Аналіз та специфікація вимог

✓*детальне опрацювання усіх бізнес-вимог (замовника та виконавця);*

✓*аналіз і пріоритизація всіх бізнес-вимог;*

✓*створення прототипу для демонстрації та перевірки технічних вимог до програмного продукту;*

✓*узгодження загального розуміння вимог до продукту і розкриття непорозуміння між замовником та виконавцем.*

1.4. Розробка технічного завдання

✓ *складання технічного завдання на розробку програмного продукту;*

✓ *затвердження технічного завдання на розробку програмного продукту.*

2. Створення технічного проекту

2.1. Проектування БД та додатків

✓*визначення об'єктів;*

✓*визначення категорій користувачів;*

✓*визначення запитів користувачів;*

✓*вибір СУБД;*

✓*проектування логічної (інфологічної та концептуальної) моделі БД;*

✓*проектування фізичної структури БД.*

2.2. Розробка модульної структури обробки даних

✓*розробка модульної структури всієї системи;*

✓*розробка модульної структури підсистеми обробки запитів, визначення категорії користувачів;*

✓*розробка модульної структури підсистеми обробки експертних оцінок;*

✓*розробка модульної структури підсистеми обробки професійних і психологічних тестів;*

✓*розробка модульної структури контролю успішності студента.*

3. Реалізація та впровадження

3.1. Реалізація

✓*створення внутрішнього рівня баз даних за підтримки відповідного процесу управління проектом;*

✓*написання програмного коду.*

3.2. Тестування додатків (на всіх етапах)

✓ *планування процесів тестування (функціонального, інтерфейсу користувача, продуктивності, здатності навантаження, стресове, установки і розгортання, приймальне, інтеграційне, регресивне тощо);*

✓ *створення варіантів тестування;*

✓ *виконання різних тестів, залежних від вимог.*

3.3. Здача та впровадження

✓ *установка системи в робоче середовище, конфігурація, перенесення і завантаження даних;*

✓ *розробка документації (технічна, призначена для користувача);*

✓ *проведення навчання користувачів і передача знань.*

3.4. Супровід і підтримка

✓ *своєчасне оновлення;*

✓ *забезпечення виконання поставлених завдань.*

Відповідно до опису структури робіт з розробки системи «Компетентність студента вишу» визначимо діаграми, додавши на них взаємозв'язки між роботами (рис. 4.30).

На стадії формування вимог та розробки технічного завдання замовник системи відіграє важливу роль, надаючи розроблювачам необхідну інформацію для створення системи.

Тому на діаграмі показаний відповідний об'єкт-посилання, що впливає на роботу «Формування вимог та технічного завдання».

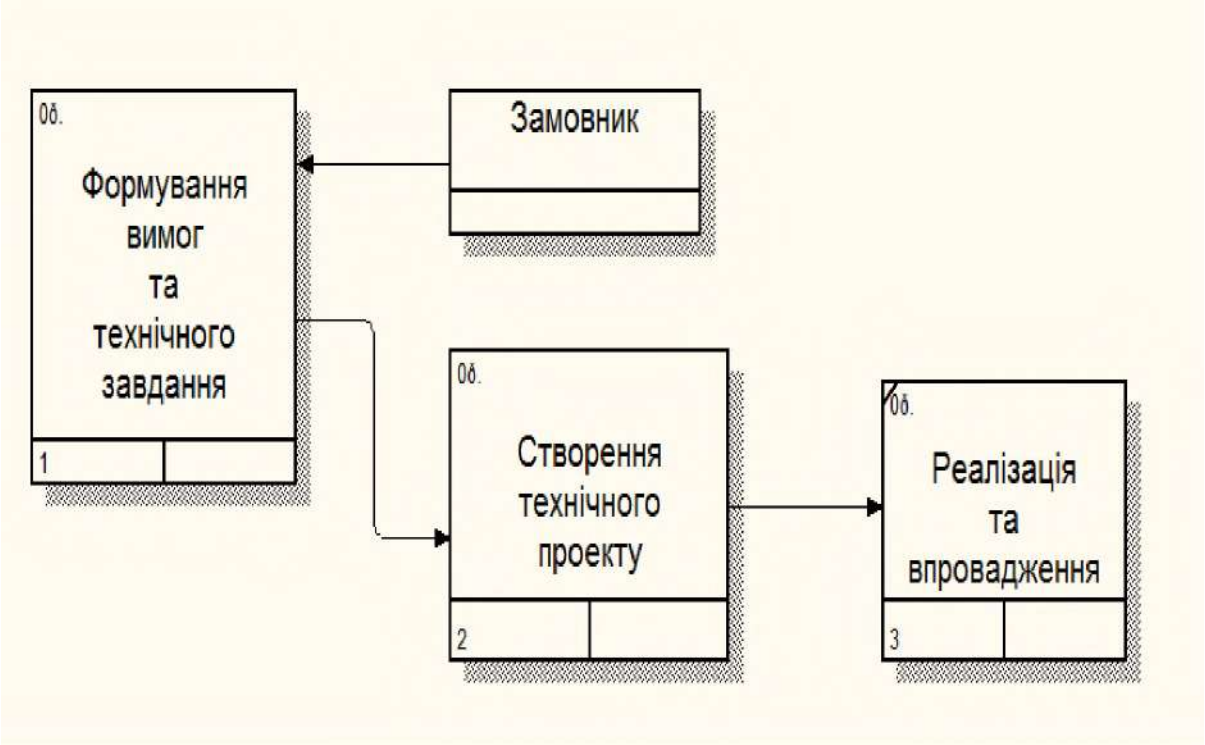


Рис. 4.30. Діаграма системи «Компетентність студента вишу»

Орієнтуючись на створену ієрархічну структуру робіт, проведемо декомпозицію робіт зі створення системи «Компетентність студента вишу», представлену на рисунках 4.31–4.35:

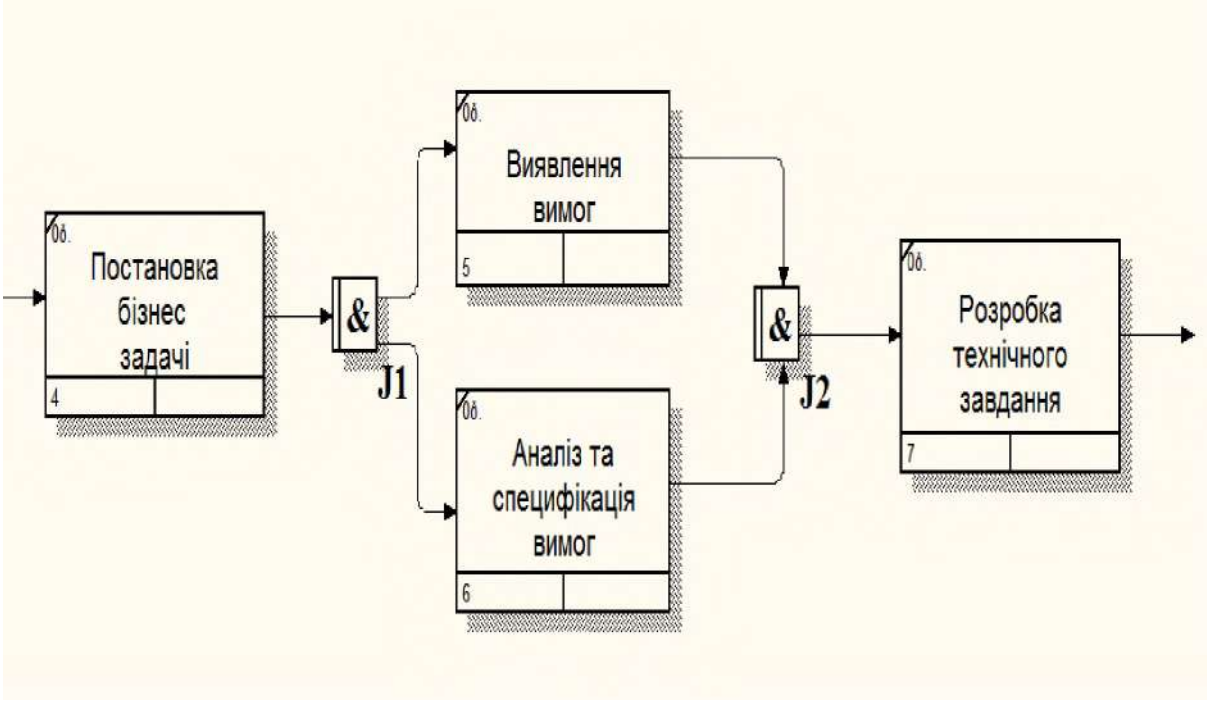


Рис. 4.31. Декомпозиція роботи «Формування вимог та ТЗ»

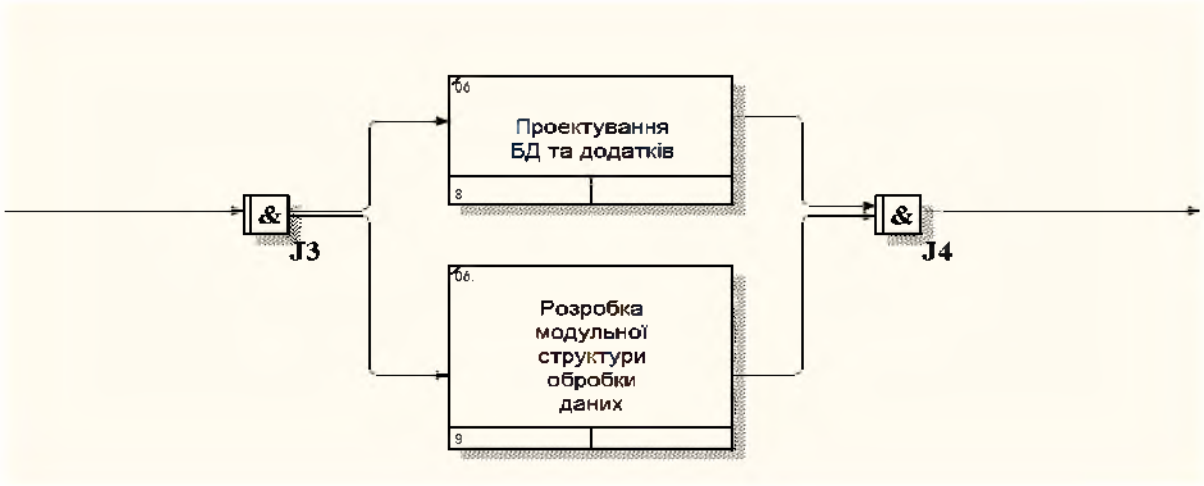


Рис. 4.32. Декомпозиція роботи «Створення технічного проекту»

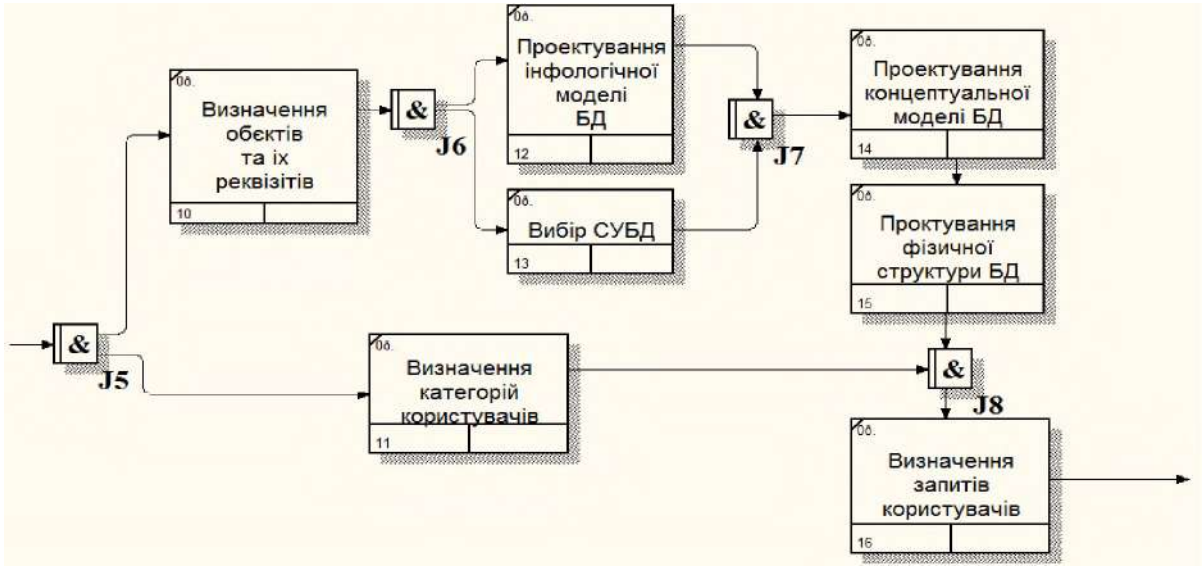


Рис. 4.33. Декомпозиція роботи «Проектування БД та додатків»

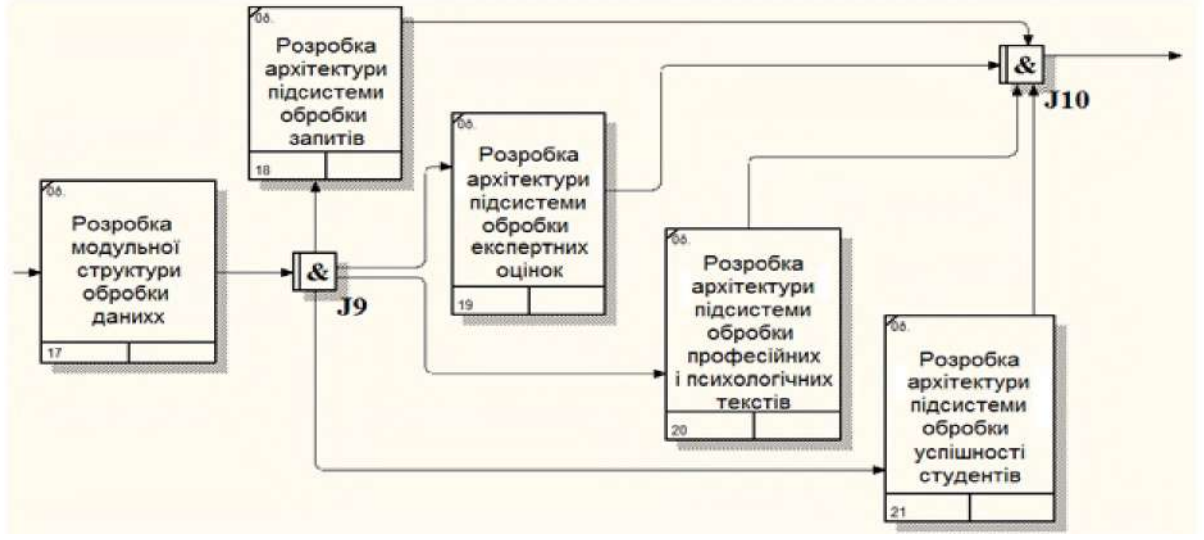


Рис. 4.34. Декомпозиція роботи «Розробка модульної структури обробки даних»

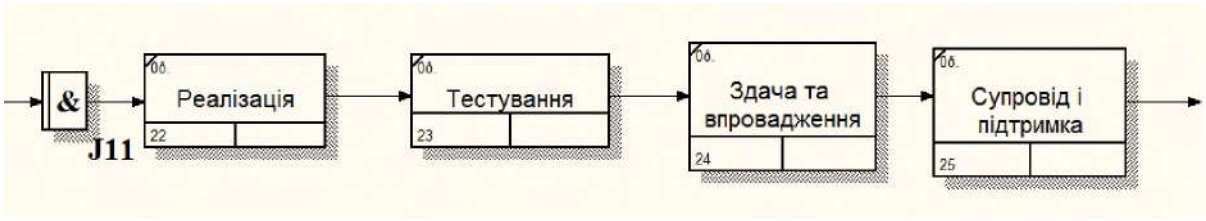


Рис. 4.35. Декомпозиція роботи «Реалізація та впровадження» (варіант 1)

Команда тестувальників бере участь у проекті, починаючи з ранніх етапів розробки (планування процесу тестування і створення варіантів тестування) і переходячи потім безпосередньо до виконання різного роду тестів залежно від вимог. З урахуванням зазначеної діаграми з декомпозиції роботи «Реалізація та впровадження» доцільно представити з деталізацією видів тестування на окремих етапах і отримати другий варіант її представлення на рисунку 4.36.

Створення пакета діаграм на основі структури робіт за підсистемами обробки даних ІС

Опишемо роботи, необхідні для створення конкретних підсистем обробки даних ІС «Компетентність студента вишу», поданих на рис. 4.34.

Структура робіт підсистем може бути представлена таким чином:

1. Розробка підсистеми обробки професійних і психологічних тестів:
   - ✓ *Визначення міжсистемних узгоджень;*
   - ✓ *Визначення категорій користувачів;*
   - ✓ *Створення запитів до системи;*
   - ✓ *Створення інтерфейсу підсистеми;*
2. Розробка підсистеми обробки запитів:
   - ✓ *визначення міжсистемних узгоджень;*
   - ✓ *визначення категорій користувачів;*
   - ✓ *створення запитів до системи;*
   - ✓ *створення інтерфейсу підсистеми.*

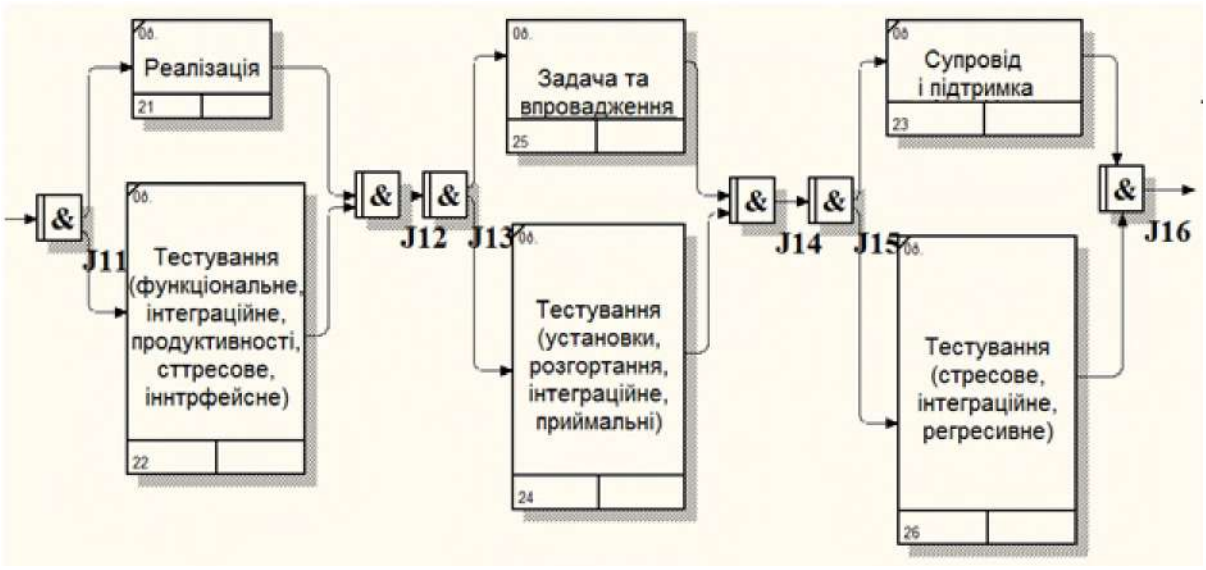


Рис. 4.36. Декомпозиція роботи «Реалізація та впровадження» (варіант 2)

3. Розробка підсистеми обробки експертних оцінок:

✓ *визначення міжсистемних узгоджень;*

✓ *визначення категорій користувачів;*

✓ *створення запитів до системи;*

✓ *створення інтерфейсу підсистеми.*

4. Розробка підсистеми контролю успішності студента:

✓ *визначення міжсистемних узгоджень;*

✓ *визначення категорій користувачів;*

✓ *створення запитів до системи;*

✓ *створення інтерфейсу підсистеми.*

5. Розробка архітектури обробки даних за типовим фрагментом.

При формуванні структури операцій «*за підсистемами*» виявилася можливість створення типового фрагмента проектування підсистеми, що включає той самий перелік *робіт*.

Такий підхід часто спрощує опис проектів, дозволяючи формувати проекти будь-якої складності з невеликих фрагментів.

Виділимо отриманий типовий фрагмент проектування підсистеми в окрему діаграму (рис. 4.37).

Отримані на рисунках 4.34 і 4.37 діаграми описують процес створення системи «Компетентність студента вишу» на основі структури робіт за підсистемами обробки даних.

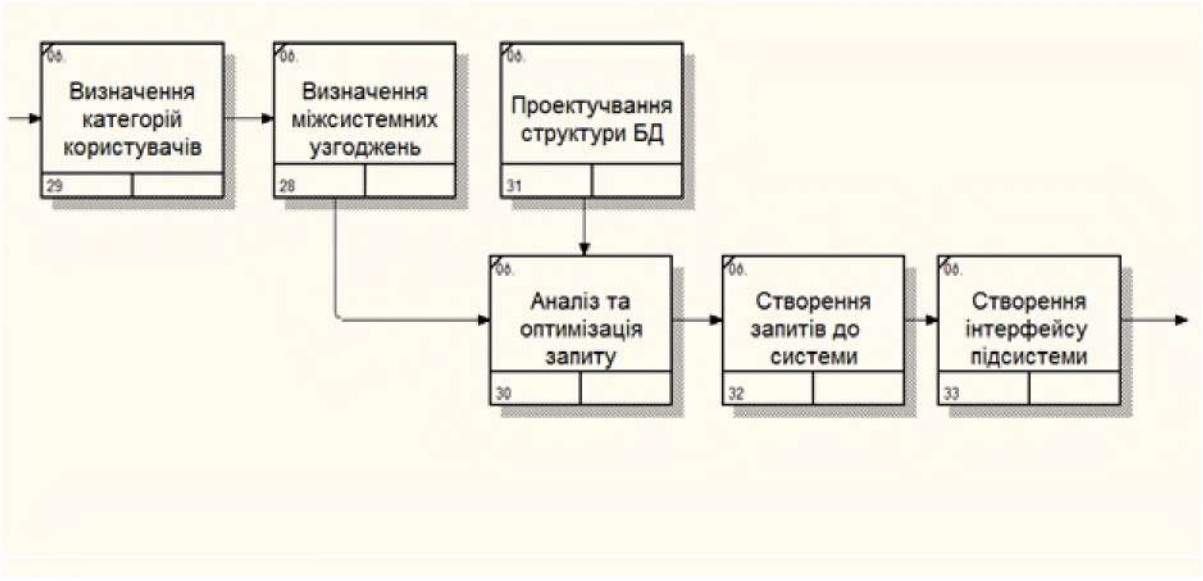


Рис. 4.37. Типовий фрагмент «Розробка підсистеми обробки даних ІС»

Завдання для закріплення

1. Розробка уточнюючих варіантів декомпозиції робіт зі створення інформаційної системи «Компетентність студента вишу».

Контрольні запитання

1. Перерахувати складові елементи діаграм IDEF3.
2. Що показують зв'язки в діаграмах IDEF3?
3. Перерахувати типи стрілок у діаграмах IDEF3.
4. Що називається перехрестям?
5. Назвіть типи перехресть.
6. Що називається об'єктом-посиланням?
7. Які бувають типи об'єктів-посилань?
8. Як додати об'єкт-посилання?

4.9. Лабораторна робота 7

Тема. Методологія IDEF3 з опису взаємозв'язків між процесами індивідуального об'єкта та розвиток системи «Індивідуальний проект»

Мета роботи:

✓закріплення навичок з реалізації основних принципів методології IDEF3;

✓створення ієрархічної структури робіт, що описує процеси індивідуального об'єкта з розробки певної системи, наприклад БД системи «Індивідуальний проект»;

✓створення нового проекту в CASE-засобі AllFussion Process Modeller пакета AllFussion Modeling Suite;

✓створення діаграм з опису послідовності реалізації процесів індивідуального об'єкта та в цілому вдосконалення системи «Індивідуальний проект».

Завдання

1. Удосконалити систему «Індивідуальний проект», який отримано під час виконання Лабораторної роботи, 6 створенням діаграм з опису послідовності реалізації процесів (IDEF3) за допомогою CASE-засобу AllFussion PM таким чином:

✓розробити ієрархічну структуру робіт, що описує процеси індивідуального об'єкта з розробки певної системи, наприклад БД системи «Індивідуальний проект»;

✓створити діаграму IDEF3, що визначає послідовність розробки концептуальної моделі БД системи «Індивідуальний проект» у 3НФ;

✓пов'язати роботи на діаграмі;

✓додати на діаграму перехрестя, що моделюють паралельні події під час заповнення БД;

✓додати об'єкт-посилання і пов'язати його з діаграмою.

Хід роботи

1. Проведення аналізу індивідуального об'єкта дослідження.

2. Виконання роботи для індивідуального об'єкта дослідження здійснюється відповідно до завдання-прототипу.

3. Виконання всіх пунктів, які передбачені у завданні.

4. Враховуючи індивідуальні особливості об'єкта дослідження, допускається:

✓розширення і поглиблення робіт «Індивідуального проекту»;

✓може ускладнитися процес отримання зворотної інформації про виконання індивідуальних завдань та їхнього корегування.

Оформлення звіту

Звіт про виконану роботу оформляється згідно із «Загальними рекомендаціями до виконання лабораторних робіт та оформлення звіту» (додаток А: А. 1, А. 2).

Завдання для моделювання

1. Розробити ієрархічну структуру робіт, що описують процеси організації, що відповідає об'єкту дослідження (БД якого дозволить отримати відомість).

2. Створити діаграму IDEF3, що визначає розроблену послідовність робіт.

ВІДОМІСТЬ 1

розрахунку відрядної індивідуальної заробітної плати

за цехом _____ за _____ місяць 201__ р.

код та назва цеху

Таб номер	Прізвище	Код деталі	Розряд	Професія		Кількість виготовлених деталей	Розцінка за деталь	Відроблений час	Сума заробітної плати
				Код	Назва				

ВІДОМІСТЬ 2

потреби в матеріалах на річну виробничу програму

за підприємством на 201__ р.

Код матеріалу	Назва матеріалу	Одиниці вимірювання	Код виробу	Назва виробу	Програма випуску виробів на рік	Норми витрат матеріалу на виріб	Потреба матеріалу на річну програму

ВІДОМІСТЬ 3

обліку руху матеріалів на складі ___ за ___ місяць

код та назва складу

Матеріал			Залишок вхідний		Надходження		Видатки		Залишок вихідний	
Код	Назва	Одиниця виміру	Кількість	Вартість	Кількість	Вартість	Кількість	Вартість	Кількість	Вартість

ВІДОМІСТЬ 4

обліку якості готової продукції за цехом ___ за __ місяць

код та назва цеху

Номер дільниці	Деталь		Операція		Вид браку		Причина (винуватець)		Кількість деталей	
	Код	Назва	Код	Назва	Код	Назва	Код	Назва	Придатних	Бракованих

ВІДОМІСТЬ 5

відвантаження готової продукції за _____ місяць

Код покупця	Назва покупця	Код продукції	Назва продукції	№ угоди на відвантаження	Кількість за угодою	Фактично відвантажено	Відхилення

4.10. Технології створення звітів у седовищі CASE-засобу AllFussion Process Modeller

Існує два способи створення звітів у CASE-засобі AllFussion Process Modeller:

- ✓ за допомогою вбудованих шаблонів;
- ✓ за допомогою Report Template Builder.

Для створення звітів за функціонального моделлю можна також використовувати зовнішні генератори звітів третіх фірм, наприклад Crystal Reports.

Типи звітів в AllFussion PM, створюваних за допомогою вбудованих шаблонів.

AllFussion Process Modeller дозволяє створювати такі типи звітів:

1) звіт за моделлю (Model Report) – містить у собі всю інформацію моделі, створеної в CASE-засобі (IDEF0, IDEF3 або DFD);

2) звіт про діаграму (Diagram Report) – містить у собі інформацію про всі об'єкти, що входять в активну діаграму AllFussion PM;

3) звіт про об'єкти діаграми (Diagram Object Report) – містить повний список об'єктів, таких як роботи, сховища, зовнішні посилання, із вказівкою їхніх властивостей;

4) звіт про вартість робіт (Activity Cost Report) – містить дані про вартість робіт і вартісні центри моделі;

5) звіт про стрілки (Arrow Report) – містить у собі інформацію про стрілки і зв'язки моделі;

6) звіт про використання даних (Data Usage Report) – містить інформацію про таблиці БД, сутності й атрибути, зіставлені з роботами моделі, а також дії, які можуть бути зроблені над ними;

7) звіт погодженості з методологією (Model Consistency Report) показує, наскільки активна IDEF0-модель відповідає обраній методології.

Перераховані вище звіти викликаються вибором – відповідно підпункту з меню Reports головного вікна. При цьому відкривається діалогове вікно для завдання параметрів формованого звіту.

Кожний отриманий звіт може бути відкритий у режимі перегляду (кнопка Preview), роздрукований (кнопка Print) або збережена у файл (кнопка Report).

Створення звіту за моделлю

Для цього необхідно:

1. Відкрити модель, за якою збираєтеся створювати звіт.
2. Обрати Model Report з меню Report головного вікна. При цьому відкриється діалог звіту за моделлю.
3. Установити у вікні, що відкрилося, опції відповідно до пунктів, які будуть включені у звіт. Порядок включення відображається поруч із прапорцем:

✓ Model Name – назва моделі.
✓ Definition – ціль бізнес-процесів моделі.
✓ Scope – предметна область моделі.
✓ View point – точка зору на модель.
✓ Time frame – тимчасові рамки моделі.
✓ Status – ступінь готовності моделі.
✓ Purpose – ціль створення моделі.
✓ Source – джерело, на підставі якого створюється модель.
✓ Author name – автор моделі.
✓ Creation date – дата створення.
✓ System last revision date – дата останнього перегляду в системі.
✓ User last revision date – дата останнього перегляду користувачем.

4. Обрати форму подання звіту (Preview, Print, Report).

Створення звіту за діаграмою

При створенні цього типу звіту необхідно звертати увагу на методологію діаграми, оскільки залежно від цього виробляється настроювання параметрів звіту.

1. Для діаграм IDEF0 параметри задаються в рамках Activity options і Link options. Параметри в інших рамках не мають змісту. Наприклад, група параметрів для сховищ даних (Data Store) не має змісту для IDEF0-діаграм.

2. Для діаграм IDEF3 параметри задаються в рамках Activity Options, Link Options, Junction Options і Referent Options.

3. Для DFD-діаграм - у рамках Activity Options, Link Options, Data store Options і External Options.

Створення звіту складається з таких дій:

1. Відкрийте діаграму, за якою хочете створити звіт.

2. Виберіть Diagram Report з меню Report, відкривши діалог створення звіту по діаграмі.

3. У вікні, що відкрилося, розташовуються списки властивостей об'єктів, згруповані в шість рамок:

- ✓ Activity Options – властивості робіт.
- ✓ Data store Options – властивості сховищ даних.
- ✓ External Options – властивості зовнішніх посилань.
- ✓ Link Options – властивості зв'язків (стрілок).
- ✓ Junction Options – властивості перехресть.
- ✓ Referent Options – довідкова інформація.

Включення кнопки, розташованої поруч із властивістю, поміщає його у звіт.

4. Оберіть форму подання звіту (Preview, Print, Report).

Створення звіту про об'єкти діаграми

Аналогічно попередньому звіту встановлювані опції повинні відповідати методології діаграми.

1. Для IDEF0-діаграм оберіть опцію Activities, що включає у звіт властивості робіт.

2. Для IDEF3-діаграм можна вибрати одну або кілька опцій: Activities – включає у звіт властивості робіт, Data stores – включає

у звіт властивості сховищ даних, External reference – включає у звіт властивості об'єктів зовнішніх посилань.

3. Для діаграм DFD можна обрати опцію Activities, що сформує звіт за властивостями робіт (інформаційними процесами).

Створення звіту здійснюється за таким алгоритмом:

1. Відкрийте діаграму, за якою хочете створити звіт.

2. Оберіть пункт Diagram Object Report з меню Report. За допомогою спадаючого списку Standard Reports можна обрати назву стандартного звіту, настроювання якого були збережені раніше. У рамках Activity Options і Arrow Options задається, відповідно перелік властивостей робіт і стрілок, що включаються у звіт. Формат звіту задається в рамці Report Format. Звіт можна створювати за всіма декомпозованими діаграмами певної роботи, що задається в спадаючому списку Start From Activity. Глибина декомпозиції задається в поле Number of Levels.

Способи впорядкування робіт і стрілок у звіті вказуються в рамках Activity Ordering і Arrow ordering.

3. Оберіть спосіб подання звіту (Preview, Print, Report).

Створення звіту за стрілками *реалізується за таким алгоритмом:*

1. Відкрийте діаграму, за якою хочете створити звіт.

2. Оберіть із меню Report пункт Arrow Report. При цьому відкриється діалогове вікно звіту за стрілками.

Склад і функції цього вікна аналогічні іншим звітам. У рамках Arrow Report Dictionary (*Основні властивості стрілок*), Source/Dest (*Початок і кінець стрілок*), Arrow Bundle (*Розгалуження й злиття стрілок*) розташовані опції, кожна з яких відповідає одній із властивостей стрілок. Установка такої опції поміщає відповідну властивість стрілки у звіт.

Опція Diagram Arrow визначає склад звіту. Якщо встановити цю опцію, то у звіт будуть включені стрілки активної діаграми. Якщо скинути – то у звіт включаються всі стрілки на всіх діаграмах відкритої моделі.

При формуванні звіту можна скористатися збереженими раніше настроюваннями (дивися стандартні звіти).

3. Оберіть спосіб подання звіту (Preview, Print, Report).

Створення звіту погодженості з методологією

Цей тип звіту фактично дозволяє виявити синтаксичні помилки в моделях IDEF0, які підрозділяються на три типи:

1. *Невиявлені помилки.* До цього типу помилок належать неправильні найменування об'єктів.

2. *Помилки, що не дозволяються.* До цього типу помилок належить відповідність граней робіт типам стрілок, що входять і виходять з них. У IDEF0 кожна грань роботи призначена тільки для певного типу стрілок. Наприклад, не можна створити внутрішню стрілку, що виходить із лівої грані роботи, й вхідну – в праву.

3. *Виявлені, але дозволені помилки.* До такого типу помилок належать, зокрема, наявність неіменованих об'єктів, незв'язаних кінців стрілок тощо.

Звіт про погодженість із методологією не має параметрів. Для його виклику необхідно скористатися пунктом Model Consistency Report, викликаним з головного меню Report.

Стандартні звіти

Для звітів про об'єкти діаграми, вартість робіт, стрілки й про використання даних можна формувати так звані стандартні звіти. Стандартні звіти являють собою сукупність настроювань, збережених під певним ім'ям. Кожний з перерахованих вище звітів має свої стандартні звіти за замовчуванням. Наприклад, звіт про стрілки має стандартний звіт Arrow Definition/Note.

При виклику стандартного звіту в діалоговому вікні відновлюються збережені в ньому опції. Наприклад, якщо в діалоговому вікні звіту про стрілки обрати в списку Standard Reports стандартний звіт Arrow Definition/Note, то встановляться опції Arrow Name, Definition, Note, Diagram Arrows, Fixed Columns, Header, Merge і Remove Special Char.

Крім існуючих стандартних звітів, можна створювати нові. Для цього в діалоговому вікні звіту встановіть усі необхідні опції, уведіть ім'я стандартного звіту в рамці Standard Report і натисніть New. Установлені параметри збережуться під уведеним ім'ям.

Створення звітів за допомогою Report Template Builder

Report Template Builder є простим генератором звітів, що входить до складу AllFusion PM. Викликається цей інструмент за

допомогою кнопки на панелі інструментів або з меню *Tools/Reports Builder*. Основною перевагою цього інструмента є можливість публікації результатів моделювання на веб-Сайті.

Пункт меню Tools/Reports Builder викликає діалог Report Templates, що слугує для створення шаблона звіту. Кнопка New слугує для створення нового шаблона, кнопка Edit – для редагування існуючого.

Список вибору Output Type дозволяє задати формат результату виконання звіту. Звіт може бути експортований у текстовий формат, RTF і HTML. Кнопка Run дозволяє виконати звіт.

Клацання по кнопці New або Edit викликає діалог діалогу Report Template Builder.

Призначення кнопок панелі інструментів діалогу Report Template Builder таке:

1) створити новий шаблон;

2) відкрити існуючий шаблон;

3) зберегти шаблон. Якщо шаблон зберігається вперше, пропонується внести ім'я шаблона;

4) виконати звіт;

5) виклик діалогу Properties; видалення пункту звіту.

Діалог Report Template Builder містить два списки й панель інструментів. У лівій частині діалогу втримується список можливих секцій звіту, що відповідають типам об'єктів моделі, у правій – список секцій, включених у звіт.

Особливий інтерес представляє розділ Graphical. За його допомогою можна включати у звіт графічні об'єкти (секція Picture) – діаграми моделей даних або моделей процесів.

Для створення нової секції звіту необхідно обрати тип об'єкта моделі й клацнути по кнопці. За замовчуванням у звіт включається тільки ім'я об'єкта. Для включення інших властивостей необхідно за допомогою меню Edit/Properties або відповідної кнопки на панелі інструментів викликати діалог Properties.

Вкладка Property Tree утримує ієрархічний список властивостей секції Activity, що дозволяє включити у звіт властивості роботи.

Вкладка Section дозволяє змінити стиль, розмір і колір шрифту секції. Залежно від типу об'єкта, що редагується, діалог Properties може мати додаткові вкладки.

У розділі Type два варіанти настроювання – Current і Specify at Run Time. Опція Current означає включення у звіт поточної діаграми або збереженого відображення. Опція Specify at Run Time дозволяє обрати необхідну діаграму під час перегляду звіту. Далі буде розглянутий приклад звіту AllFusion PM, у якому обрана опція Specify at Run Time.

Особливий інтерес представляє можливість обрати відображення властивостей об'єктів у секції Picture.

Для генерації звіту варто клацнути по кнопці «Виконати звіт». Якщо у звіт включена секція Picture і встановлена опція Specify at Run Time, виникає діалог Diagram Selection and Scaling, у якому можна обрати діаграми для включення їх у звіт.

У результаті виконання звіт експортується або в текстовий файл формату RTF або у файл формату HTML. У лівій частині сторінки перебуває фрейм із переліком секцій звіту. Якщо клацнути по найменуванню діаграми в списку, у правій частині з'являється обрана діаграма. Якщо клацнути по об'єкту діаграми, з'являється вікно зі списком параметрів об'єкта, обраних у списку секції Picture діалогу Properties.

Запитання для закріплення

1. Які існують способи створення звітів у CASE-засобі AllFussion PM?
2. Яку інформацію можна отримати за допомогою звіту щодо моделі?
3. Яку інформацію можна отримати за допомогою звіту діаграми?
4. Яку інформацію можна отримати за допомогою звіту об'єкти діаграми?
5. Яку інформацію можна отримати за допомогою звіту про вартість робіт?
6. Яку інформацію можна отримати за допомогою звіту про стрілки?

7. Яку інформацію можна отримати за допомогою звіту про використання даних?

8. Яку інформацію можна отримати за допомогою звіту погодженості?

4.11. Лабораторна робота 8

Тема. Створення звітів у середовищі CASE-засобу AllFussion Process на основі системи «Індивідуальний проект»

Мета роботи:

✓ ознайомитись із способами створення звітів в AllFusion PM;

✓ вивчити види звітів;

✓ освоїти методи пошуку помилок у діаграмах, використовуючи звіти.

Завдання

1. На основі системи «Індивідуальний проект», отриманої в результаті реалізації методології структурного моделювання за допомогою AllFussion Process Modeller пакета AllFussion Modeling Suite, сформувати зміст звіту.

2. Створити звіти за моделями діаграм, створених на лабораторних роботах 5–7:

➢ відкрити діалогове вікно звіту за стрілками і сформувати в ньому стандартний звіт, що містить інформацію про початки і кінці стрілок;

➢ зберегти стандартний звіт під ім'ям Arrows Source/Dest;

➢ створити звіт погодженості з методологією;

➢ зберегти отриманий звіт у файл;

➢ перевірити звіт на наявність повідомлень про помилки в моделі.

Хід роботи

1. Ознайомитись із способами створення звітів у CASE-засобі AllFussion Process Modeller пакета AllFussion Modeling Suite, вивчити види звітів та освоїти методи пошуку помилок у діаграмах, використовуючи звіти на прикладі системи «Компетентність студента вишу».

2. Створити необхідні звіти в CASE-засобі AllFussion Process Modeller на основі системи «Індивідуальний проект» за результатами виконання Лабораторних робіт 5, 6 і 7:

✓ створити звіт щодо моделі за діаграмою IDEF0;

✓ зберегти звіт у файл;

✓ відкрити діалогове вікно звіту за стрілками і сформувати в ньому стандартний звіт, що містить інформацію про початки й кінці стрілок;

✓ зберегти стандартний звіт під ім'ям Arrows Source/Dest;

✓ створити звіт погодженості з методологією;

✓ зберегти отриманий звіт у файл;

✓ перевірити звіт на наявність повідомлень про помилки в моделі.

Оформлення звіту

Звіт про виконану роботу оформляється згідно із «Загальнимих рекомендаціями до виконання лабораторних робіт та оформлення звіту» (додаток А: А.1, А.2).

Контрольні запитання

1. Назвіть типи звітів в AllFussion PM.
2. Опишіть процедуру створення звіту щодо моделі.
3. Що містить у собі звіт щодо моделі?
4. Опишіть процедуру створення звіту за діаграмою.
5. Що містить у собі звіт за діаграмою?
6. Опишіть процедуру створення звіту про об'єкти діаграми.
7. Що містить у собі звіт про об'єкти діаграми?
8. Опишіть процедуру створення звіту за стрілками.
9. Що містить у собі звіт за стрілками?
10. Опишіть процедуру створення звіту погодженості з методологією.
11. Що містить у собі звіт погодженості з методологією?
12. Яким чином здійснюється пошук помилок у діаграмах за допомогою звіту погодженості з методологією?
13. У які формати можна експортувати звіти?
14. Які види стандартних звітів існують в AllFussion PM?
15. Опишіть процедуру створення користувального звіту.

РОЗДІЛ 5
МОДЕЛЮВАННЯ ЗА ДОПОМОГОЮ CASE-ЗАСОБУ ALLFUSION ERWIN DATAMODELER (DM) ПАКЕТА ALLFUSSION MODELING SUITE COMPUTER ASSOCIATES

5.1. Теоретичні аспекти методології IDEF1X

Методологія структурного аналізу і проектування може бути орієнтована як на опис функцій, виконуваних системою, так і на опис об'єктів, що складають систему.

Якщо в першому випадку методологія структурного аналізу і проектування призначена для побудови функціональної моделі системи, тобто для відображення виконуваних системою дій і зв'язків між цими діями, то в другому випадку вона призначена для опису об'єктів, що входять у систему, їх властивостей і взаємозв'язків між ними. Фактично, у другому випадку методологія структурного аналізу і проектування слугує для опису моделі даних мовою і в термінах реляційної моделі даних. Методологія структурного аналізу і проектування, що слугує для опису моделей даних у термінах реляційної моделі, відома як методологія IDEF1X.

CASE-засоби, зокрема AllFussion ERwin Data Modeler, що підтримують цю методологію, дозволяють будувати логічну, незалежну від СУБД модель даних для загального представлення системи і її складових об'єктів, а також фізичну модель даних, яка може бути трансформована в будь-яку реляційну СУБД і описана на мові опису даних цієї СУБД. CASE-засіб AllFussion ERwin Data Modeler, наприклад, підтримує більше 20 СУБД, на мові опису даних яких може бути згенерована фізична модель даних.

Проектування реляційної моделі. Реляційна модель даних передбачає опис даних з урахуванням певної сукупності правил і обмежень. Основним з них є вимога відсутності надмірності і дублювання даних. Вимоги до проектування реляційної моделі даних відображені в процесі декомпозиції даних, яка виконується з метою виключення надмірності даних і називається нормалізацією.

Розрізняють декілька нормальних форм, які представляють дані на різних етапах декомпозиції. Для коректного використання даних у середовищі реляційної СУБД вважається необхідним і достатнім приведення даних до третьої нормальної форми.

Основні поняття реляційної моделі. Дані в реляційній моделі зберігаються у вигляді таблиць. Таблиця складається із заголовка (стовпців або атрибутів) і рядків або записів (кортежів).

Поле таблиці – це значення, що лежить на перетині рядка і стовпця.

Сукупність значень, які може приймати поле (або всі значення стовпця таблиці), називається доменом таблиці.

Реляційні таблиці називаються відношеннями (Relation – відношення).

Число стовпців таблиці називається степенем відношення.

Число рядків таблиці називається потужністю відношення.

Внутрішні обмеження реляційної моделі даних. Дані в реляційній моделі повинні відповідати таким вимогам, які називаються внутрішніми обмеженнями моделі:

1. У таблиці кожен запис повинен бути унікальним.
2. Конкретні дані повинні зберігатися лише в одній таблиці.
3. Число відношень у моделі має бути оптимальним.

Унікальність запису в таблиці. Первинний ключ відношення. Для ідентифікації запису в таблиці служить атрибут або сукупність атрибутів, яка носить назву первинного ключа відношення. Для кожного запису таблиці значення первинного ключа є унікальним. Декомпозиція (нормалізація) таблиць заснована на дослідженні залежностей атрибутів таблиці від атрибутів первинного ключа.

Нормалізація. Перша нормальна форма (1НФ). Таблиця (Відношення) знаходиться в першій нормальній формі, якщо значення кожного атрибута в кожному полі таблиці має атомарну структуру, тобто воно неподільне, не є списком, не містить вкладеності значень.

Друга нормальна форма (2НФ). Відношення знаходиться в другій нормальній формі, якщо воно вже знаходиться в першій нормальній формі, і всі неключові атрибути функціонально повно

залежать від атрибутів первинного ключа, тобто повністю ними визначаються.

Таким чином, усі неключові атрибути залежать від усього ключа. При цьому ключ може бути складовим, тобто конкатенацією кількох атрибутів.

Якщо у відношенні є атрибути, які залежать від частини первинного ключа, то вони виносяться в окрему таблицю і їм зіставляється частина ключа, від якої вони залежать.

Для прикладу розглянемо таблицю «Комп'ютерний клас» (рис. 5.1) БД для зберігання даних про комп'ютер навчального класу.

Номер класу	№ комп'ютера	ПІП адміністратора класу	Телефон адміністратора класу	Тип процесора	Наявність Інтернету

Рис. 5.1. Таблиця «Комп'ютерний клас» БД

Таблиця знаходиться в 1НФ, оскільки значення кожного атрибута є атомарним. У цій таблиці первинний ключ є складовим, тобто конкатенацією кількох атрибутів. Це сукупність атрибутів «Номер класу» та «Номер комп'ютера». Атрибути «ПІП адміністратора класу» і «Телефон адміністратора класу» залежать тільки від частини ключа – від атрибута «Номер класу». Для приведення таблиці до 2-ї нормальної форми її треба декомпозувати на дві таблиці БД: «Комп'ютер» (рис. 5.2) і «Клас» (рис. 5.3).

Номер класу	Номер комп'ютера	Тип процесора

Рис. 5.2. Таблиця «Комп'ютер» БД

Номер класу	ПІП адміністратора класу	Телефон адміністратора класу	Наявність Інтернету

5.3. Таблиця «Клас» БД

Третя нормальна форма (3НФ). Відношення знаходиться в третій нормальній формі, якщо воно вже знаходиться в другій, і в ньому відсутні транзитивні залежності між атрибутами.

Транзитивна залежність – це залежність одного атрибута від іншого через третій. Якщо А залежить від В (В → А), а С залежить від А (А → С), то С залежить від В транзитивно (В →А → С).

Для виключення транзитивної залежності атрибути, які залежать від первинного ключа транзитивно, виносяться в окрему таблицю, де їм зіставляється атрибут, через який вони залежать від первинного ключа.

Для прикладу доповнимо таблицю «Комп'ютер» БД даними про фірму-виробника процесора. У результаті отримаємо таку таблицю (рис. 5.4).

Номер класу	Номер комп'ютера	Тип процесора	Фірма-виробник процесора	Телефон фірми-виробника процесора

Рис. 5.4. Таблиця «Комп'ютер» БД

У даній таблиці атрибути «Фірма-виробник процесора» та «Телефон фірми-виробника процесора» залежать від первинного ключа «Номер класу» транзитивно через атрибут «Тип процесора». Для приведення до 3НФ доповнена таблиця «Комп'ютер» декомпозується дві таблиці БД: «Комп'ютер» (рис. 5.5) і «Процесор» (рис. 5.6).

Номер класу	Номер комп'ютера	Тип процесора

Рис. 5.5. Таблиця «Комп'ютер» БД

Тип процесора	Фірма-виробник процесора	Телефон фірми-виробника процесора

Рис. 5.6. Таблиця «Процесор» БД

Приведення таблиць до третьої нормальної форми в модельному прикладі вважається достатнім для того, щоб завершити декомпозицію.

У цілому процес проектування реляційної моделі даних можна описати у вигляді такої послідовності дій:

1. Виділити інформаційні об'єкти модельованої системи.

2. Описати кожен інформаційний об'єкт набором характеристик (атрибутів), які представляють важливість з точки зору виконуваних системою функцій.

3. Визначити для кожного інформаційного об'єкта первинний ключ-атрибут або сукупність атрибутів.

4. Описати дані кожного інформаційного об'єкта у вигляді таблиці (відношення) так, щоб дані в кожному полі таблиці були атомарними, тобто кожна таблиця повинна бути приведена до 1 нормальної форми.

5. Привести відношення до другої нормальної форми. Для цього декомпозують за необхідності кожну таблицю так, щоб у ній залишилися тільки атрибути, які залежать від усього первинного ключа. Тобто видаляють у нові таблиці елементи даних (атрибути), які залежні від окремих компонентів первинного ключа. У нових таблицях (відношеннях) компоненти первинного ключа вихідного відношення, від яких залежали видалені атрибути, зіграють роль первинного ключа.

6. Привести відношення до третьої нормальної форми. Для цього у нові відношення винести елементи даних (атрибути), які залежать від атрибутів первинного ключа транзитивно.

7. Описати кожне з отриманих відношень у вигляді:

<Ім'я_відношення> (<атрибут, який є первинним ключем>, <атрибут>, ... <атрибут>).

Моделі даних у середовищі CASE-засобу AllFussion ERwin DM. Проектування моделі даних у стандарті IDEF1X у середовищі CASE-засобу AllFussion ERwin DM виконується відповідно до вимог реляційної моделі даних.

У середовищі AllFussion ERwin DM можна побудувати моделі даних двох основних типів: логічні і фізичні моделі даних. Моделі даних є багаторівневими. На кожному рівні модель представляє певне бачення даних.

Логічні моделі даних. Логічні моделі даних є універсальними. За допомогою цих моделей відображається абстрактний по-

гляд на дані. Об'єкти логічної моделі відповідають об'єктам реального світу і можуть носити такі ж назви. Наприклад: «Відділ», «Співробітник», «Фірма» тощо.

Логічна модель незалежна і не пов'язана з конкретною СУБД. Сутності в логічній моделі описуються набором атрибутів, але при цьому не вказується тип даних атрибутів. Реалізація логічної моделі у середовищі конкретної СУБД буде оригінальною для кожної СУБД. Тому одній логічній моделі можуть відповідати кілька фізичних моделей, а саме для кожної СУБД – своя фізична модель. Розрізняють три рівні логічної моделі, що представляються своїм виглядом діаграм: діаграма «Сутність»/«Зв'язок» (Entity Relationship Diagram), модель даних, що заснована на ключах (Key-Based Model), повна атрибутивна модель (Fully Attributed Model).

Діаграма «Сутність»/«Зв'язок» представляє найвищий рівень у моделі даних. Модельована система описується набором сутностей і зв'язків між ними. Діаграма представляє дані системи в цілому для їх подальшої деталізації.

Модель даних, що заснована на ключах, являє більш детальний опис даних системи. Модель включає таблиці, їхні ключові атрибути та зв'язки між таблицями за однаковими значеннями атрибутів.

Повна атрибутивна модель представляє докладний опис усіх атрибутів таблиць і зв'язки між таблицями. Відповідає реляційному поданню даних у третій нормальній формі.

Фізичні моделі даних. Фізична модель даних представляє всю інформацію про дані, яка необхідна для реалізації моделі в конкретній СУБД. Розрізняють два рівні фізичної моделі: трансформаційна модель (Transformation Model) та модель СУБД (DBMS Model).

Трансформаційна модель містить усю інформацію, необхідну для реалізації в середовищі конкретної СУБД. Модель надає можливість перевірити відповідність фізичної моделі даних вимогам модельованої системи.

Модель СУБД формується шляхом автоматичної генерації з трансформаційної моделі і є відображенням системного каталогу СУБД.

5.2. Технологія реалізації методології IDEF1X у середовищі CASE-засобу AllFussion ERwin DM

Для освоєння методології IDEF1X зі створення проекту БД за допомогою CASE-засобу AllFusion ERwin Data Modeler як прототипу використовується концептуальна модель БД «Успішність», що розроблена для об'єкта дослідження «Навчальний заклад» (рис. 5.7).

Для проектування бази даних у середовищі CASE-засобу AllFussion ERwin DM необхідно виконати таку послідовність дій:

- ✓ запуск засобу AllFussion ERwin DM;
- ✓ створення нової моделі бази даних або відкриття раніше створеної моделі;
- ✓ створення сутностей та робота з моделлю;
- ✓ заповнення сутностей атрибутами;
- ✓ побудова зв'язків;
- ✓ встановлення параметрів таблиць бази даних та друк звіту.

***Запуск* CASE-засобу**. Запуск засобу здійснюється із середовища ОС Windows завдяки завантаженню через кнопочне меню «Пуск» за таким форматом:

Пуск/ Программы/ Computer Associates/ AllFusion ERwin Data Modeler r.7/ ERwin Data Modeler r.7.

Після запуску пакета автоматично відкриється вікно діалогу, у якому слід обрати опцію:

Open an existing file – для вибору та відкриття вже створеної моделі (рис. 5.8) та натиснути кнопку ОК.

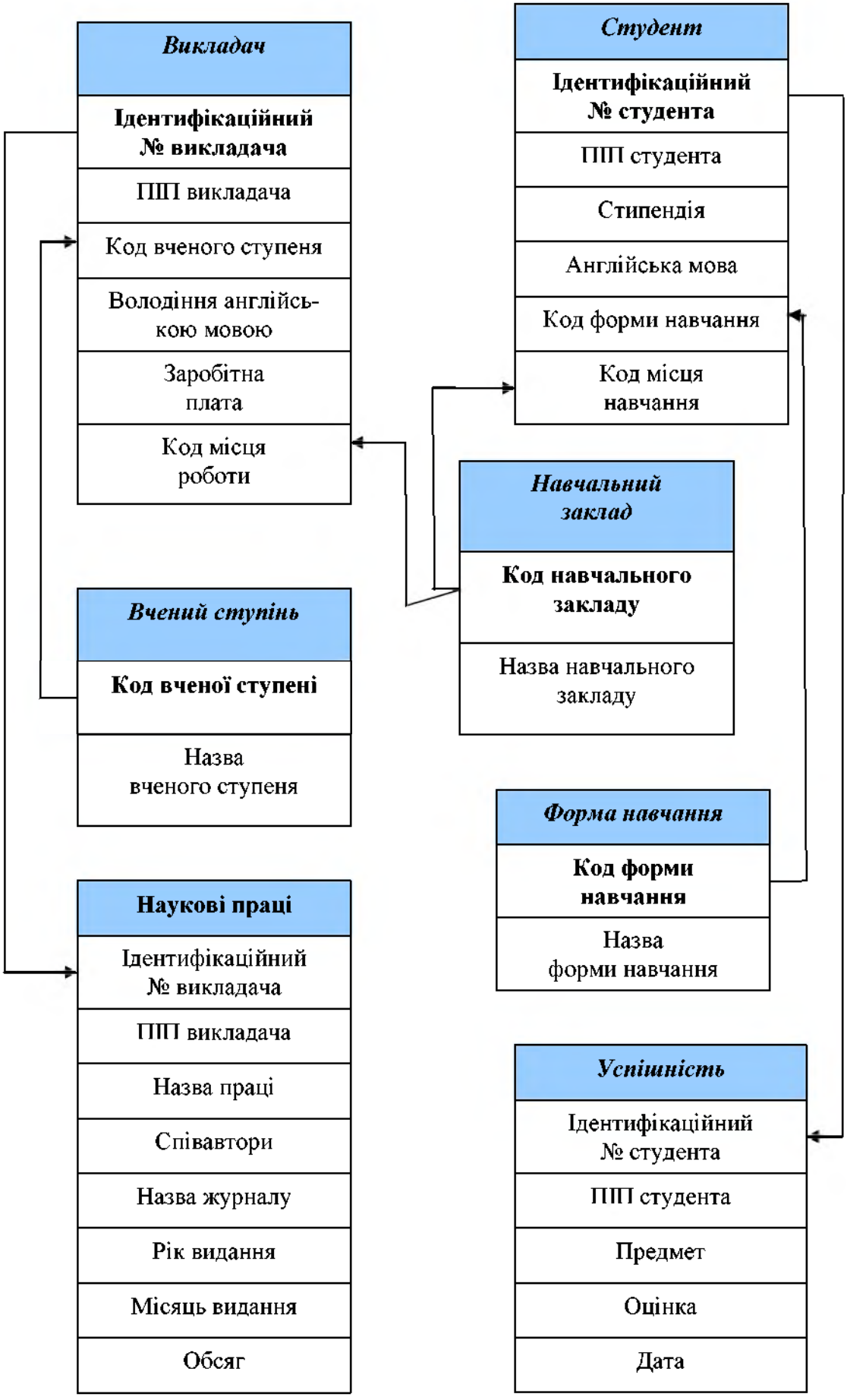


Рис. 5.7. Концептуальна модель БД «Успішність»

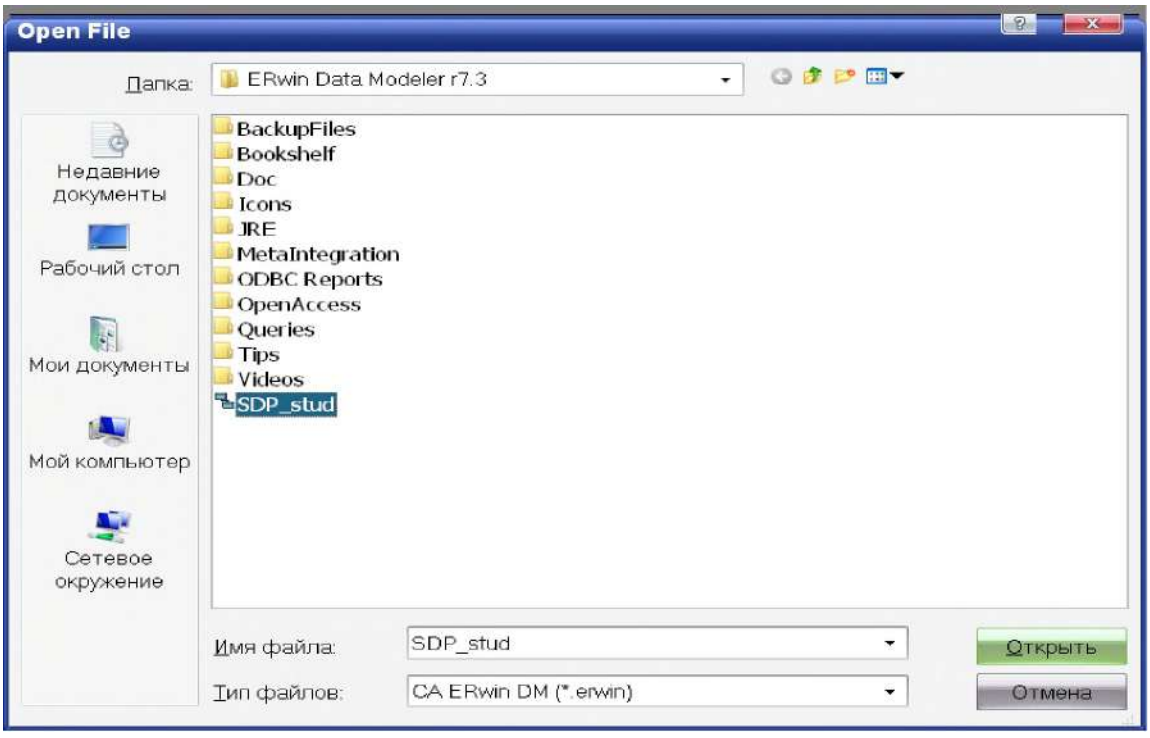


Рис. 5.8. Початкове діалогове вікно пакета AllFusion ERwin DM

Створення нової моделі. Обрати пункт меню ***File/New*** – відкриється вікно ***Create Model - Select Template***. За допомогою групи кнопок ***New Model Type*** обрати ***Logical/Physical*** та в цьому ж вікні, у діалозі ***Target Database***, обрати цільову *СУБД (Access)* та натиснути кнопку ***OK*** (рис. 5.9).

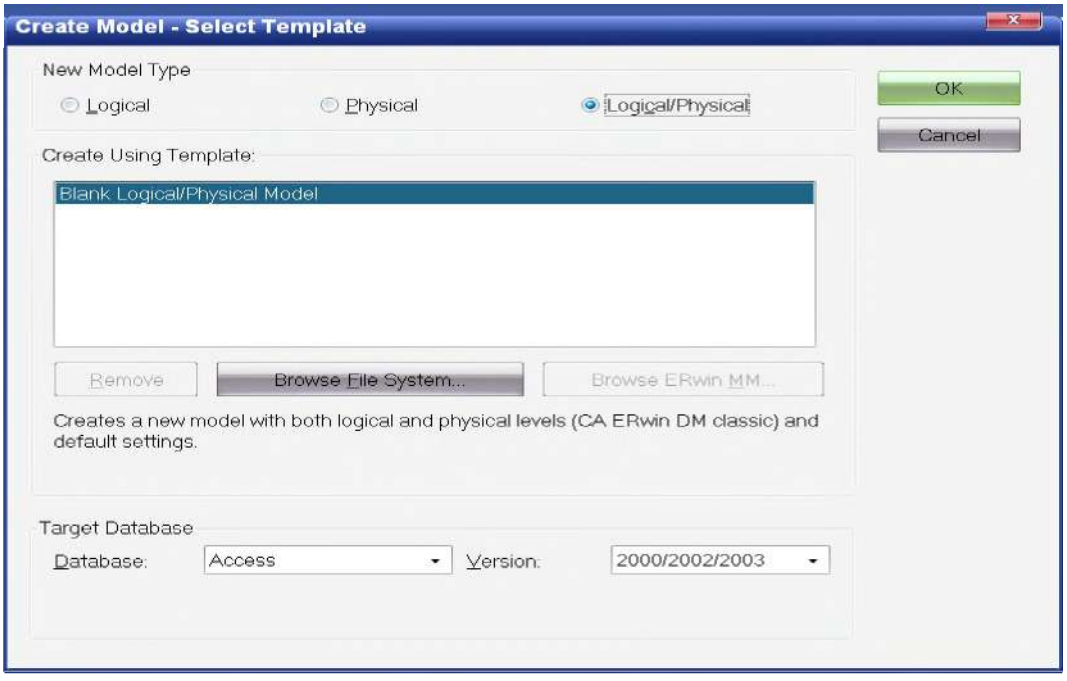


Рис. 5.9. Діалогове вікно для вибору типу моделі та шаблону

Далі необхідно задати загальні властивості моделі: з головного меню ***Model/Model Properties*** – відкрити вікно опису властивостей (рис. 5.10).

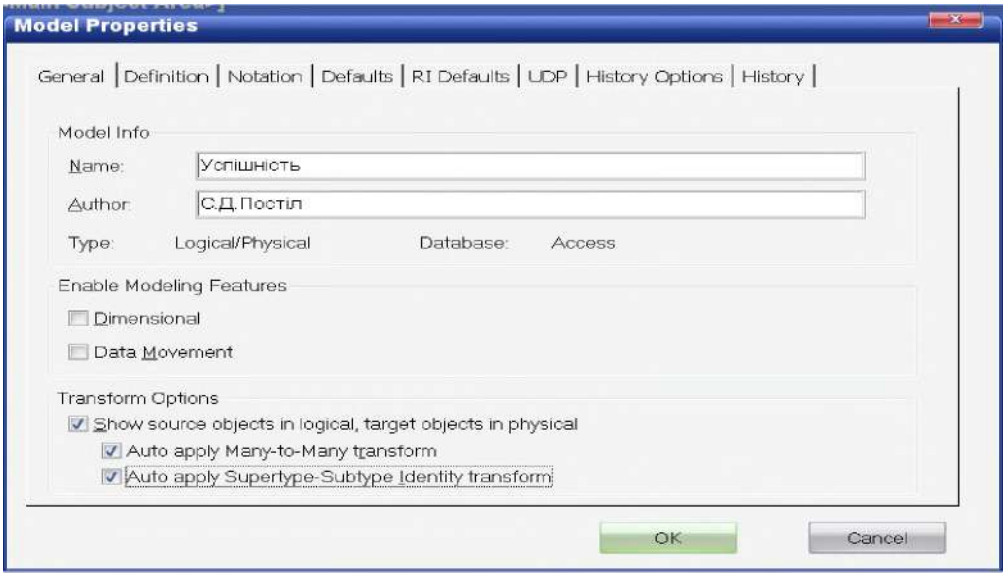


Рис. 5.10. Діалогове вікно властивостей моделі

Спочатку на вкладці ***General*** вводимо: ім'я моделі бази даних, ПІП автора моделі. Також необхідно активізувати групу опцій ***Auto-Transform Logical Objects***, яка надає додаткової функціональності моделі бази даних при переході до логічного рівня: вмикає автоматичне розв'язання проблеми відтворення зв'язку «*багато до багатьох*» та проблеми відображення конструкції «*супертип – підтип*». На вкладці ***Definition*** записуємо розгорнутий опис моделі. На вкладці ***Notation*** слід пересвідчитися, що для логічного та фізичного рівнів обрана методологія ***IDEF1X*** (рис. 5.11).

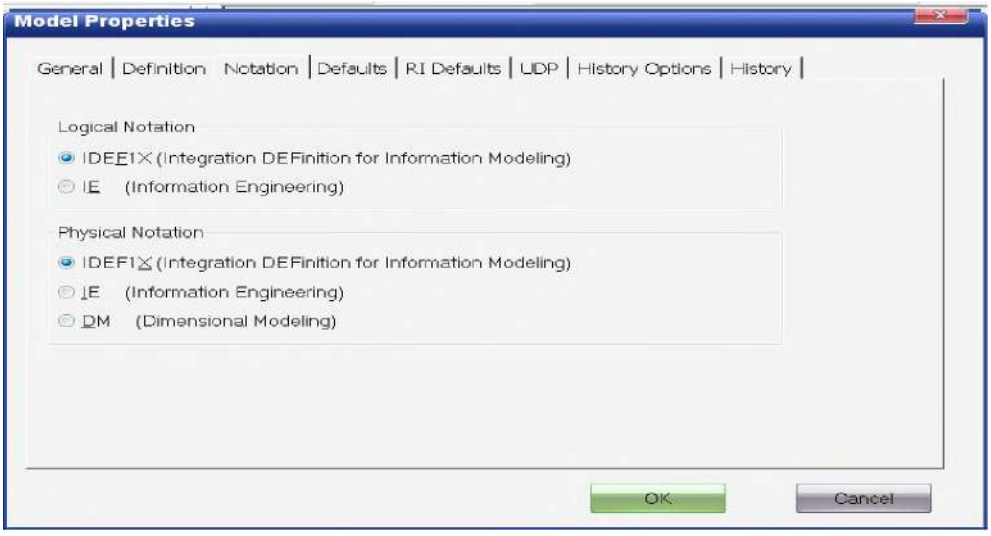


Рис. 5.11. Діалогове вікно вибору методології *IDEF1X*

Створення сутностей та робота з моделлю

Для роботи з моделлю у середовищі CASE-засобу AllFussion ERwin DM існує такий перелік панелей інструментів (рис. 5.12):

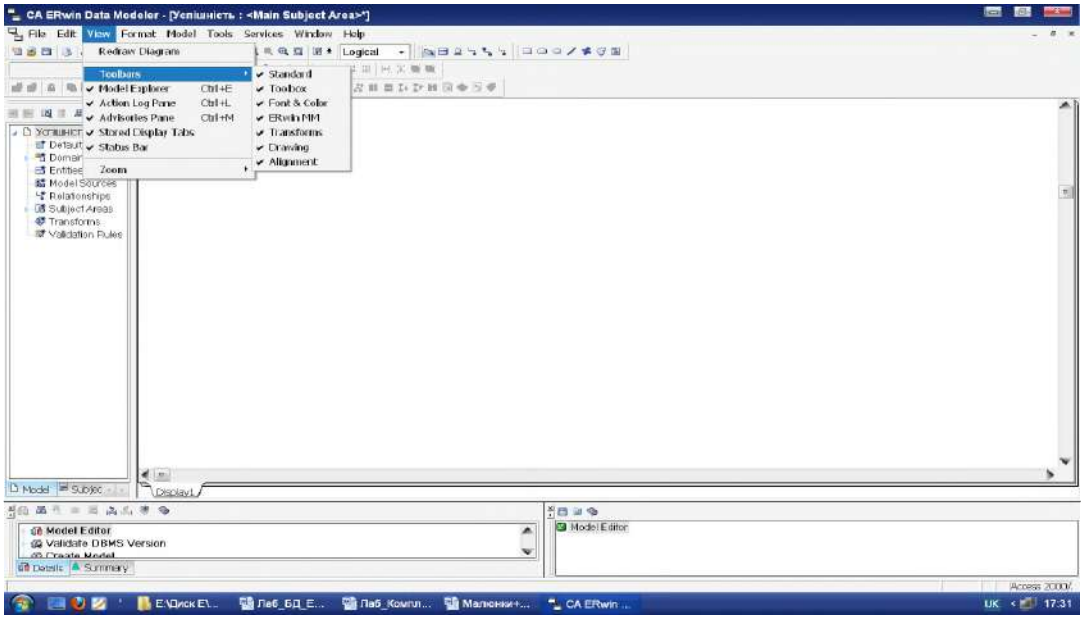


Рис. 5.12. Перелік панелей інструментів для роботи з моделлю у середовищі CASE-засобу AllFussion ERwin DM

При проектуванні бази даних використовуються такі основні елементи панелей інструментів:

✓ створення, відкриття, збереження і друк моделі;

✓ виклик діалогу Report Browser для генерації звітів;

✓ вибір рівня відображення моделі: рівень сутностей, рівень атрибутів і рівень опису;

✓ зміна масштабу;

✓ перемикач між предметними областями моделі;

✓ генерація схеми БД, синхронізація схеми БД з моделлю і вибір сервера СУБД (доступно лише на фізичному рівні);

✓ перемикач між логічним та фізичним рівнями моделі;

✓ панель інструментів Toolbox на логічному рівні бази даних, відповідно: кнопка вибору об'єктів, кнопка створення сутності, кнопка категоріального зв'язку, ідентифікуючий зв'язок, зв'язок Б:Б, неідентифікуючий зв'язок;

✓ панель інструментів Toolbox на фізичному рівні бази даних, відповідно: кнопка вибору об'єктів, кнопка створення сутності, кнопка створення таблиці-представлення, кнопка категоріального

зв'язку, ідентифікуючий зв'язок, створення зв'язку між таблицею і представленням, неідентифікуючий зв'язок;

✓ панель інструментів «Малювання» для створення графічних об'єктів удосконалення зовнішнього вигляду та сприйняття моделі;

✓ панель інструментів для поліпшення зовнішнього вигляду та сприйняття моделі.

Оскільки в CASE-засобі AllFussion ERwin DM існує два рівні представлення і моделювання БД: логічний і фізичний, то для початку роботи зі створення елементів моделі за допомогою поля зі списком перемкнемо CASE-засіб на роботу з логічним рівнем моделі.

Обравши піктограму створення сутності та клацнувши в робочій області моделі, створюємо в CASE-засобі AllFussion ERwin DM попередньо спроектовані на папері сутності відповідно до концептуальної моделі БД «*Успішність*» (рис. 5.7)

Для редагування параметрів сутності необхідно двічі підряд клацнути лівою кнопкою миші по створеній сутності або в контекстному меню сутності обрати Entity Properties.

У цьому вікні існує набір вкладок (рис. 5.13), на яких ідентифікується сутність та визначаються її різні властивості: ім'я та дефініція, примітки для зауважень чи бізнес-правил з організації діаграм, примітки для документування запитів, які можуть бути поставленими до сутності, примітки для прикладів даних сутності тощо.

Заповнення сутностей атрибутами. Заповнюємо створені та поіменовані сутності атрибутами, користуючись діалоговим вікном ***Attributes*** (рис. 5.14), обравши ***Attributes***. У цьому вікні за допомогою кнопок ***New, Rename*** та ***Delete*** можна здійснювати створення, перейменування та вилучення атрибутів сутності.

При створенні нового атрибута необхідно вказати, крім імені на логічному рівні ***Attribute name***, ім'я атрибута при переході на фізичний рівень – ***Column name***.

Опція ***Primary key*** вмикається для первинного ключа таблиці, крім того, для кожного атрибута необхідно задати домен

(***Domain***) та на вкладці ***Datatype*** конкретизувати домен, тобто обрати відповідний тип даних.

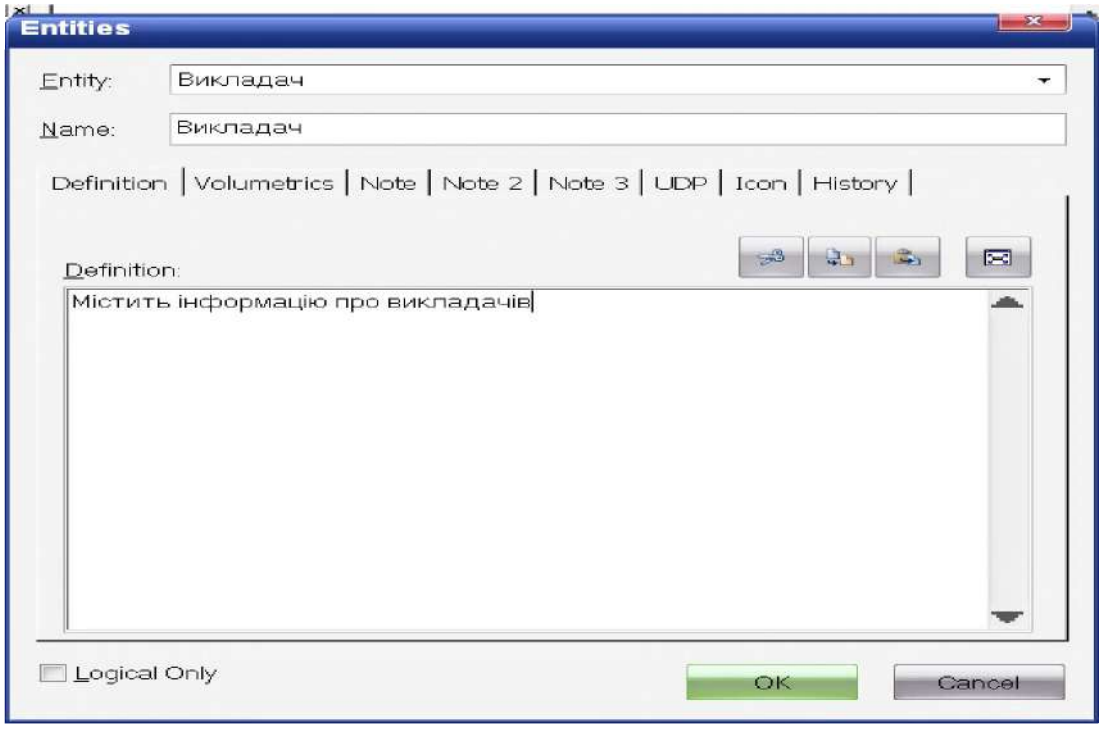


Рис. 5.13. Вікно властивостей сутності

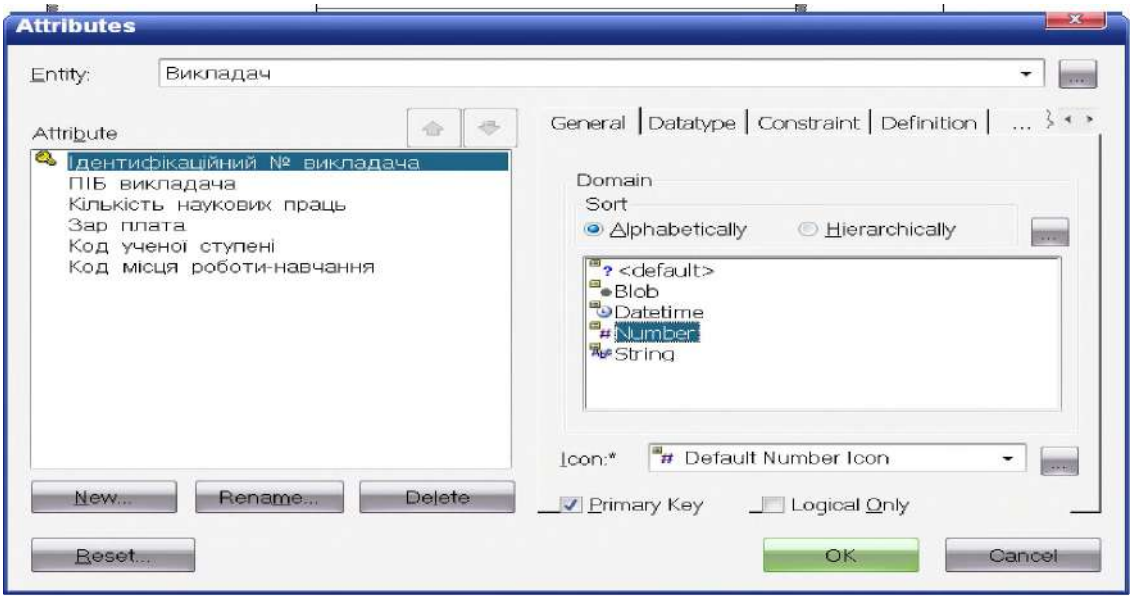


Рис. 5.14. Діалогове вікно властивостей атрибутів: вкладка *General*

Установка умов на значення атрибутів. AllFussion ERwin DM надає можливість при проектуванні БД встановити для атрибутів значення за умовчуванням (***Default values***) та умови на

значення (***Validation rules***), які потім при генерації будуть перенесені в БД цільової СУБД.

Для цього при описі атрибутів сутності у вікні ***Attributes*** на вкладці ***Datatype*** (рис. 5.15) для обраного атрибута слід з поля зі списком обрати вже створену умову перевірки його значення – ***Valid*** та створене значення за умовчуванням – ***Default***.

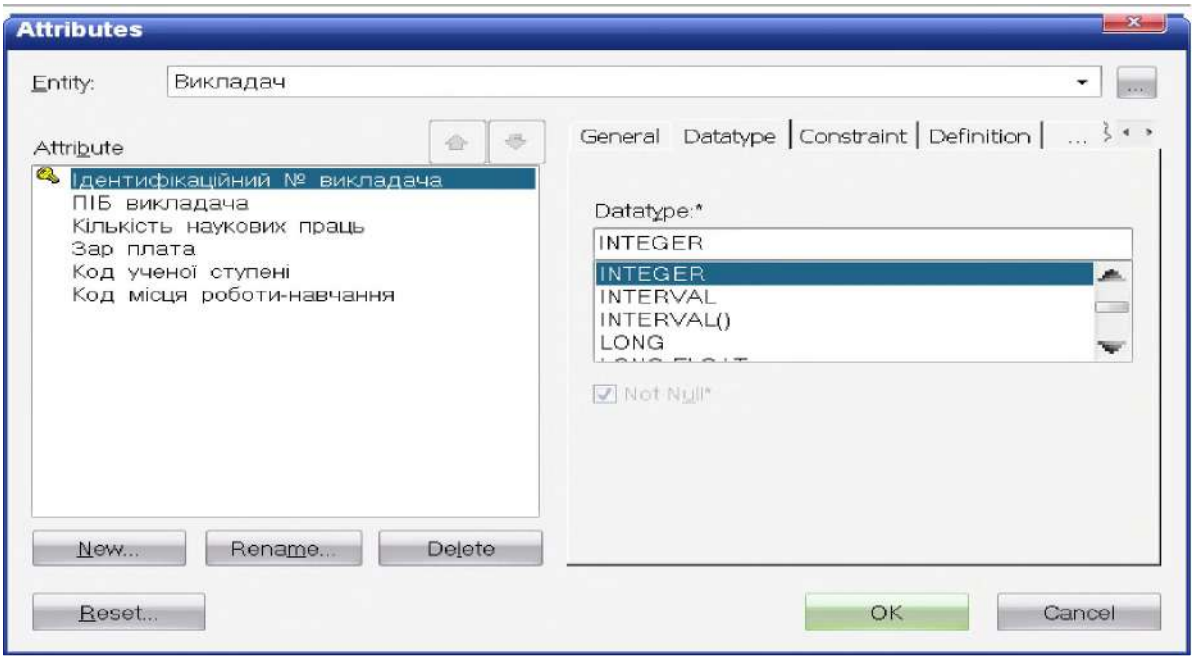


Рис. 5.15. Діалогове вікно властивостей атрибутів: вкладка *Datatype*

Отже, попередньо необхідно створити за допомогою меню ***Model/Default values*** **та** ***Model/Validation rules,*** відповідно, правила установки значення за умовчуванням та правила перевірки на значення.

Створення значення за умовчуванням. ***Для створення значення за умовчуванням*** слід у вікні ***Default values*** (рис. 5.16) натиснути кнопку ***New*** і ввести логічну та фізичну назви правила установки значення за умовчуванням. Безпосередньо значення за умовчуванням слід ввести у полі ***Defaul Value***.

Створення умови на значення. ***Для створення умови на значення*** слід у вікні ***Validation Rules*** (рис. 5.17) натиснути кнопку ***New*** та ввести логічну і фізичну назви правила установки значення за умовчуванням.

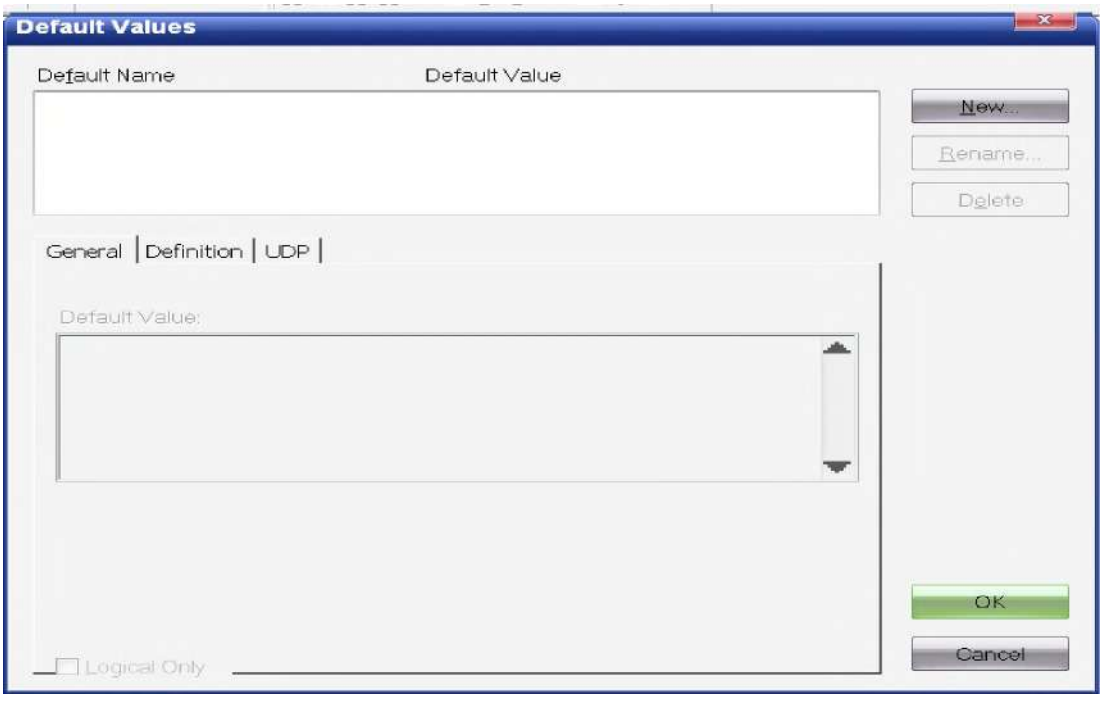


Рис. 5.16. Вікно редактора правил встановлення значень за умовчуванням

Безпосередньо значення за умовчуванням слід ввести на вкладці ***General*** за допомогою одного з трьох методів: 1) *User defined* – написання користувачем виразу для встановлення можливих значень згідно із синтаксисом обраної цільової СУБД; 2) *m/Max* – використання операторів *min* або/та *max*; 3) *Valid values list* – створення списку можливих значень.

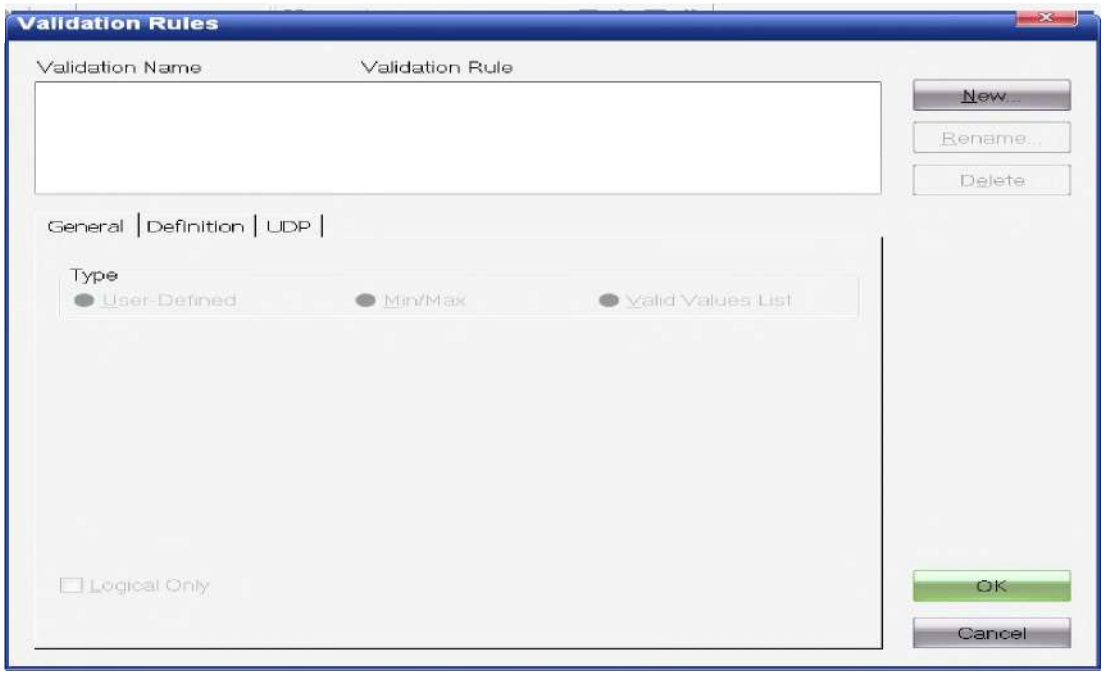


Рис. 5.17. Вікно *Validation rules*

У цих методах та вкладці ***General*** використовуються прапорці ***Not*** (логічне «перевертання» виразу на значення) та ***Quote*** (значення береться у лапки, якщо це передбачено синтаксисом обраної цільової СУБД).

Побудова зв'язків. Питання про тип зв'язку вирішується виходячи з аналізу семантики предметної області. Після вибору типу зв'язку необхідно клацнути курсором спочатку на таблиці – «власника зв'язку», а потім на «підлеглій» таблиці.

Для редагування параметрів зв'язку достатньо лише двічі клацнути лівою кнопкою миші на відповідному зв'язку, і тоді відкриється вікно ***Relationships***, натиснувши на вкладку ***Definition,*** перейдем у вікно опису зв'язку (рис. 5.18).

У вікні ***Relationships*** можна деталізувати різні характеристики зв'язку, зокрема, на вкладці ***General*** (рис. 5.19): 1) ім'я зв'язку (*Verb phrase*) в обох напрямах: власник → підлеглий та підлеглий → власник; 2) тип зв'язку (*Relationship type*) – ідентифікуючий чи неідентифікуючий; 3) обов'язковість (*Nulls*) для неідентифікуючого зв'язку; 4) потужність зв'язку (*Cardinality*) – відношення кількості екземплярів батьківської сутності до відповідної кількості екземплярів дочірньої сутності. За необхідності заповнюєте ім'я ролі на вкладці *Rolename*.

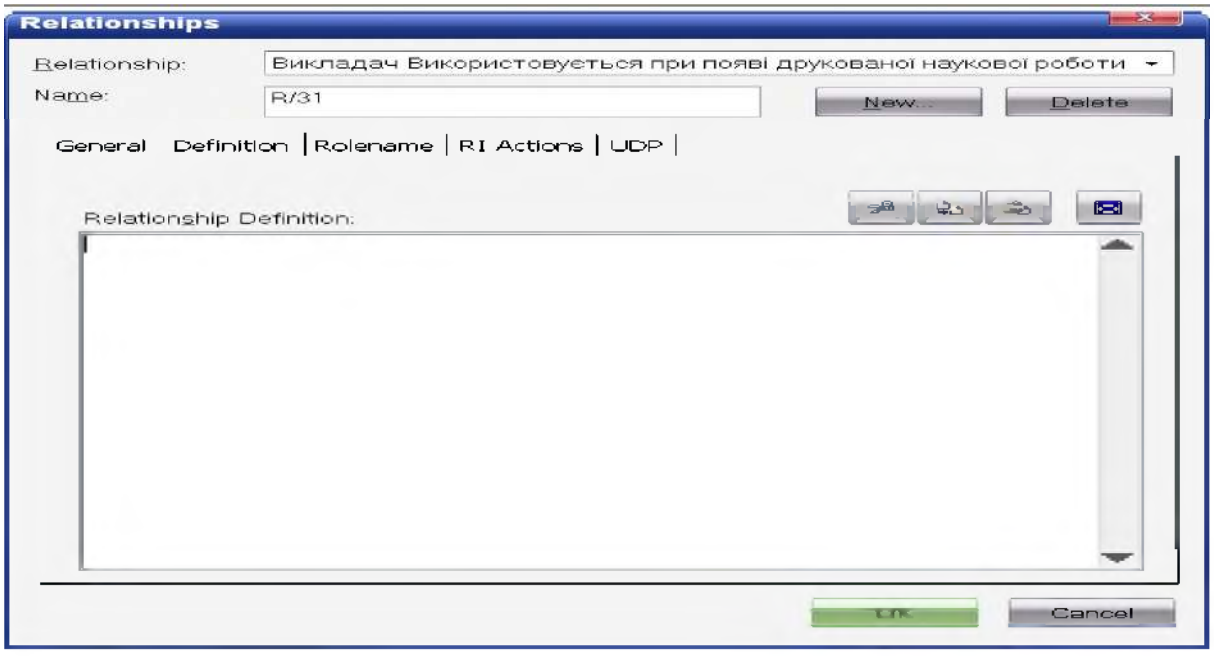


Рис. 5.18. Вікно опису звязку: вкладка *Definition*

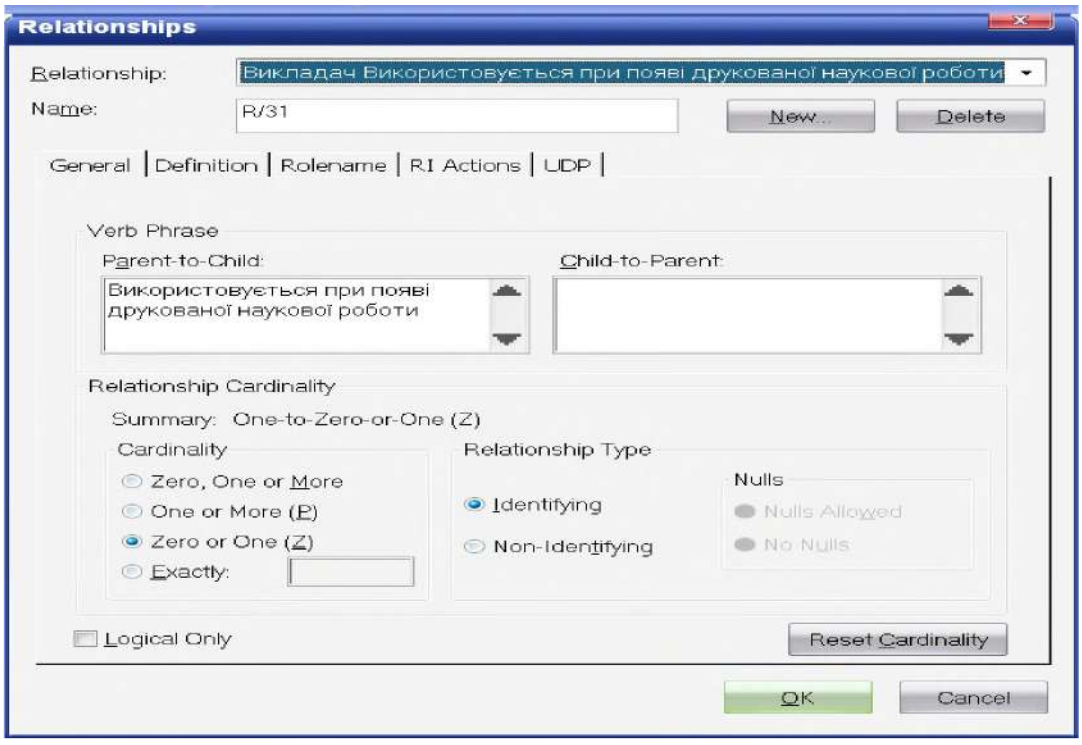


Рис. 5.19. Вікно властивостей зв'язку: вкладка *General*

На вкладці ***RI Actions*** (рис. 5.20) необхідно встановити бізнес-правила, що задають обмеження посилкової цілісності (***referential integrity – RI***) БД при виконанні операцій як для батьківської (***Parent***), так і для дочірньої (***Child***) сутностей:

- ✓ вилучення (***Delete***);
- ✓ вставки (***Insert***);
- ✓ заміни (***Update***).

AllFussion ERwin DM пропонує такі варіанти обробки цих подій:

- ✓ заборона операції (***Restrict***);
- ✓ каскадне виконання операції (***Cascade***);
- ✓ установка пустого значення (***Set null***);
- ✓ установка заданого значення за замовчуванням (***Set default***);
- ✓ відсутність перевірки (***None***).

Перехід до логічного і фізичного рівня проектування БД здійснюється за допомогою відповідного поля зі списком на панелі інструментів.

На рисунку 5.21 подано логічний рівень спроектованої БД «Успішність».

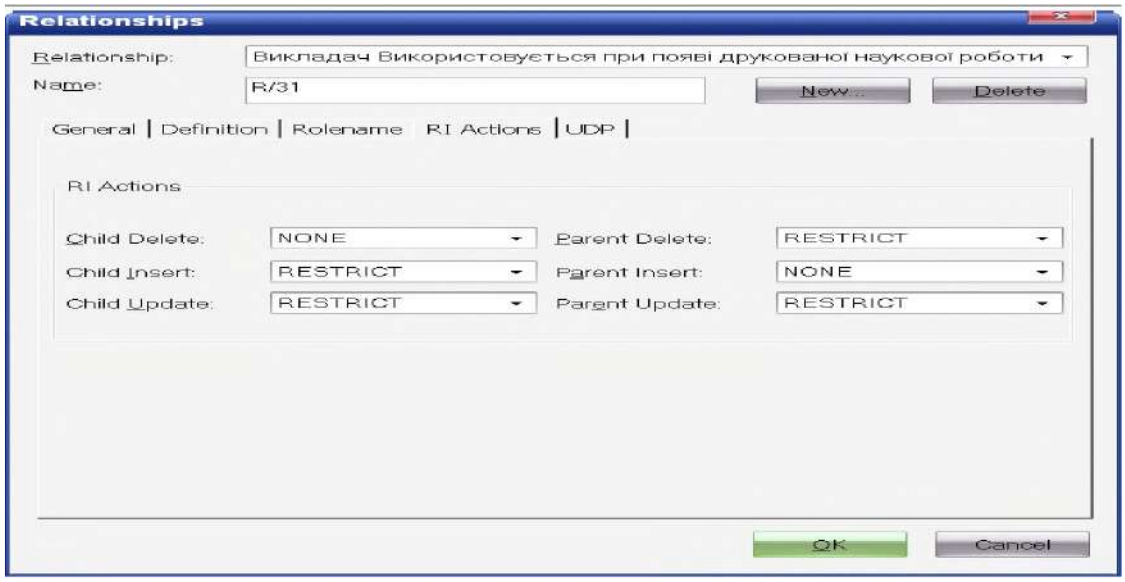


Рис. 5.20. Вікно редактора правил встановлення умов на значення: вкладка *RI Actions*

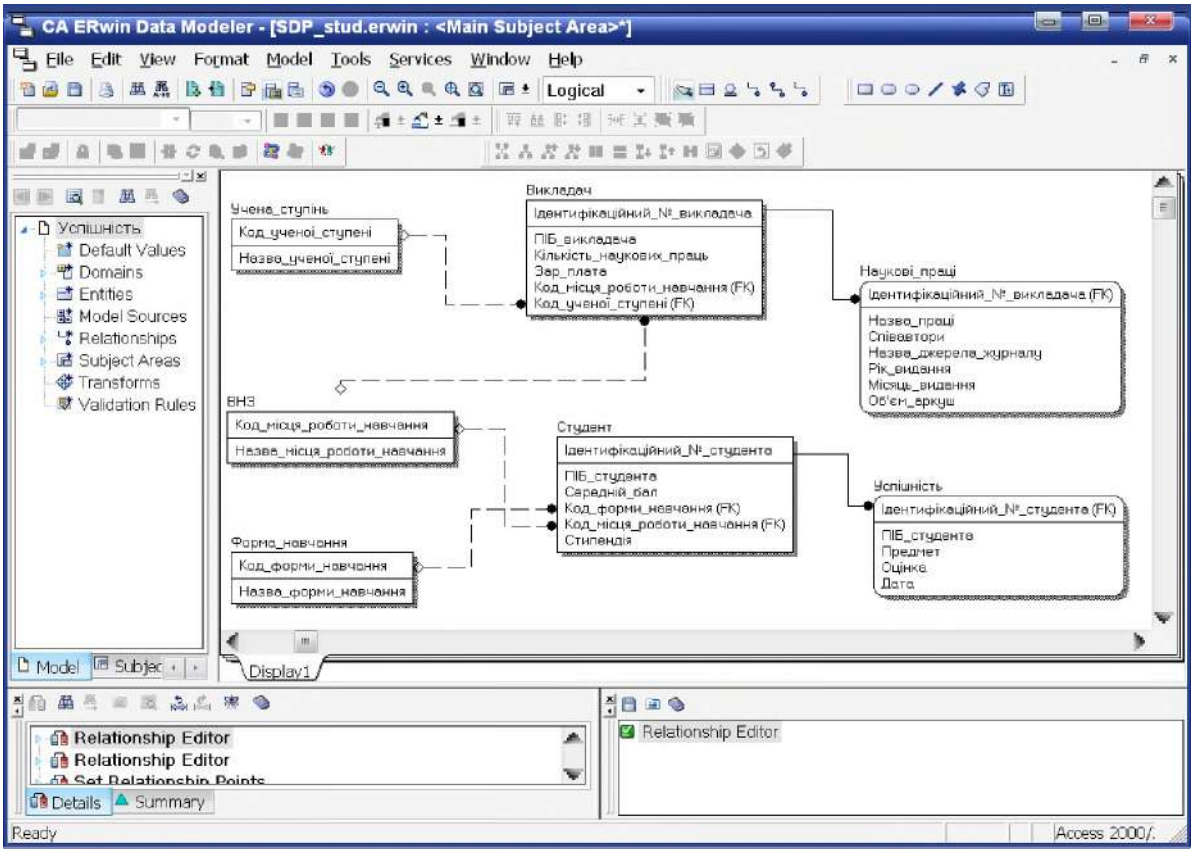


Рис. 5.21. Логічний рівень спроектованої БД

На рисунку 5.22 наведений також фізичний рівень спроектованої БД.

Створення індексів. Для підвищення продуктивності БД використовують індекси, надається можливість встановити два

типи індексів: 1) унікальні індекси ***Alternate key***); 2) неунікальні індекси (***Inversion entries***).

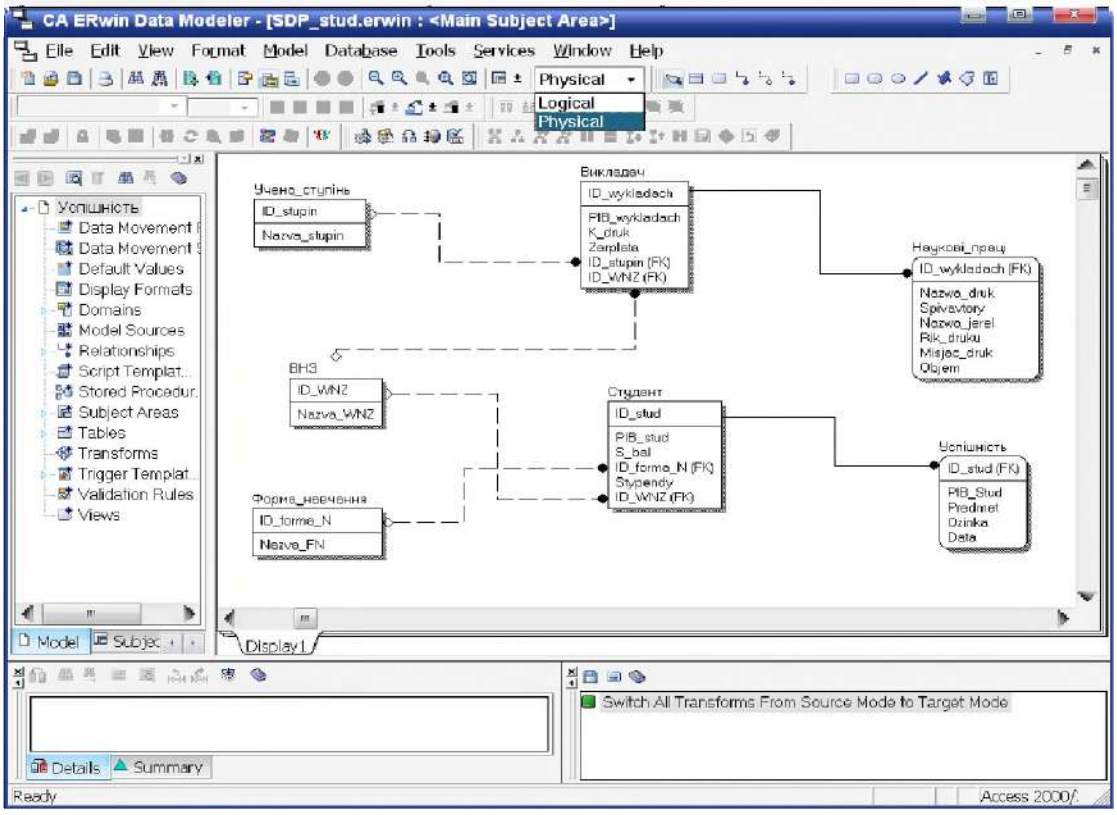


Рис. 5.22. Фізичний рівень спроектованої БД

Необхідно обрати ті атрибути з проектованої БД, які у практичній роботі зі створеною БД можуть потребувати встановлення індексів. Для цього з контексного меню таблиці обираємо ***Indexes*** та працюємо у ***Access indexes*** вікні (рис. 5.23).

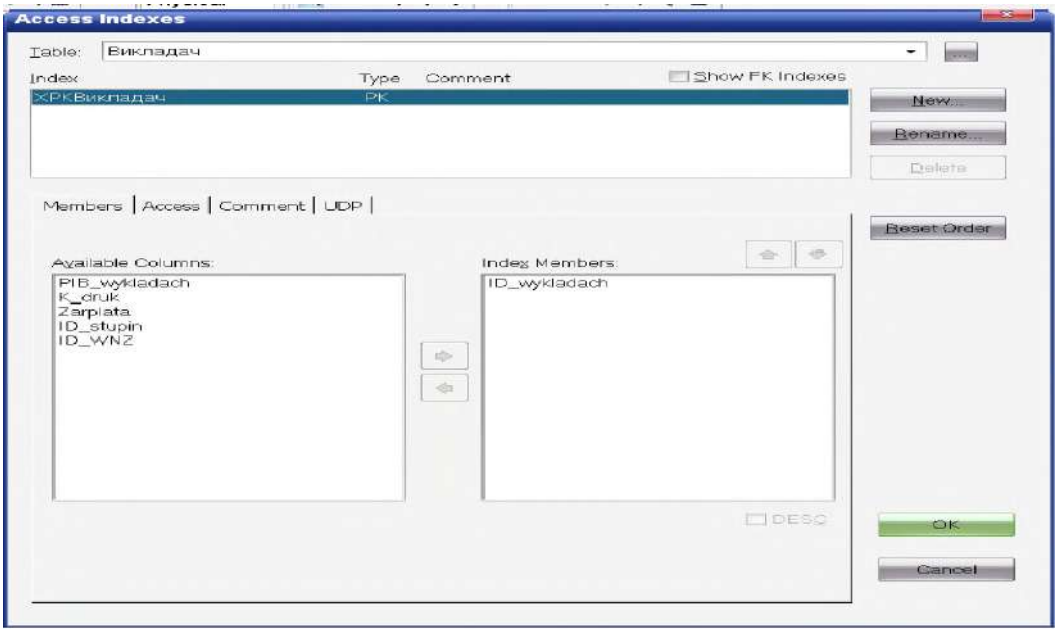


Рис. 5.23. Вікно індексів моделі

Зокрема, на вкладці ***Members***:

- кнопки ***New, Rename*** та ***Delete*** використовуються відповідно для створення, зміни ім'я та вилучення індексу;
- опція ***DESC*** використовується для встановлення сортування значення індексу за спаданням;
- при створенні нового індексу необхідно задати назву індексу на логічному (***Key group***) та фізичному (***Index***) рівнях (за умовчуванням вони встановлюються автоматично) (рис. 5.24);
- якщо опція ***Unique*** увімкнена –> створюється унікальний індекс, і навпаки –> не унікальний індекс.

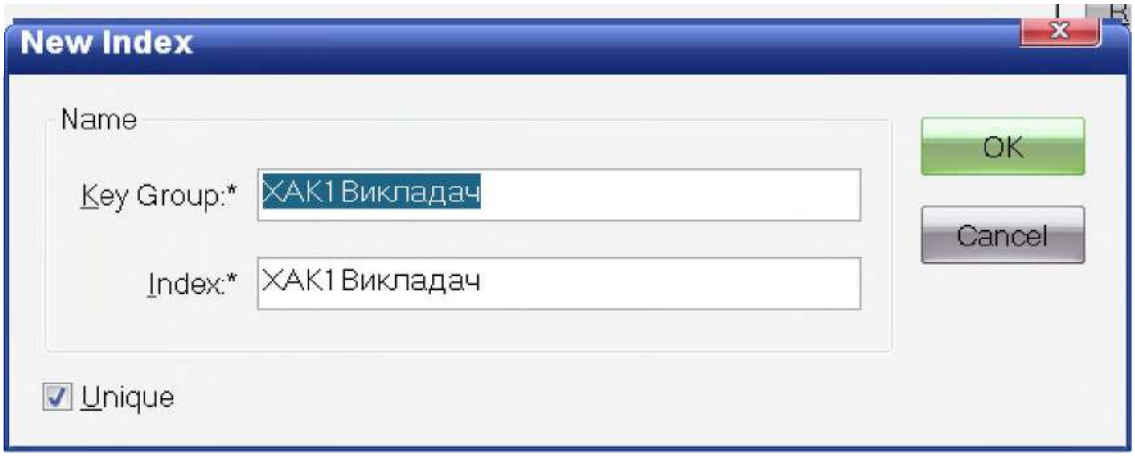


Рис. 5.24. Вікно створення нового індексу

Наступна вкладка відповідає обраній СУБД, цього разу це – Access (рис. 5.25).

Можна визначити такі характеристики індексу:

1. *Unique* – унікальність/неунікальність значення індексу.
2. *Clustered* – кластеризований індекс, коли записи фізично розташовані в індексованому порядку, що підвищує швидкість роботи БД.
3. *Generate* – генерація/негенерація індексу в Access.
4. *Physical only* – індекс буде задіяний тільки на фізичному рівні.
5. *Ignore nulls* – забороняє наявність в індексованому полі пустих значень.

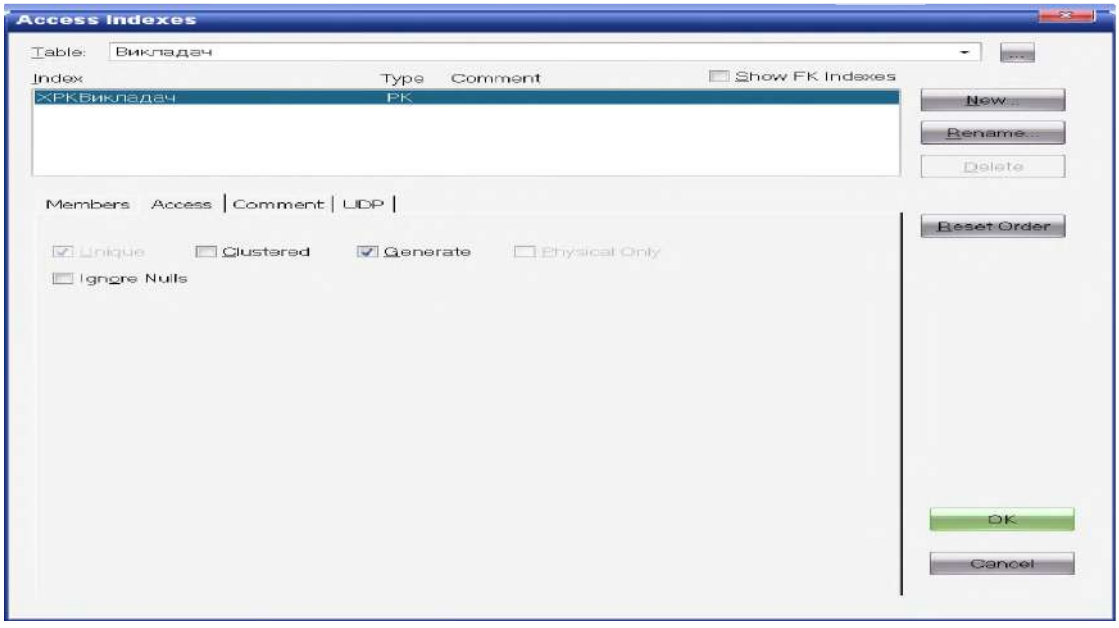


Рис. 5.25. Вкладка Access вікна індексів

Генерація створеного проекту БД у середовище СУБД Access. AllFussion ERwin DM надає можливості для вибору різноманітних цільових СУБД, у яких реалізовувалася б створена модель БД.

Для цього слід скористатися пунктом меню ***Database/Choose database***.

У вікні ***Target Server*** (рис. 5.26) користувач може обрати для реалізації чи ***SQL***-подібну серверну ***СУБД***, чи настільну ***СУБД*** та відповідні її параметри, зокрема тип даних за умовчуванням.

Використовується механізм ***ODBC*** для передачі БД у СУБД, цього разу – ***MS Access***. Тому треба відповідним чином налаштувати ***ODBC***.

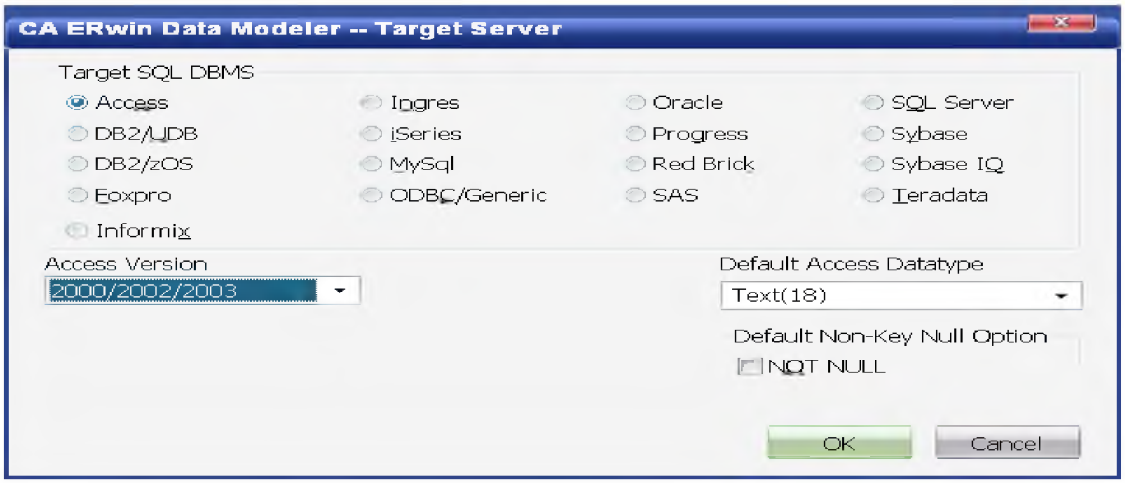


Рис. 5.26. Вікно вибору СУБД

Спочатку необхідно створити пустий файл ***MS Access db1.mdb*** у каталозі ***Q:\PIB***, де ***Q*** – *доступний для розробника диск,* ***PIB*** *– розробник.*

Далі – відкрити ***Пуск/Настройка/Панель управления*** та запустити піктограму ***Источники данных ODBC***.

У вікні ***Администратор источников данных ODBC*** на вкладці ***Пользовательский DNS*** потрібно обрати джерело даних ***MS Access Database*** та натиснути кнопку ***Настройка***.

У вікні, що відкрилося, необхідно натиснути кнопку ***Выбрать*** і вказати на створений файл ***Q:\PIB\db1.mdb*** та натиснути ***OK***.

У ***CASE-засобі ERwin Data Modeller*** для безпосередньої генерації БД маємо обрати пункт меню ***Tools/Forward engineer/Schema generation***.

У вікні, що відкрилося (рис. 5.27), на вкладці ***Options*** можна обрати, які об'єкти слід переносити, а на вкладці ***Summary*** – згрупований перелік об'єктів, які будуть згенеровані. Далі необхідно натиснути кнопку ***Generate***.

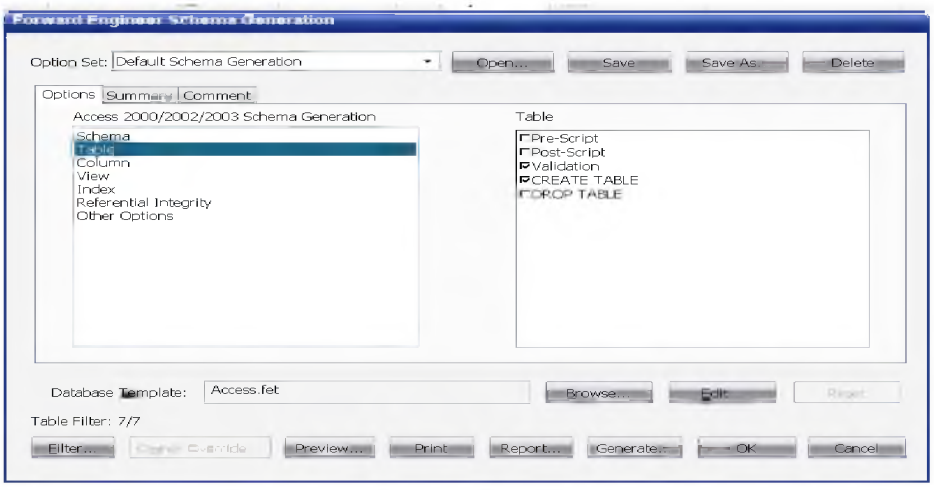


Рис. 5.27. Діалогове вікно генерації БД

У вікні, що відкрилося, необхідно вказати параметри, як це подано на рис. 5.28.

Процес генерації коду БД відтворюватиметься у вікні ***Generate Database Shema*** (рис. 5.29). При появі помилки генерації процес зупиниться, після чого потрібно зняти прапорець ***Stop if Failure*** та натиснути кнопку ***Continue***. Для перегляду результатів необхідно в середовищі MS Access відкрити файл ***Q:\PIB\db1.mdb.***

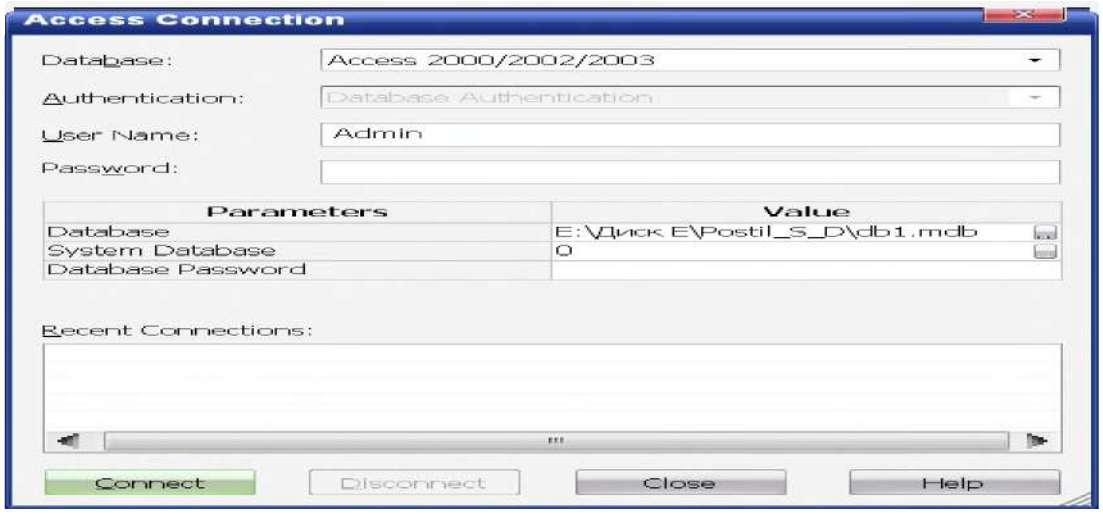


Рис. 5.28. Вікно встановлення параметрів з'єднання з СУБД

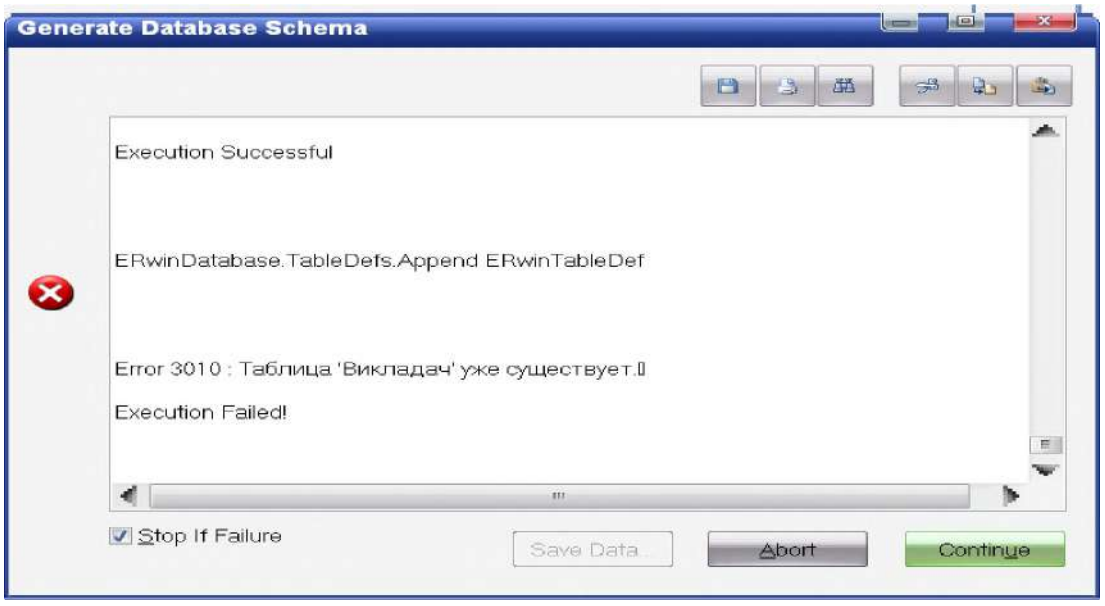


Рис. 5.29. Вікно відтворення процесу генерації БД

Перед повторною генерацією важливо фізично вилучити файл ***Q:\PIB\db1.mdb*** і створити знову пустий файл MS Access ***db1.mdb*** у каталозі ***Q:\PIB***.

На рис. 5.30 наведено схему даних згенерованої в ***CASE-засобі ERwin Data Modeller*** та переданої в MS Access бази даних.

Прогнозний підрахунок обсягу БД. ***CASE-засіб ERwin Data Modeller*** надає змогу підрахувати можливий розмір спроектованої БД у обраній СУБД. Для цього слід скористатися меню ***Tools/Volumetrics.***

На вкладці ***Settings*** (рис. 5.31):

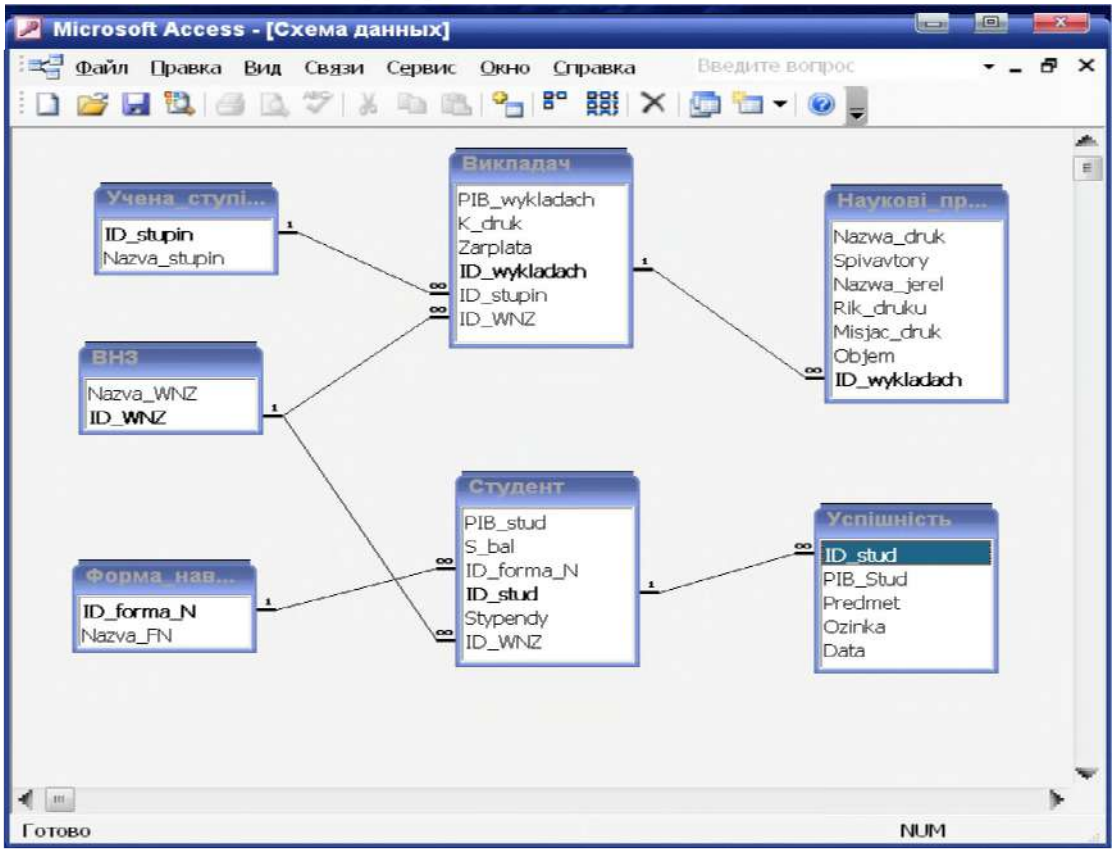


Рис. 5.30. Схема даних, згенерованої в *CASE-засобі ERwin Data Modeller* та переданої в MS Access БД

1. У групі полів ***Sizing Estimates*** для активної таблиці зі списку відтворюються: середній розмір одного запису (*Average Row Size*), початковий розмір таблиці (*Initial Table Size*) та початковий розмір індексів (*Initial size of index*).

2. У групі полів ***Table row counts*** можна задати такі параметри для прогнозування обсягу БД: *початкова кількість записів; максимальна кількість записів; зростання кількості записів за місяць*.

На вкладці ***Report*** (рис. 5.32) можна побачити результати підрахунку:

1. У групі полів ***Time*** вказується часовий горизонт прогнозування обсягу БД: початковий обсяг (*Initial*) чи прогнозований через вказану кількість місяців (*Projection*).

2. У групі полів ***Report*** вказується розріз, в якому подано дані підрахунку обсягу БД: фізичні об'єкти, об'єкти БД, таблиці.

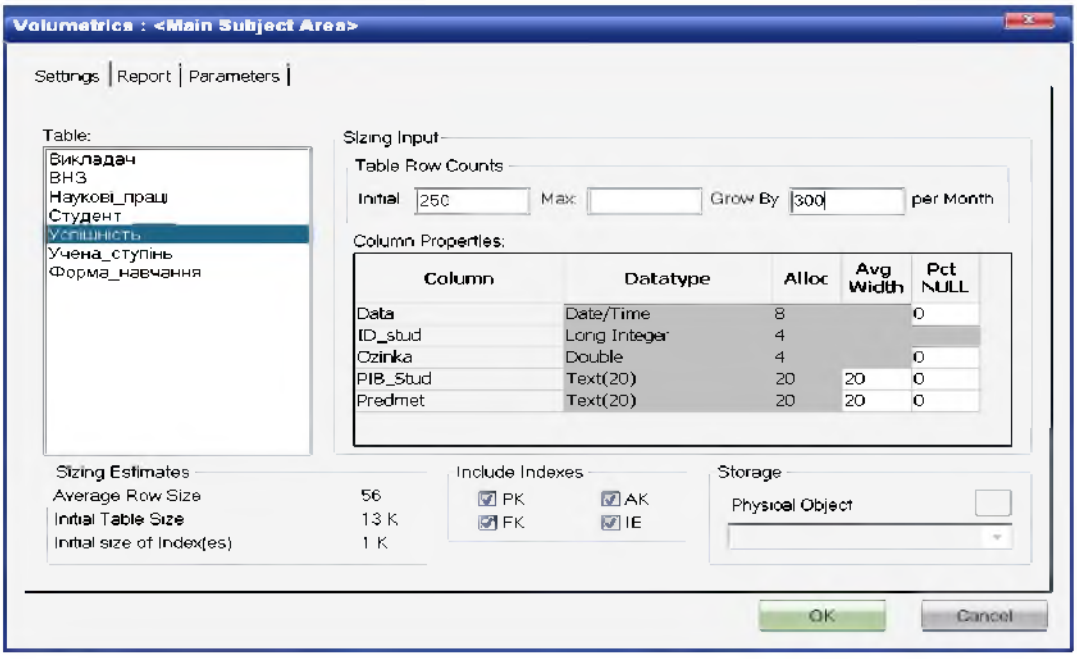


Рис. 5.31. Вікно прогнозного підрахунку обсягу БД: вкладка Settings

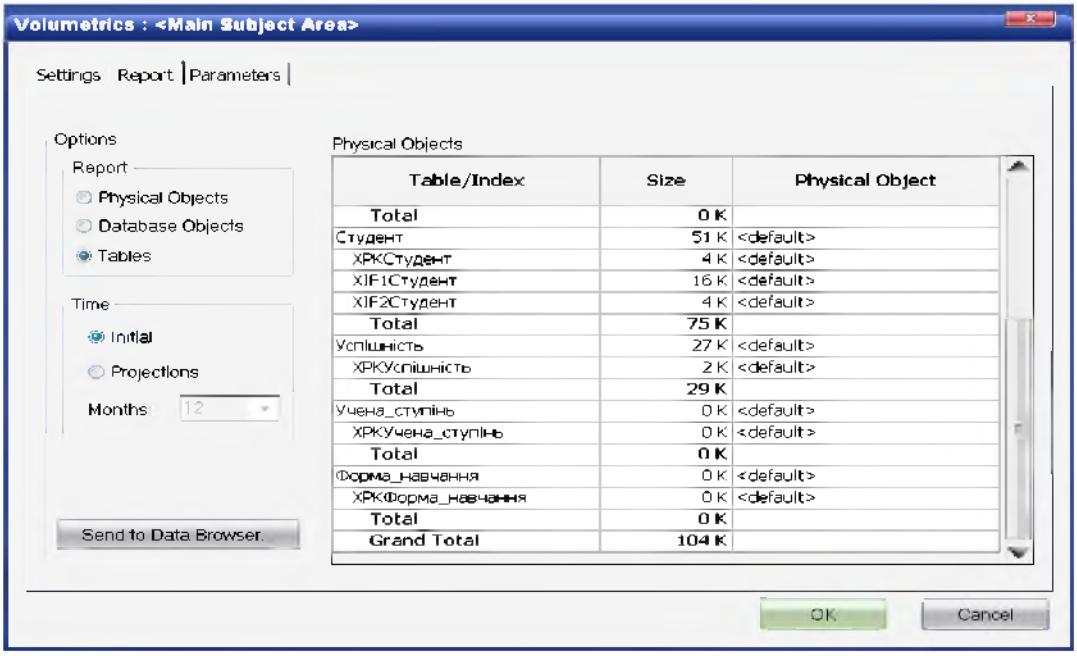


Рис. 5.32. Вікно прогнозного підрахунку обсягу БД: вкладка *Report*

Натиснувши кнопку ***Send to data browser***, ви передасте результат підрахунку обсягу БД до браузера даних, звідки його зручно роздрукувати (*знову задавши часовий горизонт прогнозування та обравши один із трьох варіантів відтворення*). **Друк звіту.** За допомогою майстра генерації звітів, що викликається з меню ***Tools/Report template Builder***, побудувати і роздрукувати звіт.

У першому вікні діалогу ***Report template*** можна обрати для роботи або вже сконфігурований та збережений шаблон звіту чи за допомогою кнопки *New* створити новий, вказавши вихідний формат (***Output type)***: HTML, RTF, TXT.

Далі у вікні конфігурування *ЗВІТУ* ***Report template builder*** (рис. 5.33) слід натиснути на кнопку для встановлення глобальних параметрів звіту: зокрема, формату вихідного файла та наявності змісту звіту, назви звіту та його оформлення, верхнього колонтитула, нижнього колонтитула, опису звіту та його розташування.

Обов'язково у колонтитулі звіту має бути внесено ПІП студента, що виконує лабораторну роботу.

Для конфігурування вмісту звіту слід обрати з лівої частини вікна (***Available sections***) з доступних для документування розділів необхідні та натиснути на кнопку для внесення їх у звіт.

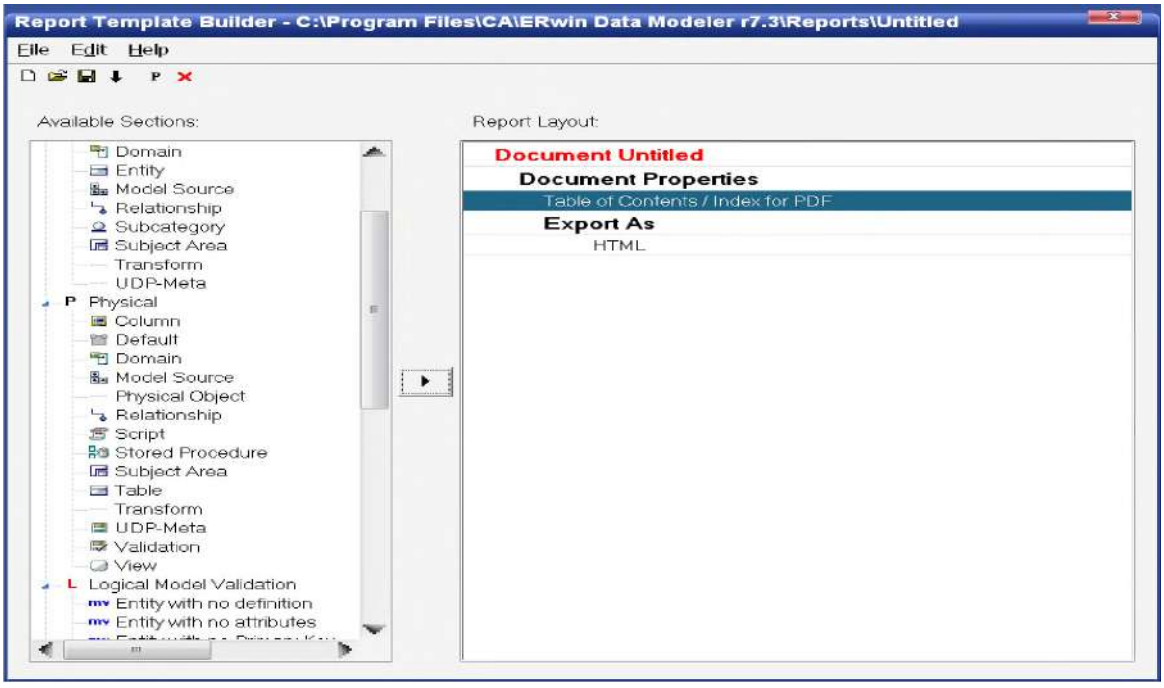


Рис. 5.33. Вікно конфігурування звіту

Для усунення обраного розділу слід скористатися відповідною кнопкою. Далі у правій частині вікна (***Report layout***) слід двічі клацнути на внесеній групі параметрів для безпосереднього виділення у дереві властивостей тип параметрів, що відображатимуться у звіті. Для виконання цього завдання лабораторної роботи потрібно у лівій частині вікна ***Report Template Builder***

обрати у групі ***Physical*** об'єкт ***Table*** та внести його у зміст звіту. Далі необхідно виділити в обраному розділі ***Table*** всі основні параметри документування (рис. 5.34).

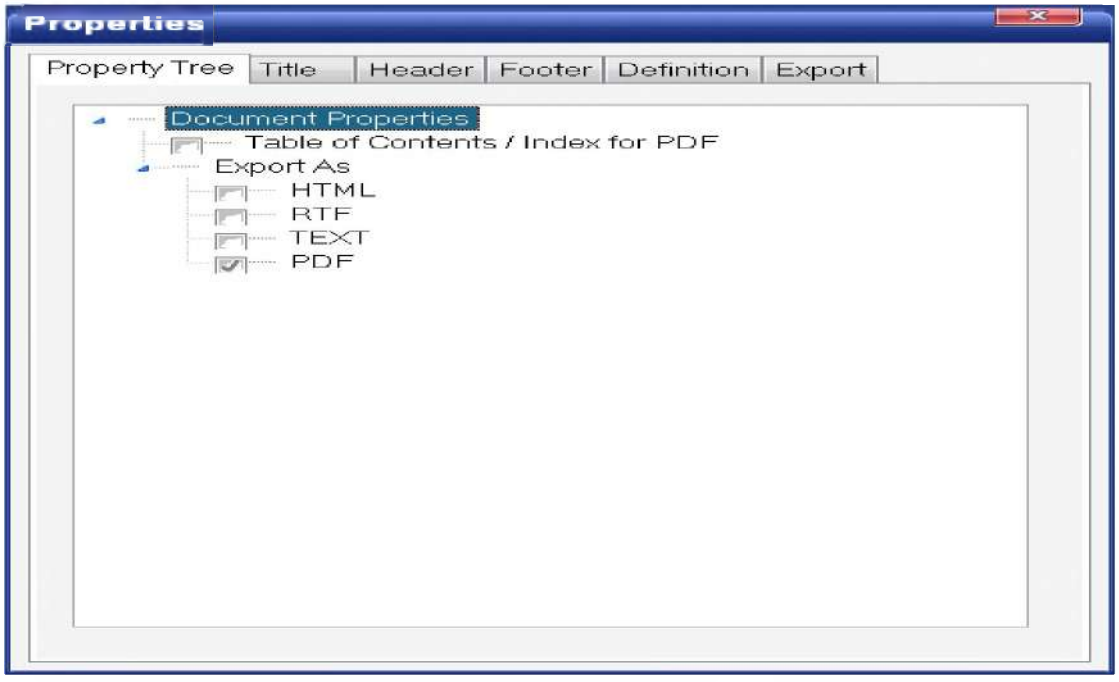


Рис. 5.34. Діалогове вікно вибору розділів для документування

Тут же, на вкладці ***Section***, можна задати оформлення обраного розділу звіту. Для запуску звіту на генерацію необхідно натиснути відповідну кнопку.

У звіт про виконану лабораторну роботу обов'язково включається роздруківка:

1. Моделі бази даних на логічному та фізичному рівнях за допомогою меню ***File/Print Setup та File/Print*** (*обов'язково із наявним текстовим блоком, в якому вказане ПІП студента та номер групи)*, які побудовані студентом.
2. Результатів підрахунку обсягу БД.
3. Схеми даних згенерованого файла БД в Access.
4. Звіту, отриманого засобами документування баз даних у CASE-засобі ERwin Data Modeller.

5.3. Лабораторна робота 9

Тема роботи. Методологія IDEF1X зі створення проекту БД за допомогою CASE-засобу AllFusion ERwin Data Modeler

Мета роботи:

- ознайомитись з особливостями методології IDEF1X зі створення БД;
- вивчити концепцію та функціональні можливості сучасних CASE-засобів для проектування баз даних за допомогою CASE-засобу AllFussion ERwin Data Modeller (DM) пакета AllFussion Modeling Suite CA

Завдання

1. Освоїти технологію проектування реляційної моделі на основі принципів нормальних форм, що представлена в п. 5.2.

2. Вдосконалити концептуальну модель бази даних «Індивідуального проекту» на основі методології ERD та принципів нормальних форм.

3. На прикладі концептуальної моделі БД «Успішність» (рис. 5.7) ознайомитись з особливостями проектування баз даних у середовищі CASE-засобу AllFussion ERwin DM шляхом виконання наступного:

- запуск CASE-засобу AllFussion ERwin DM;
- створення нової моделі бази даних;
- створення сутностей та робота з моделлю;
- заповнення сутностей атрибутами;
- побудова зв'язків;
- встановлення параметрів таблиць бази даних;
- генерація моделі СУБД;
- друк звіту.

Хід роботи

1. Проведення аналізу індивідуального об'єкта дослідження.
2. Виконання роботи до завдання-прототипу.
3. Виконати всі пункти, які передбачені у Завданні.
4. Враховуючи індивідуальні особливості об'єкта дослідження допускається розширення і поглиблення робіт «Індивідуального проекту» (може ускладнитись процес отримання зворот-

ної інформації про виконання індивідуальних завдань та їхнього корегування).

Оформлення звіту

Звіт про виконану роботу оформляється згідно із «Загальних рекомендаціями до виконання лабораторних робіт та оформлення звіту» (додаток А: А. 1, А. 2).

Контрольні запитання

1. Як здійснюється запуск засобу AllFussion ERwin DM?
2. Як здійснюється створення нової моделі бази даних або відкриття раніше створеної моделі?
3. Як здійснюється створення сутностей та робота з моделлю?
4. Як здійснюється заповнення сутностей атрибутами?
5. Як здійснюється побудова зв'язків?
6. Як здійснюється встановлення параметрів таблиць бази даних та друк звіту?

5.4. Лабораторна робота 10

Тема. Впровадження методології IDEF1X створенням БД «Наскрізний індивідуальний проект» за допомогою CASE-засобу AllFusion ERwin Data Modeler

Мета роботи:

- закріпити навики проектування баз даних за допомогою CASE-засобу ERwin Data Modeller (DM) пакета AllFussion Modeling Suite;
- згенерувати створений проекту БД «Наскрізного індивідуального проекту» в середовище необхідної СУБД, наприклад в MS Access;
- здійснити прогнозний підрахунок обсягу БД.

Завдання

1. На основі концептуальної моделі бази даних «Наскрізного індивідуального проекту» розробити проект баз даних у середовищі CASE-засобу AllFussion ERwin DM шляхом виконання наступного:
   - запуск пакета;
   - створення нової моделі бази даних «Наскрізний індивідуальний проект»;

- створити таблиці відповідно до концептуальної моделі БД «Наскрізного індивідуального проекту»;
- заповнити таблиці атрибутами та описати їх властивості;
- побудувати зв'язки;
- встановити параметри таблиць бази даних;
- згенерувати БД у вигляді моделі СУБД;
- формування звіту з побудованої студентом моделі прототипу бази даних на фізичному рівнях у середовищі CASE-засобу AllFussion ERwin DM за допомогою меню ***File/Print Setup та File/Print*** (*обов'язково з наявним текстовим блоком, у якому вказане ПІП студента та назва групи*);
- відповідь на контрольні питання.

Хід роботи

1. Проведення аналізу індивідуального об'єкта дослідження.
2. Виконання роботи для індивідуального об'єкта дослідження здійснюється відповідно до завдання-прототипу.
3. Виконати всі пункти, які передбачені у Завданні.
4. Враховуючи індивідуальні особливості об'єкта дослідження, допускається розширення і поглиблення робіт «Індивідуальний проект» (може ускладнитись процес отримання зворотної інформації про виконання індивідуальних завдань та їхнього корегування).

Оформлення звіту

Звіт про виконану роботу оформляється згідно із «Загальними рекомендаціями до виконання лабораторних робіт та оформлення звіту» (додаток А: А. 1, А. 2).

Контрольні запитання

1. Охарактеризуйте функціональні можливості CASE-засобу ERwin Data Modeller.
2. Яку методологію інформаційного моделювання підтримує CASE-засіб ERwin Data Modeller?
3. Які типи зв'язків між даними підтримує CASE-засіб ERwin Data Modeller.
4. У чому полягає відмінність ідентифікуючого зв'язку від неідентифікуючого?

5. Викладіть порядок інформаційного моделювання з використанням CASE-засобу ERwin Data Modeller.

Завдання для моделювання

Для досліджуваного об'єкта розробити в специфікації IDEF1X проект БД, який дозволить отримати відомість:

ВІДОМІСТЬ 1

розрахунку відрядної індивідуальної заробітної плати

за цехом _____ за _____ місяць 201__ р.

код та назва цеху

Таб. номер	Прізвище	Код деталі	Розряд	Професія		Кількість виготовлених деталей	Розцінка за деталь	Відроблений час	Сума заробітної плати
				Код	Назва				

ВІДОМІСТЬ 2

потреби в матеріалах на річну виробничу програму

за підприємством на 201__ р.

Код матеріалу	Назва матеріалу	Одиниці вимірювання	Код виробу	Назва виробу	Програма випуску виробів на рік	Норми витрат матеріалу на виріб	Потреба матеріалу на річну програму

ВІДОМІСТЬ 3

обліку руху матеріалів на складі ___ за ___ місяць

код та назва складу

Матеріал			Залишок вхідний		Надходження		Видатки		Залишок вихідний	
Код	Назва	Одиниця виміру	Кількість	Вартість	Кількість	Вартість	Кількість	Вартість	Кількість	Вартість

ВІДОМІСТЬ 4

обліку якості готової продукції за цехом ___ за __ місяць

код та назва цеху

Номер дільниці	Деталь		Операція		Вид браку		Причина (винуватець)		Кількість деталей	
	Код	Назва	Код	Назва	Код	Назва	Код	Назва	Придатних	Бракованих

ВІДОМІСТЬ 5

відвантаження готової продукції за _____ місяць

Код покупця	Назва покупця	Код продукції	Назва продукції	№ угоди на відвантаження	Кількість за угодою	Фактично відвантажено	Відхилення

5.5. Методологія вимірного моделювання сховищ даних

Використання інформаційних технологій на основі сховищ даних (СД) припускає задачний підхід у його організації. СД створюється для вирішення конкретних, чітко визначених задач аналізу даних, які слід вирішувати для досягнення цілей бізнес-процесів. СД визначається як погоджене інтегроване представлення даних з різних джерел за їхнім типом, розмірністю і змістовним описом від бізнес-процесів.

У вимірному моделюванні прийнятий стандарт моделі, названий схемою «зірка», що забезпечує високу швидкість виконання запиту за допомогою денормалізації та поділу даних. Схема «зірка» будується так, щоб забезпечити найвищу продуктивність при виконанні одного найважливішого запиту або для групи схожих запитів. Схема «зірка» звичайно містить одну велику таблицю, яка називається таблицею змінних (або показників) фактів, поміщену в центр, і оточуючі її менші таблиці, названі таблицями вимірів фактів, з'єднаних у вигляді зірки радіальними зв'язками. У цих зв'язках таблиці вимірів є батьківськими, таблиця змінних (або показників) – дочірньою.

Схема «зірка» може мати також консольні таблиці, приєднаних до таблиці вимірів і призначених для їх нормалізації.

Консольні таблиці є батьківськими, таблиці вимірів – дочірніми. Модель сховища, в якій за рахунок введення консольних таблиць нормалізовані всі таблиці вимірів, називається «сніжинкою».

Створення СД за схемою типу «зірка» полягає в тому, що створюють таблиці для кожного виміру, а всі змінні (або показники) уміщують в одну таблицю, яка індексується множинним ключем, складеним із ключів окремих вимірів. Кожний промінь схеми «зірка» задає, за термінологією Кодда, напрям консолідації даних за відповідним виміром.

У технології багатомірного моделювання вимір – це аспект, у розрізі якого можна одержувати, фільтрувати, групувати та відображати інформацію про факти. Типові виміри, що зустрічаються практично в будь-якій моделі: клієнт, продукт, час, географія, співробітник тощо. Виміри, як правило, мають багаторівневу ієрархічну структуру. Наприклад, розмірність «Час» може мати таку структуру: «Рік», «Квартал», «Місяць», «Тиждень», «День».

Таблиця змінних (або показників) – центральна в схемі «зірка». Вона може складатися з великої кількості рядків і містити підсумовуючі або фактичні дані, які допоможуть відповісти на запитання, що виникають. У ній акумульовані дані, що могли б бути розосереджені за багатьма таблицями традиційних реляційних БД.

Змінні (або показники) фактів можуть бути адитивними, напівадитивними і неадитивними.

Адитивність визначає можливість підсумовування змінних уздовж визначеного виміру. Адитивні змінні можна підсумовувати і групувати уздовж усіх вимірів на будь-яких рівнях ієрархії.

Напівадитивні змінні фактів, тобто ті, які можна підсумовувати уздовж визначених вимірів і не можна – уздовж інших. Прикладом може слугувати залишок грошей на рахунку (або залишок товару на складі). Цю величину не можна підсумовувати уздовж виміру «Час». Однак сума залишків на рахунках уздовж даного виміру має сенс для аналізу.

Неадитивні змінні фактів взагалі не можна підсумовувати. Приклад неадитивного факту – відношення (наприклад, виражене у відсотках). Фахівці рекомендують моделювати неадитивні фак-

ти таким чином, щоб зробити їх адитивними. Наприклад, представити відсоток складовими його величинами.

Між таблицею змінних (або показників) і таблицями вимірів встановлені ідентифікуючі зв'язки, при цьому первинні ключі таблиці вимірів мігрують у таблицю фактів як зовнішні ключі. У вимірній моделі напрями зв'язків явно не показуються – вони визначаються типом таблиць. Первинний ключ таблиці змінних (або показників) цілком складається з первинних ключів усіх таблиць вимірів. Таблиці вимірів мають меншу кількість рядків, ніж таблиці фактів і містять описову інформацію. Ці таблиці надають змогу користувачеві швидко переходити від таблиці змінних (або показників) до додаткової інформації.

Таблиця змінних (або показників) фактів є основною таблицею СД. Як правило, вона містить відомості про об'єкти або події, сукупність яких буде надалі аналізуватися. Звичайно, говорять про чотири типи фактів, що зустрічаються найчастіше. До них належать:

✓ факти, пов'язані з транзакціями. Вони базуються на окремих подіях (типовими прикладами яких є телефонний дзвінок або зняття грошей з рахунку за допомогою банкомата);

✓ факти, пов'язані з «моментальними знімками». Базуються на стані об'єкта (наприклад, банківського рахунку) у певні моменти часу, наприклад, на кінець дня або місяця. Типовими прикладами таких фактів є обсяг продажу за день або денний виторг;

✓ факти, пов'язані з елементами документа. Базуються на тому або іншому документі (наприклад, рахунку за товар чи послуги) і містять докладну інформацію про елементи цього документа (наприклад, кількість, ціна, відсоток знижки);

✓ факти, пов'язані з подіями або станом об'єкта. Представляють виникнення події без подробиць про неї (наприклад, просто факт продажу або факт відсутності такого без будь-яких подробиць).

Створення вимірної моделі СД. Вимірна модель СД може бути реалізована у процесі виконання наступного сценарію дій.

1. Визначення цілей побудови СД.

2. Визначення вимог користувачів та формулювання бізнес-запитів.

3. Побудова таблиці взаємозв'язку змінних та вимірів.

4. Виявлення запитів, що описуються однаковими вимірами, і формування початкового набору фактів.

5. Аналіз змінних на адитивність, ступеня деталізації фактів та сумісності фактів за часовими вимірами.

6. Уточнення структур початкового набору фактів.

7. Дослідження фактів на можливість їх об'єднання.

8. Формування кінцевого набору фактів.

9. Проектування таблиць вимірів.

10. Об'єднання таблиці змінних фактів і таблиць вимірів фактів у вимірну модель СД.

Перш ніж створити СД за схемою типу «зірка», необхідно проаналізувати бізнес-правила предметної області з метою з'ясування центрального питання, відповідь на яке найважливіше.

Для визначення даних, які необхідно зберігати в СД, передусім необхідно провести збір вимог кінцевих користувачів.

Вимоги до СД кінцевих користувачів можна умовно розбити на дві групи: вимоги, орієнтовані на процес, та інформаційно орієнтовані вимоги.

Вимоги, орієнтовані на процес, пов'язані з процесами обробки даних при виконанні бізнес-аналізу. При вивченні цих вимог з'ясовуються основні види перетворень, групувань та обчислень, які необхідно виконувати з даними при обчисленні агрегованих даних.

Інформаційно орієнтовані вимоги дозволяють визначитись з основними запитами та основними бізнес-цілями побудови СД. Аналіз цієї групи вимог надає змогу встановити одну або кілька цілей СД, а також ідентифікувати основні бізнес-події та показники, які аналізуватимуться кінцевим користувачем.

Наприклад, бізнес-цілі можуть визначатися таким чином: метою створення СД є аналіз витрат на «Випуск продукції» та доходу від її «Збуту споживачам». Отже, основними бізнес-подіями, що аналізуватимуться у СД, виступають *випуск продук-*

ції та *її збут*, які будуть представлені в СД показниками «*Виробничі витрати*» і «*Дохід від збуту*» з вимірами «*Продукція*» і «*Споживач*».

Основні бізнес-запити до сховища. Вивчивши вимоги до СД кінцевих користувачів, необхідно сформулювати основні бізнес-запити до СД, що проектується. Бізнес-запити являють собою інформаційні потреби кінцевих користувачів в аналітичній інформації. При опитуванні користувачів необхідно визначити основні типи запитів. Найпоширенішими є такі:

1. *Запити перевірки*, результат виконання яких підтверджує якусь бізнес-подію. Наприклад, з'ясування факту надходження коштів від певного споживача за певний товар.

2. *Запити порівняльного аналізу*, результат виконання яких порівнює дані про якісь факти. Наприклад, порівняння даних про збут певного виду продукції за останній місяць з даними попереднього місяця.

3. *Запити тенденції*, результати виконання яких аналізують та виявляють тенденції у зміні певних фактів у часі. Наприклад, як змінювався дохід від реалізації певного виду продукції за останні 12 місяців.

4. *Запити для аналізу відношень, ранжування та кластеризації*, результати виконання яких дозволяють певним чином класифікувати та ранжувати дані про певні бізнес-події. Наприклад, ранжування кращих споживачів продукції за обсягами продажу упродовж останнього року.

5. *Запити статистичного аналізу* дозволяють провести статистичний аналіз фактів, що характеризують певні бізнес-події. Наприклад, обчислення середнього (мінімального, максимального) доходу від реалізації продукції за певним регіоном.

При формулюванні запиту необхідно описувати його так, щоб можна було визначити основні змінні, які підлягають аналізу, та виміри, за якими формуватимуться аналітичні звіти. З'ясувавши всі запити користувачів до СД, необхідно побудувати таблицю взаємозв'язків вимірів та змінних із запитами.

Вимірне моделювання СД на конкретному прикладі. Нехай дослідження предметної області та опитування користувачів дозволило виявити такі запити до СД:

1. Проаналізувати *середні залишки* кожної моделі «Виробу» в розрізі «Виробників» за певний місяць.

2. Проаналізувати підсумкові *витрати* та *дохід* на поточну дату від реалізації кожної моделі «Виробу» в розрізі «Магазинів».

3. Надати підсумкові дані на поточну дату про *витрати* та *дохід*, отримані від реалізації кожної моделі «Виробу» в розрізі «Регіонів» і «Виробників».

4. Проаналізувати за останній тиждень (місяць) у розрізі «Магазинів» *відсотки товарів*, на які можна встановити знижку, та тих, за якими фактично було проведено уцінку.

5. Проаналізувати *відсотки продажу* за минулий місяць за кожною моделлю «Виробу» через оптову і роздрібну торгівлю, через столи замовлень.

6. Які «Моделі» та «Вироби» не були продані минулого тижня (місяця)?

7. Проаранжувати моделі «Виробів», які були продані минулого місяця за обсягами продажу, витратами та отриманим доходом.

8. Які «Магазини» не мали жодного продажу певної моделі «Виробу» за минулий місяць?

9. За кожною моделлю «Виробів» виявити ті «Магазини», в яких не було продажу за минулий місяць.

Первинний аналіз запитів дозволив виокремити певну сукупність вимірів та змінних і згрупувати їх у табл. 5.1.

Більш детальний аналіз семантики запитів вказує на необхідність введення ще такого виміру, як «Час». При введенні в модель СД виміру часу необхідно визначитись зі ступенем його деталізації. З аналізу запитів можна визначити одиниці часу, які необхідні для аналізу, та ієрархію між ними. У цьому конкретному випадку це такі одиниці, як день, тиждень, місяць, квартал, рік.

Вивчення та аналіз таблиці дозволяє виокремити початковий набір фактів, які необхідно відображати в СД. Початковий набір фактів визначається за таким правилом: змінні, що описуються однаковими наборами вимірів, можна об'єднати в один факт.

Таблиця 5.1

Таблиця співвідношень вимірів і показників

Виміри та змінні	Номер запиту								
	1	2	3	4	5	6	7	8	9
Виміри									
Магазини		+		+				+	+
Виробники	+		+						
Вироби	+	+	+		+	+	+	+	+
Змінні									
Середні залишки	+								
Витрати		+	+				+		
Дохід		+	+				+		
Кількість продано							+		
Відсоток товарів, що мають право на знижки				+					
Відсоток товарів, за якими було проведено знижки				+					
Відсоток роздрібного продажу					+				
Відсоток продажу через стіл замовлень					+				
Відсоток оптового продажу					+				

Тобто виокремлення фактів здійснюється шляхом перегляду таблиці і виявлення однакових вимірів. Виявивши однакові набори вимірів, до них, приєднуються відповідні їм змінні. Результат виокремлення фактів на основі такого аналізу даних таблиці (табл. 5.1) наведено на рис. 5.35.

Необхідно звернути увагу, що деякі запити (запит 6, 8, 9) не мають жодної змінної, які були з ними пов'язані. Якби ці запити не були об'єднані з іншими запитами, що мають змінні до відповідних фактів, то вони б створили факти, що не мають змінних. Факти, що не мають змінних, називаються недостатніми, або неповними. Вони фіксують лише факт певної події, не описуючи її конкретними даними.

Після виокремлення фактів необхідно провести уточнення їх структури шляхом оцінки адитивності змінних та визначення ступеня деталізації вимірів. Аналізуючи факт 1, можна виявити таку проблему деталізації. Середні залишки є величиною, яка визначається щомісяця (запит 1), дохід та витрати (запит 3) – що-

денними характеристиками. Тобто зберігати в такому вигляді факт неможливо. У такому разі є дві альтернативи: розбити факт 1 на два факти чи деталізувати вимір часу до найнижчого його рівня, яким буде календарна дата, тобто день.

Факт 1

Код виробника
Код виробу
Код часу
Середні залишки
Витрати
Дохід

Факт 2

Код магазину
Код виробу
Код часу
Витрати
Дохід

Факт 3

Код виробу
Код часу
Витрати
Дохід
Кількість продано
Відсоток роздрібного продажу
Відсоток продажу через стіл замовлень
Відсоток оптового продажу

Факт 4

Код магазину
Код часу
Відсоток товарів, що мають право на знижки
Відсоток товарів, по яких було проведено знижки

Рис. 5.35. Набір початкових фактів

Оскільки середня кількість за місяць є величиною *неадитивною*, то краще зберігати у факті 1 адитивну змінну, що характеризує фактичну величину наявності виробів за кожен день.

При реалізації запиту 1 *середні залишки* за місяць розраховуватимуться на основі фактичних даних про наявність виробів на кожну дату. Факт 2 теж має проблеми з виміром часу. Запит 2 стосується аналізу за конкретну дату, тобто день, а запити 8 і 9 потребують даних за місяць. Тобто факт 2 об'єднує дані, не сумісні за виміром часу. Проте змінні, що входять до цього факту, є повністю адитивні, тому коригування факту 2 полягатиме лише в тому, що найменшою одиницею часу для нього буде конкретна дата, тобто день.

У фактах 3 і 4 теж об'єднані запити з різними вимірами часу – тиждень і місяць. Крім того, факти 3 і 4 мають змінні відсотків, які є неадитивними, тобто неможливо змінити ступінь деталізації часу, оскільки відсотки не підсумовуються. Таким чином,

першим рішенням, що лежить на поверхні, є представлення замість одного факту кількох, кожен з яких окремо характеризує дані за тиждень і місяць. Але це призведе не лише до збільшення кількості фактів, а і до ускладнення моделі, що зробить її заплутанішою для аналітиків. Тому краще визначити за найнижчий ступінь деталізації часу день, а дані відсотків замінити фактичними даними.

Таким чином, у факті 3 змінні «Відсоток роздрібного продажу», «Відсоток продажу через стіл замовлень» і «Відсоток оптового продажу» будуть замінені на «Кількість роздрібного продажу», «Кількість продажу через стіл замовлень»; «Кількість оптового продажу». Враховуючи, що в цьому факті є змінна, яка характеризує загальні обсяги продажу за кожною моделлю виробу, то змінні, які замінять відсотки, дозволять за потреби її розрахувати. Аналогічно у факті 4 відсотки замінимо, відповідно, на «Кількість товарів, що мають право на знижку» та «Кількість товарів, за якими було проведено знижки».

Набір фактів після визначення ступеня деталізації вимірів та перевірки на адитивність змінних представлено на рис. 5.36.

Факт 1

Код виробника
Код виробу
Код часу (день)
Залишки за день
Витрати
Дохід

Факт 2

Код магазину
Код виробу
Код часу (день)
Витрати
Дохід

Факт 3

Код виробу
Код часу (день)
Витрати
Дохід
Кількість продано
Кількість роздрібного продажу
Кількість продажу через стіл замовлень
Кількість оптового продажу

Факт 4

Код магазину
Код часу (день)
Кількість товарів, що мають право на знижки
Кількість товарів, за якими було проведено знижки

Рис. 5.36. Уточнений набір фактів після вирішення проблем деталізації та адитивності

Об'єднання фактів. Вирішення проблеми зі ступенем деталізації вимірів та адитивністю змінних надає змогу розглянути питання можливості об'єднання фактів, що розширює діапазон досліджень, оскільки, як правило, призводить до збільшення кількості вимірів.

Об'єднання фактів надає такі переваги:

✓ за рахунок зменшення кількості фактів спрощується модель;

✓ розширюються можливості аналізу за рахунок зв'язування більшої кількості змінних і вимірів на вищому рівні деталізації;

✓ спрощується адміністрування СД та робота з ним користувачів-аналітиків, оскільки для адміністрування та реалізації запитів потрібно переглядати менше фактів.

Першим кроком оцінки перспектив об'єднання фактів є визначення за кожною змінною можливістю підключення нових вимірів для підвищення її інформативності та ступеня деталізації. Об'єднувати можна факти, що містять спільні виміри. Виміри факту 1 (код виробу та код часу) представлені у фактах 1 і 3. Об'єднати факт 1 і 2 неможливо і недоцільно, оскільки факт 1 містить дані про залишки виробів у розрізі кожного «Виробника», а факт 2 – дані про продаж виробів у розрізі «Магазину», а потужність відношень «Виробник» : «Магазин» = Б : Б.

Дослідимо можливість об'єднання фактів 2 і 3, оскільки вони містять семантично споріднену інформацію, що характеризують витрати і дохід та обсяги продажу за виробами. Об'єднання цих фактів можливе шляхом включення виміру, що характеризує магазин до факту 3. Таким чином, у результаті об'єднання отримаємо факт, представлений на рис. 5.37.

Цьому факту присвоюємо номер 2_3 і проаналізуємо його на можливість об'єднання з фактом 4.

Змінні «Кількість товарів, що мають право на знижки» та «Кількість товарів, за якими було проведено знижки» доречніше вказувати в розрізі кожного *виробу*. Причому досить залишити лише першу змінну, включивши її у факт, що утворився внаслідок об'єднання фактів 2 і 3.

Факт 2_3

Код магазину
Код виробу
Код часу (день)
Витрати
Дохід
Кількість продано
Кількість роздрібного продажу
Кількість продажу через стіл замовлень
Кількість оптового продажу

Рис. 5.37. Результати об'єднань фактів 2 і 3

Змінну «Кількість товарів, за якими було проведено знижки» можна не враховувати, оскільки є змінні, які характеризують обсяги продажу, які у разі необхідності дозволять розрахувати кількість товарів, проданих зі знижками. Результат об'єднання фактів 2, 3, і 4 наведено на рис. 5.38.

Таким чином, після виконання процедури об'єднання фактів на завершальному етапі маємо два факти: «факт 1» та «факт 2_4», який дістали в результаті об'єднання фактів 2, 3 і 4. Назвемо, відповідно, ці факти «Виробництво» і «Продаж» (структури цих фактів наведені на рис. 5.39).

Можливості аналізу факту «Продаж» розширяться при добавленні виміру «Виробник». Спроектувавши таблиці змінних фактів «Виробництво» і «Продаж» та підключивши до них відповідні таблиці вимірів, отримаємо такі дві моделі СД типу «зірка» (рис. 5.40 і 5.41).

Факт 2_4

Код магазину
Код виробу
Код часу (день)
Витрати
Дохід
Кількість продано
Кількість роздрібного продажу
Кількість продажу через стіл замовлень
Кількість оптового продажу
Кількість товарів, що мають право на знижки

Рис. 5.38. Результати об'єднань фактів 2, 3 і 4

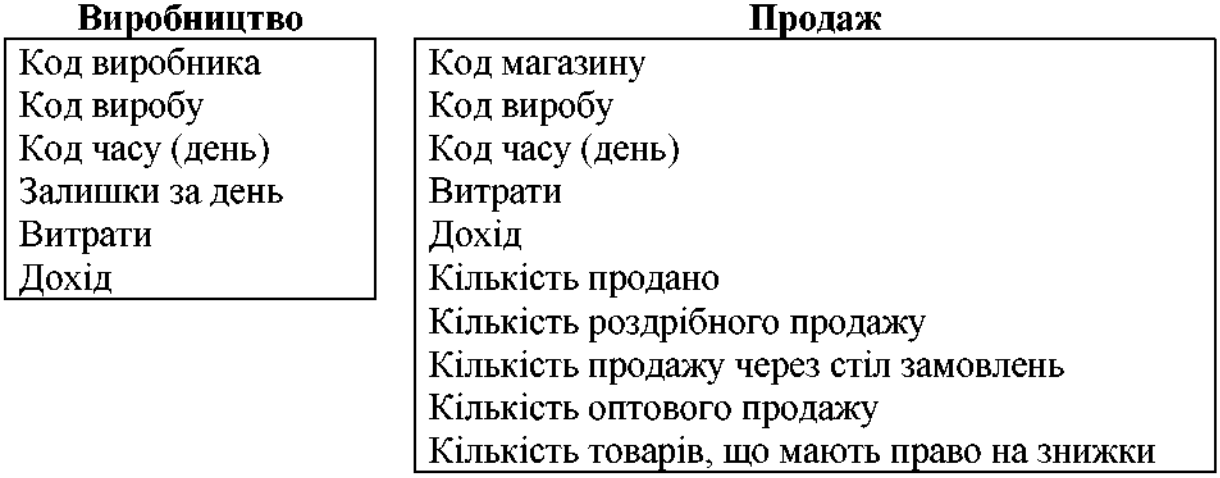


Рис. 5.39. Кінцевий набір фактів



Рис. 5.40. Модель типу «зірка» – «виробництво»



Рис. 5.41. Модель типу «зірка» – «продаж»

У моделі СД «Виробництво» може виникнути необхідність ввести інформацію про банк. Виникає необхідність нормалізації даних у таблиці вимірів «Виробник» створенням консольної таблиці «Банк». У такому випадку отримаємо модель типу «сніжинка» – «виробництво» (рис.5.42).



Рис. 5.42. Модель типу «сніжинка» – «виробництво»

Якщо денормалізована таблиця вимірів є надто великою і при цьому до деяких її стовпчиків запити робляться частіше, ніж до інших, то доцільно для підвищення ефективності виконання запитів розбити цю таблицю на дві окремі таблиці вимірів. Дві нові таблиці вимірів, які будуть отримані внаслідок цієї операції, пов'язуються неідентифікуючим зв'язком. Виконання такої операції зменшить час виконання запитів.

Вимірне проектування подібне до проектування реляційних баз даних, але відрізняється акцентами і цілями, які ставляться перед проектувальником. Якщо основною задачею проектування реляційних баз даних є забезпечення цілісності і узгодженості даних при їх завантаженні, то основною метою при проектуванні сховищ є мінімізація часу на виконання складних аналітичних запитів.

Запитання для закріплення

1. Яка послідовність етапів формування вимірної моделі СД?

2. Надати визначення цілей побудови СД.

3. Надати визначення вимог користувачів.

4. Яка особливість аналітичних бізнес-запитів?

5. Як побудувати таблицю взаємозв'язку змінних та вимірів?

6. Як виявити запити, що описуються однаковими вимірами?

7. Як формується початковий набір факту?

8. Як здійснюється аналіз змінних на адитивність?

9. Як здійснюється аналіз ступеня деталізації фактів?

10. Як здійснюється аналіз сумісності фактів за часовими вимірами?

11. Для чого уточнюється структура початкового набору фактів?

12. Для чого досліджуються факти на можливість їх об'єднання?

13. Як формується кінцевий набір фактів?

14. Як здійснюється проектування таблиць вимірів?

15. Як здійснюється об'єднання таблиці змінних фактів і таблиць вимірів фактів у вимірну модель СД?

5.6. Лабораторна робота 11

Тема. Розробка вимірної моделі СД індивідуального об'єкта дослідження і розвиток «Наскрізного індивідуального проекту»

Мета роботи:

✓освоїти методологію вимірного моделювання СД;

✓виробити навички ескізного проектування вимірного СД для OLAP системи.

Завдання

1. Розробити зміст 2–3 аналітичних запитів (бізнес-завдань) до сховища даних індивідуального об'єкта дослідження.

2. На основі аналітичних запитів (бізнес-завдань) розробити багатовимірну модель сховища даних індивідуального об'єкта дослідження.

3. Розробити SQL-запити на спроектованій, відповідно до п. 2, багатовимірній моделі сховища даних, що розроблені в п. 1.

Хід роботи

1. Проведення аналізу індивідуального об'єкта дослідження.
2. Виконати всі пункти, які передбачені у завданні.
3. Враховуючи індивідуальні особливості об'єкта дослідження допускається розширення і поглиблення робіт «Індивідуального проекту» (може ускладнитись процес отримання зворотної інформації про виконання індивідуальних завдань та їхнього корегування).

Оформлення звіту

Звіт про виконану роботу оформляється згідно із «Загальними рекомендаціями до виконання лабораторних робіт та оформлення звіту» (додаток А: А.1, А.2).

Контрольні запитання

1. Яка послідовність етапів формування вимірної моделі СД індивідуального об'єкта дослідження?
2. Надати визначення цілей побудови СД індивідуального об'єкта дослідження.
3. Яка особливість аналітичних бізнес-запитів індивідуального об'єкта дослідження?
4. Як здійснюється проектування таблиць вимірів індивідуального об'єкта дослідження?
5. Сформулювати способи проектування сховищ даних.
6. Сформулювати підходи до проектування сховищ даних.
7. Сформулювати підходи до визначення основних елементів СД.
8. Сформулювати визначення та вимоги до змінних.
9. Сформулювати визначення ступеня деталізації змінних.
10. Сформулювати визначення та вимоги до вимірів.
11. Сформулювати визначення та вимоги до фактів.
12. Надати визначення мотивації (мета) створення та розвитку сховища даних (навіщо?).
13. Надати визначення дій, які виконуються з даними при їх завантаженні (як?).

14. Надати визначення місцю розташування даних (де?).

15. Надати визначення користувачів, які використовують сховище даних (хто?).

16. Надати визначення часу завантаження та обчислення підсумкових даних (коли?).

17. Надати визначення сутності елементів та процесу наповнення сховища даних (що?).

5.7. Технологія реалізації вимірної моделі СД за допомогою CASE-засобу AllFusion ERwin DM

Прототип вимірної моделі СД. На рис. 5.43 відображено модель СД типу «зірка». Центральною таблицею цієї моделі є таблиця змінних фактів, що відображає обсяги та суму продажу продукції (таблиця «Реалізація») у розрізі календарних періодів та ринків. Таблицями вимірів є: «Довідник_продукції», «Довідник_часових_проміжків», «Довідник_ ринків» та «Довідник_клієнтів», які пов'язані ідентифікуючими зв'язками з таблицею змінних.



Рис. 5.43. Модель типу «зірка» – «реалізація»

Остання може складатися з великої кількості записів і містити фактичні чи консолідовані дані, необхідні для аналізу обсягів продажу. Слід звернути увагу, що при проектуванні таблиці змінних фактів до неї заносяться лише поля «Кількість_реалізованої_продукції», «Сума_ реалізованої_продукції» та «Сума_оплаченої_продукції».

Ключові поля: «Код_продукції», «Код_ринку», «Код_час» та «Код_клієнта» мігрують (автоматично переносяться) з таблиць вимірів при створенні ідентифікуючих зв'язків. Тобто ключові поля таблиці змінних фактів являють собою набір полів, що складається з ключів таблиць вимірів.

При створенні таблиць вимірів слід пам'ятати, що цей тип таблиць містить дані, що практично не змінюються або ж змінюються рідко. Враховуючи те, що таблиці у сховищі типу «зірка» мають ненормалізовану структуру, то зміна таблиць вимірів може призвести до аномалій і різного роду колізій.

Для того щоб запобігти виникненню різного роду протиріч при зберіганні даних у середовищі CASE-засобу AllFusion ERwin DM, дозволяється задавати такі параметри редагування таблиць вимірів:

✓ перезаписування старих даних новими, при якому старі дані знищуються;

✓ створення нового запису в таблиці вимірів з новими даними і тимчасовими вимірами. У такому разі старі дані зберігаються і можливе відстеження історії внесення змін до даних, але при цьому необхідно генерувати ключі для посилань на старі дані;

✓ запис нових даних у додатковому полі того самого запису.

У такому разі зберігається початкове й останнє значення. Усі проміжні значення втрачаються, тобто не видно історії внесення змін до таблиць вимірів упродовж усього терміну експлуатації сховища даних.

Етапи проектування СД. Для проектування СД необхідно виконати таку послідовність дій у середовищі AllFusion ERwin Data Modeler Computer Associates:

✓ запуск пакета;

✓ створення нової моделі сховища даних;

✓ створення сутностей та робота з моделлю;

✓ заповнення сутностей атрибутами;

✓ побудова зв'язків;

✓встановлення параметрів таблиць сховища даних (*особливості вкладки Dimensional, вибір типу правил маніпулювання даними*);

✓визначення метаданих *(документування інформації про джерела даних, внесення додаткової інформації про метод)*;

✓ друк звіту.

Запуск пакета. Запуск пакета здійснюється із середовища ОС Windows завдяки завантаженню через кнопочне меню «Пуск» за таким форматом: ***Пуск/ Программы/ Computer Associates/ AllFusion/ ERwin Data Modeler r.7/ ERwin Data Modeler r.7.***

Після запуску пакета автоматично відкриється вікно діалогу, у якому слід обрати опцію: ***Open an existing file*** – для вибору та відкриття вже створеної моделі (рис. 5.44), і натиснути кнопку ОК.

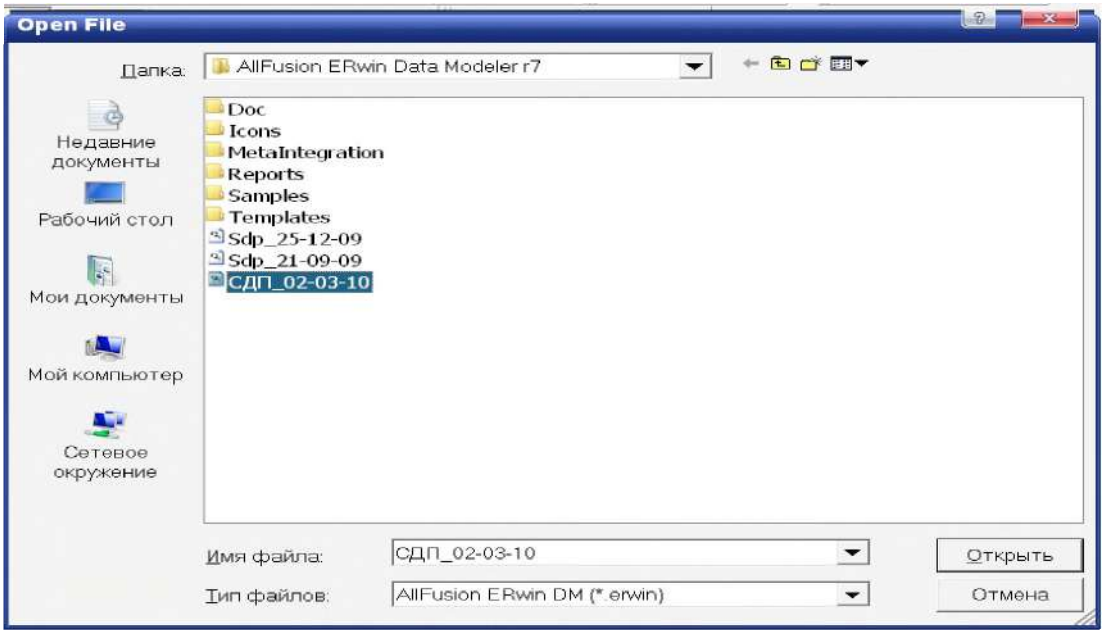


Рис. 5.44. Початкове діалогове вікно пакета AllFusion ERwin Data Modeler

Створення нової моделі сховища даних. Для створення нової моделі слід обрати пункт меню ***File/New*** або натиснути відповідну піктограму, після чого відкриється вікно ***Create Model – Select Template***.

Для вибору методології моделювання сховища даних треба у відкритому вікні ***Create Model – Select Template*** обрати:

в опції ***New Model Type*** тип моделі ***Physical***;
в опції ***Target Database*** тип ***Database*** і версію ***OK*** (рис. 5.45).

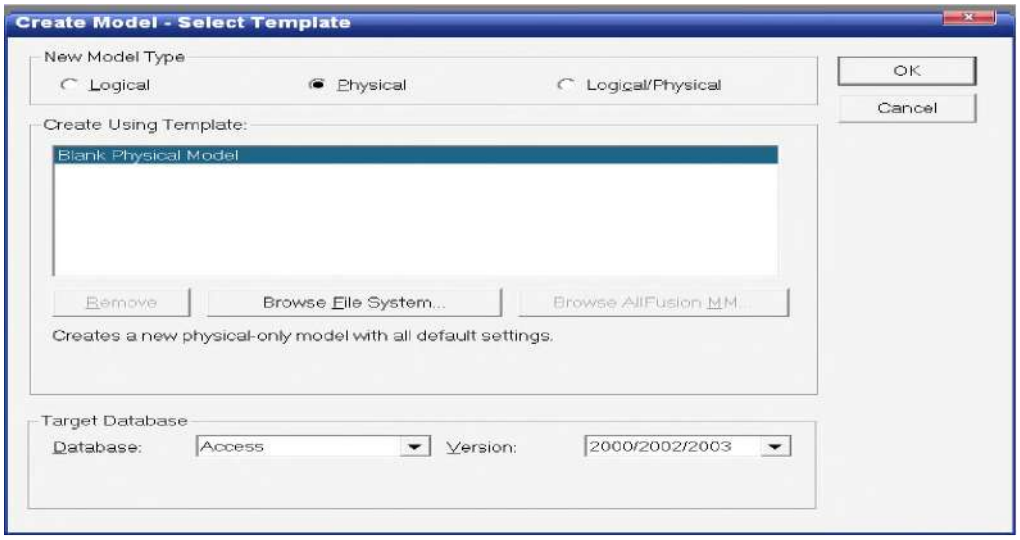


Рис.5.45. Діалогове вікно AllFusion ERwin DM типу моделей і СУБД

Перший шлях вибору вимірного моделювання СД (*до натискування кнопки* ***OK***): у відкритому вікні ***Create Model — Select Template*** за допомогою кнопки ***Browse File System*** обрати запропонований шаблон СД:

C:\ProgramFiles\ComputerAssociates/ AllFusion DM Tempiate\Dimens.ert.

Використовуємо інший шлях для вибору вимірного моделювання СД – за допомогою головного меню ***Model/Model Properties*** – відкриється вікно для опису властивостей моделі.

Із пропонованого набору вкладок, з погляду на мету лабораторної роботи, нас цікавлять дві вкладки: ***General*** та ***Notation***.

Спочатку на вкладці ***Notation*** (рис. 5.46) слід пересвідчитися, що активована методологія вимірного моделювання ***(Dimensional Modeling).***

На вкладці ***General*** (рис. 5.47) вводимо ім'я моделі СД, ПІП автора моделі.

Важливо активізувати групу опцій ***Enable Modeling Features***, яка надає *вимірної* функціональності розглядуваній моделі (*специфічні ролі таблиць у сховищі даних, іконки, специфічні рівні відтворення тощо*).

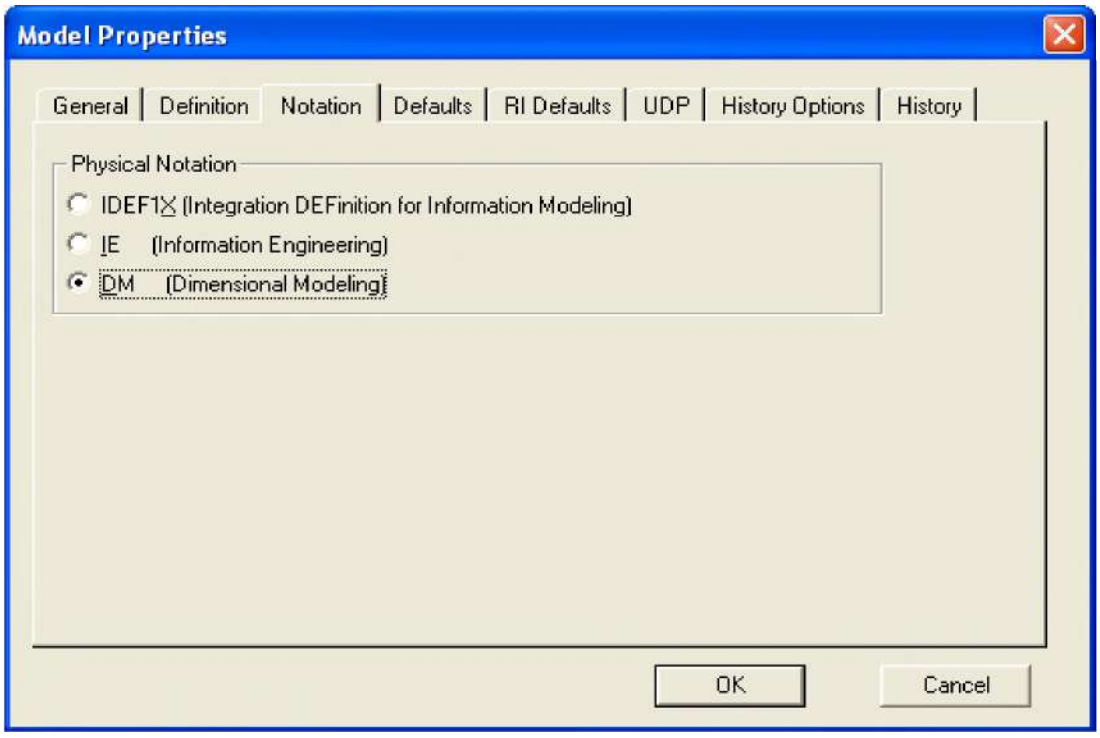


Рис. 5.46. Вкладка нотацій діалогового вікна властивостей моделі

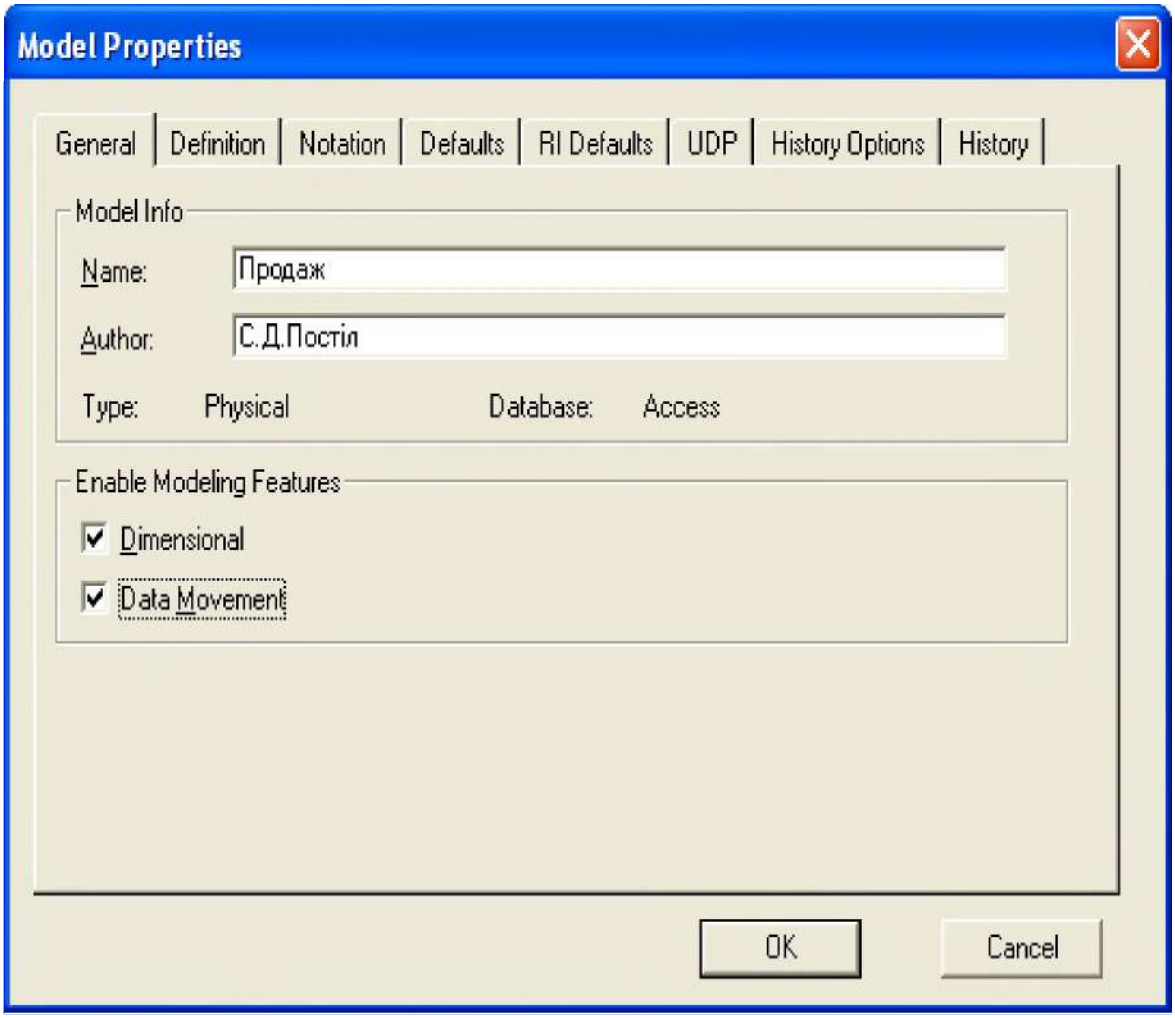


Рис. 5.47. Вкладка загальних властивостей моделі

Зокрема, активізація прапорця ***Display conformance warnings*** вмикає автоматичну перевірку коректності вимірної моделі і видає на екран діагностичні повідомлення у таких випадках порушення синтаксису:

- ✓ таблиця факту не є у зв'язку дочірньою;
- ✓ консольна таблиця не є у зв'язку з батьківською;
- ✓ установлений ідентифікуючий зв'язок між консольною таблицею й таблицею фактів.

Обрана методологія передбачає відтворення зв'язків діагональними лініями (***Diagonal***), *не ортогональними* (ця опція встановлюється за допомогою меню ***Format/Stored Display Settings*** у групі ***Relationship lines***).

Створення сутностей та робота з моделлю. Для роботи з моделлю в AllFusion DM ERwin існує такий перелік панелей інструментів (рис. 5.48):

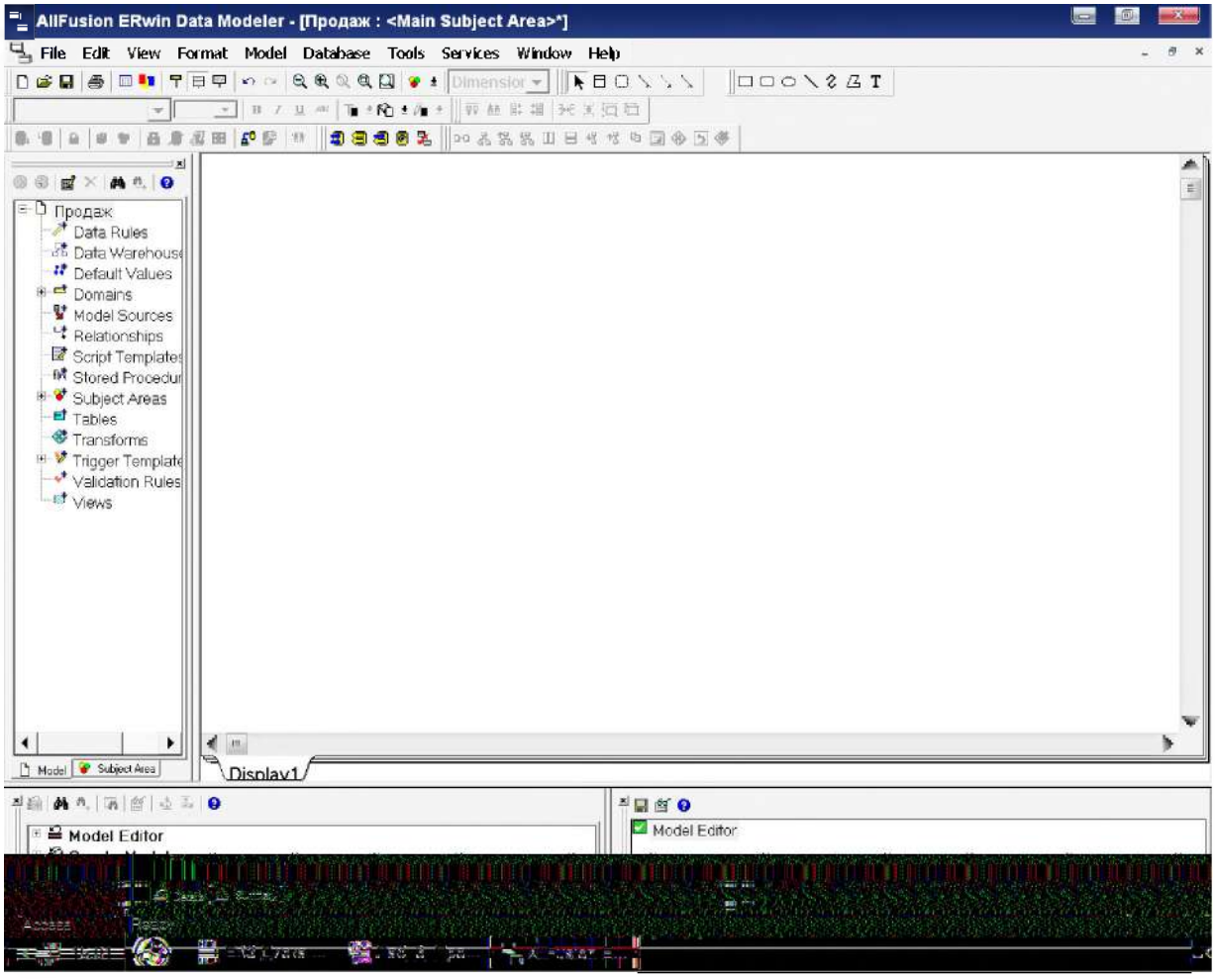


Рис. 5.48. Перелік панелей інструментів AllFusion ERwin DM

За допомогою випадаючого списку перемкнемо ERwin на роботу з фізичним рівнем моделі, оскільки саме на цьому рівні наочно відтворюються можливі ролі таблиць у сховищі даних:



– таблиця фактів (***fact table***);
– таблиця вимірів (***dimensional table***);
– консольна таблиця (***outrigger table***).

Отже, через активізацію піктограми створення незалежної таблиці здійснюємо в робочій області ERwin спроектовані сутності та наповнюємо їх атрибутами, встановлюючи зв'язки *(з урахуванням методології побудови сховища даних)*.

Заповнення сутностей атрибутами. Заповнюємо створені та пойменовані сутності атрибутами, користуючись діалоговим вікном ***Attributes*** (рис. 5.49).

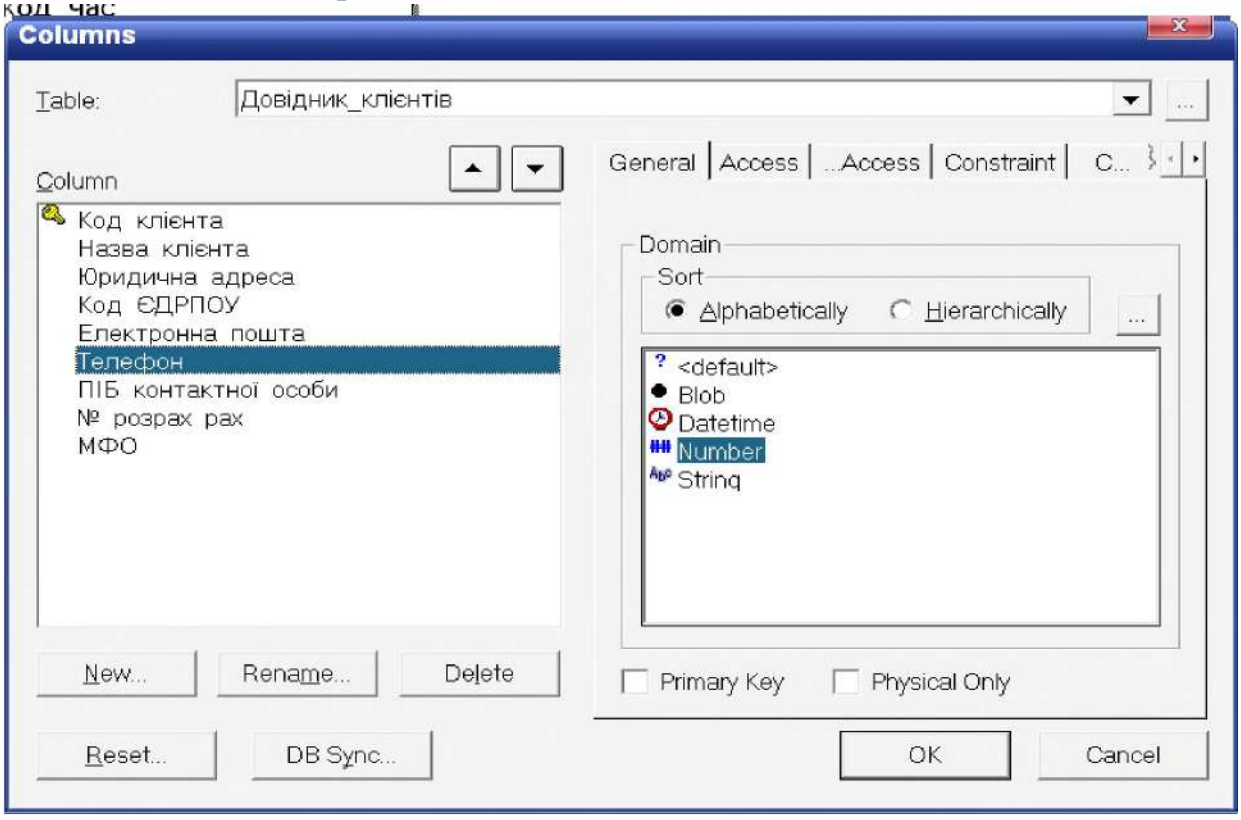


Рис. 5.49. Діалогове вікно атрибутів сутності

Для виклику необхідно у контекстному меню сутності обрати ***Attributes***. У цьому вікні за допомогою кнопок ***New, Rename та Delete*** можна здійснювати створення, перейменування та вилучення атрибутів сутності.

При створенні нового атрибута необхідно вказати, крім імені на логічному рівні ***Attribute name***, ім'я атрибута при переході до фізичного рівня – ***Column name.***

Опція ***Primary key*** вмикається для первинного ключа сутності, крім того, для кожного атрибута необхідно задати домен, а на вкладці ***Datatype*** – відповідний тип даних.

Побудова зв'язків. Для побудови зв'язків між таблицею вимірів та таблицею фактів слід клацнути на кнопці ідентифікуючого зв'язку, потім – на таблиці вимірювань, а далі – на таблиці фактів.

Консольну таблицю можна пов'язувати з відповідною таблицею вимірювань ідентифікуючими або не ідентифікуючими зв'язками. Питання про тип зв'язку вирішується виходячи з аналізу семантики предметної області. На рис. 5.50 подано створений проект сховища даних на фізичному рівні.

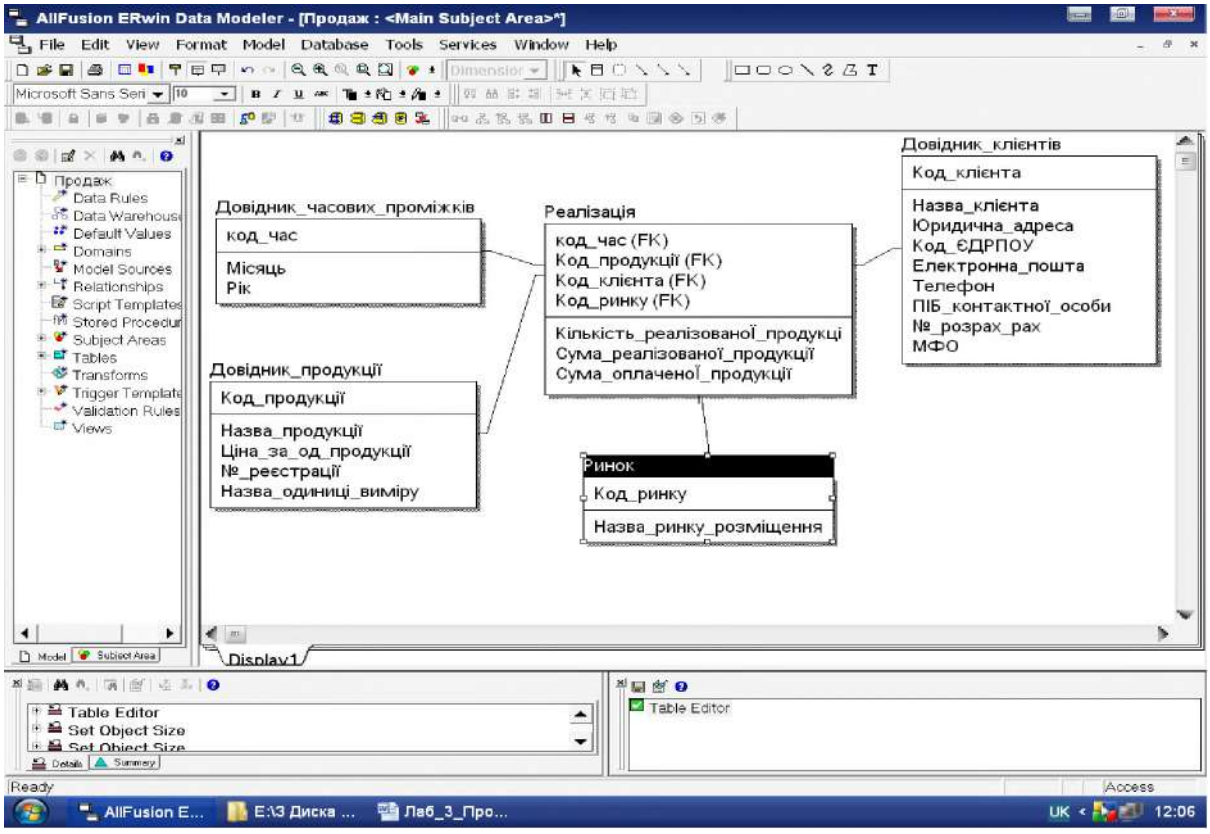


Рис. 5.50. Проект СД на фізичному рівні

Встановлення параметрів таблиць СД. Для редагування специфічних параметрів таблиць сховища даних необхідно клац-

нути правою кнопкою мишки по створеній таблиці й у контекстному меню обрати ***Table Properties***/.

У цій групі контекстного меню запропонований набір вкладок, на яких ідентифікується таблиця СД та визначаються її специфічні властивості.

Особливості вкладки Dimensional подані на рис. 5.51. На основі створених зв'язків ERwin автоматично за умовчанням визначає роль таблиці в СД (*таблиця фактів, вимірювань чи консольна*).

Для визначення ролі таблиці *вручну* необхідно *деактивізувати* опцію ***Calculate Automatically***.

При створенні таблиць вимірів (***Dimension table***) слід пам'ятати, що цей тип таблиць містить дані, які практично не змінюються або змінюються досить рідко, тому зміна таблиць вимірювань може призвести до аномалій і різного роду колізій.

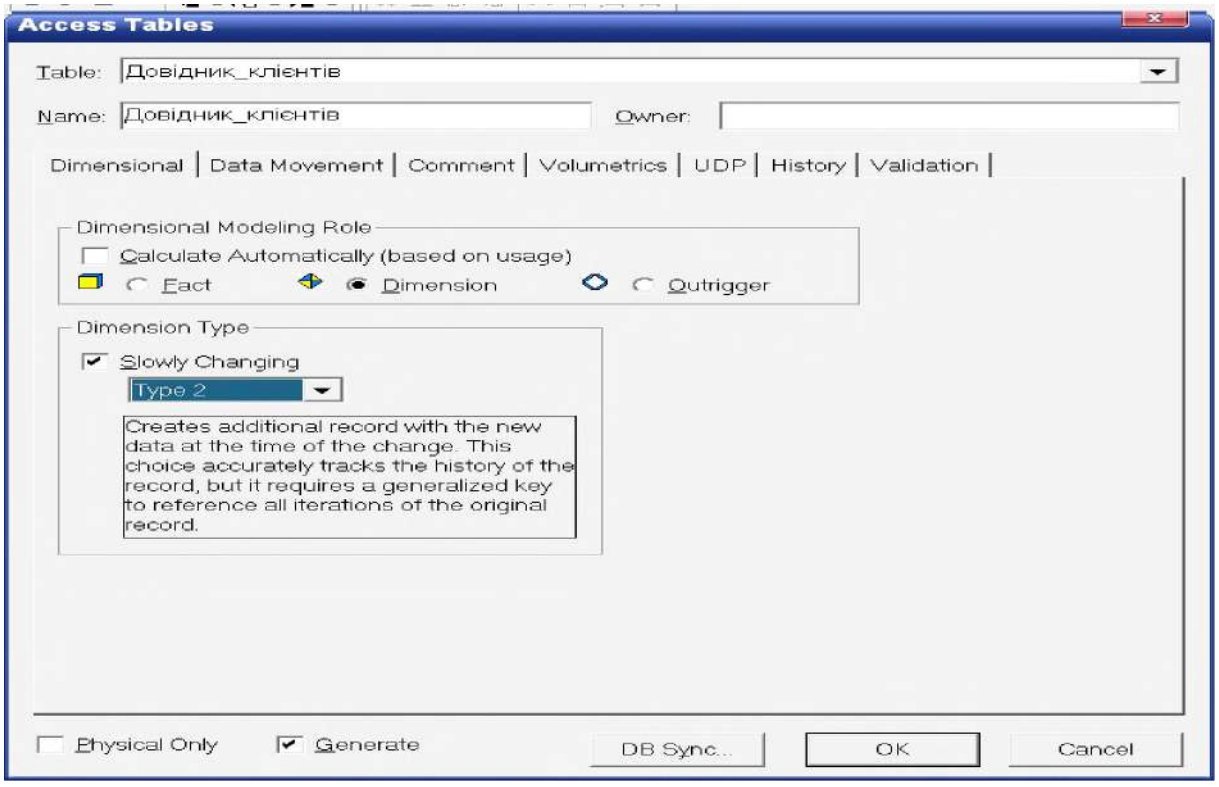


Рис. 5.51. Вкладка вимірів діалогового вікна властивостей сутності в СД

Для того щоб запобігти виникненню різного роду суперечностей при зберіганні даних ERwin уможливлює задання таких параметрів редагування таблиць вимірів (*ключ* ***Slowly Changing***):

Type 1 – перезаписування старих даних новими, при якому старі дані знищуються.

Type 2 – створення нового запису в таблиці вимірювань із новими даними і тимчасовими вимірами. Таким чином, старі дані зберігаються і можливе відстеження історії внесення змін до даних, але при цьому необхідно генерувати ключі для посилань на старі дані.

Type 3 – запис нових даних у додатковому полі того самого запису. Таким чином, зберігається початкове й останнє значення. Усі проміжні значення втрачаються, тобто не видно історії внесення змін до таблиць вимірювань упродовж усього терміну експлуатації СД.

Вибір типу правил маніпулювання даними. Обравши закладку ***Data Movement*** (рис. 5.52), слід обрати для таблиці СД (*із випадного списку вже створених попередньо правил*) відповідні шість типів правил маніпулювання даними: *оновлення* ***(Refresh)***, *поповнення* ***(Append)***, *резервне копіювання* ***(Backup)***, *відновлення* ***(Recovery)***, *архівування* ***(Archiving)***, *очистка* ***(Purge)***.

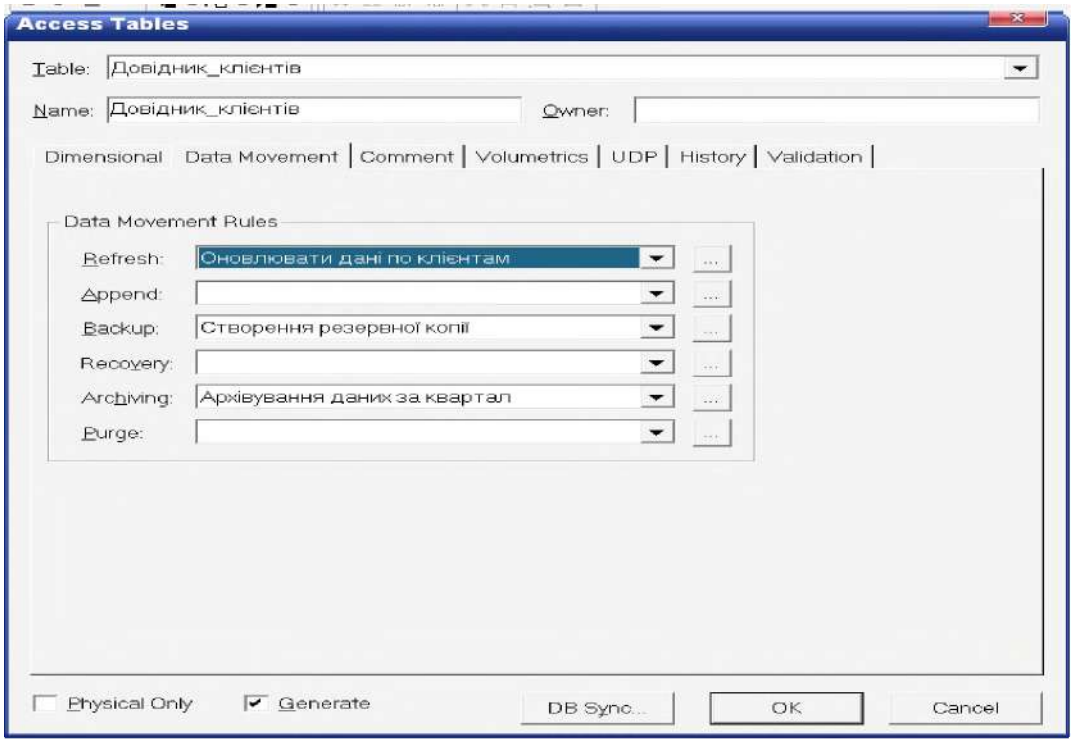


Рис. 5.52. Вікно маніпулювання даними властивостей сутності в СД

Для створення нових правил потрібно натиснути кнопку для виклику вікна редактора правил, а для створюваного правила необхідно задати його ім'я (***Rule name***), обравши один із пропоно-

ваних шести типів ***(Type)***, та записати словесний опис правила на вкладці ***Definition***, а на вкладці ***Attachment*** (рис. 5.53) слід обрати, яка таблиця є об'єктом створеного правила.

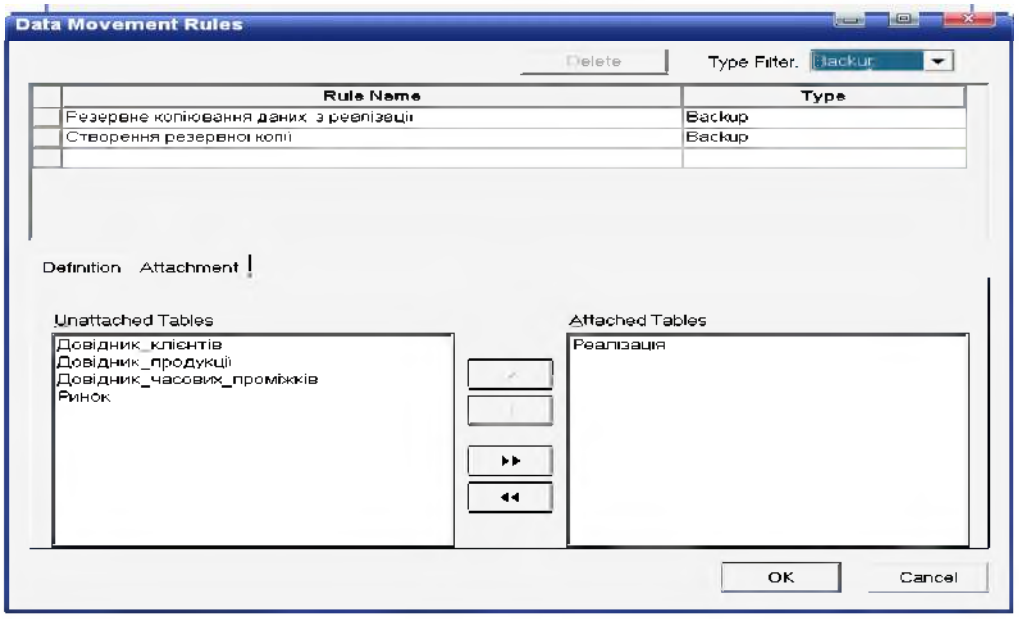


Рис. 5.53. Діалогове вікно редактора правил маніпулювання даними у СД

Визначення метаданих. При проектуванні СД в ERwin можна визначити метадані:

– джерела даних для СД;

– метод, за допомогою якого дані обираються, фільтруються, очищаються та перетворюються, перш ніж потрапити до СД.

Документування інформації про джерела даних. Для документування інформації про джерела даних використовується редактор ***Data Warehouse Sourses***, який викликається з меню ***Model/Data Sources*** (рис. 5.54).

У цьому вікні на трьох вкладках необхідно визначити опції джерел даних для проектованого СД, зокрема для обраної у верхній частині вікна у таблиці назви джерела даних:

✓ *на вкладці* ***General*****:** увести системне ім'я БД, назву сервера, де розташована БД, тип СУБД та її версію;

✓ *на вкладці* ***Detail:*** визначити імена таблиць – джерел даних, назви їхніх атрибутів, тип даних атрибутів;

✓ *на вкладці* ***Definition*****:** увести текстове визначення джерела даних.

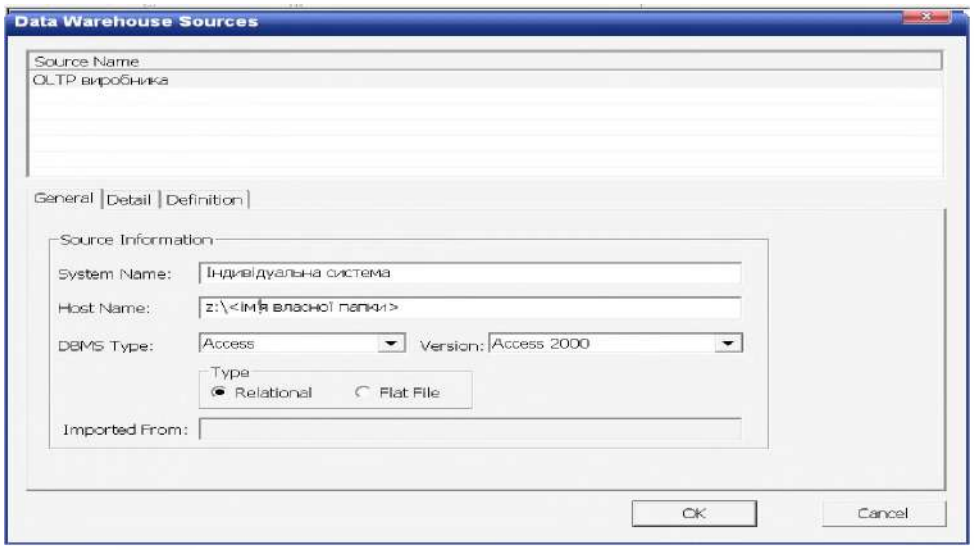


Рис. 5.54. Діалогове вікно редактора джерел даних у СД

Практичного застосування визначених вищепотенційних джерел даних набувають у вкладці ***Data Source*** (рис. 5.55) вікна редактора колонок таблиці (*викликається з меню* ***Model/Colums***). У полі ***Data Source*** для відповідного стовпчика таблиці можна призначити попередньо певне джерело даних: обрати відповідну колонку та натиснути ***Select.***

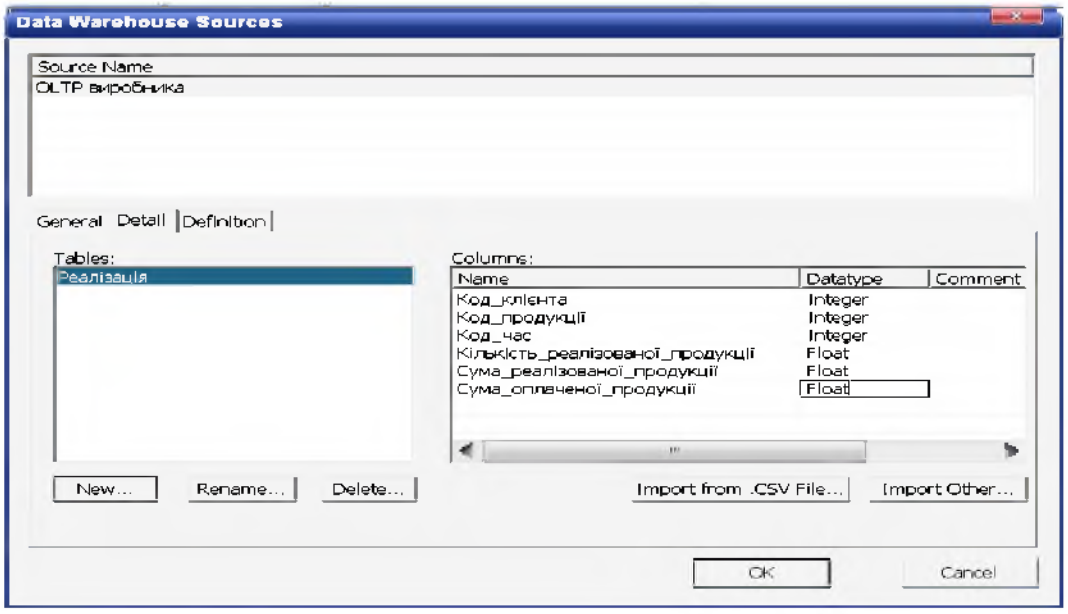


Рис. 5.55. Вкладка встановлення джерел даних діалогового вікна колонок таблиці в СД (на фізичному рівні)

Внесення додаткової інформації про метод. У полі ***Transform Comment*** вноситься додаткова інформація: про метод,

за допомогою якого дані обираються, фільтруються, очищаються та перетворюються, перш ніж потрапити до сховища даних.

Друк звіту. За допомогою майстра генерації звітів, що викликається з меню ***Took/Report Builder***, побудувати й роздрукувати звіт, що містить: інформацію про таблиці та їхні параметри, інформацію про правила маніпулювання даними та інформацію про джерела даних.

У першому вікні діалогу Report template можна обрати для роботи вже сконфігурований та збережений шаблон звіту (рис. 5.56).

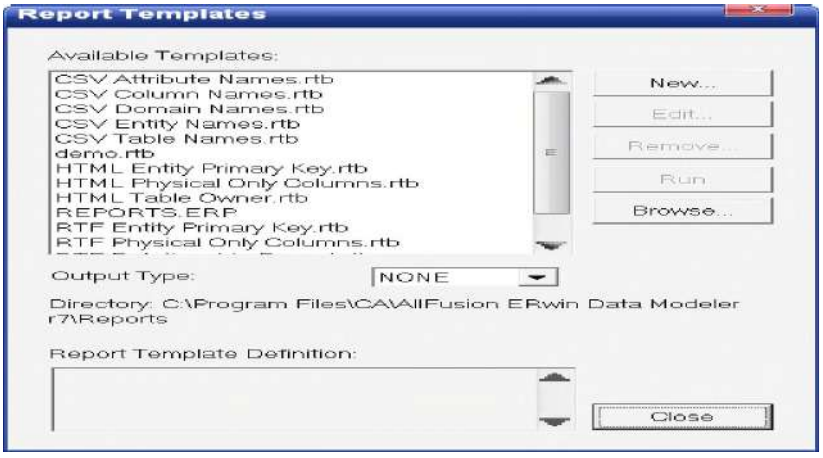


Рис. 5.56. Вікно сконфігурованих та збережених шаблонів звіту

За допомогою кнопки ***New*** можна створити новий звіт, вказавши вихідний формат ***(Output type): HTML, RTF, TXT.***

Далі у вікні конфігурування звіту ***Report template builder*** (рис. 5.57) слід встановити глобальні параметри звіту:

- ✓ формат вихідного файла;
- ✓ наявності змісту звіту;
- ✓ назви звіту;
- ✓ його оформлення;
- ✓ верхнього колонтитулу;
- ✓ нижнього колонтитулу;
- ✓ опису звіту;
- ✓ його розташування.

Примітка. ОБОВ'ЯЗКОВО у колонтитулі звіту має бути внесено ПІП студента, що виконує лабораторну роботу.

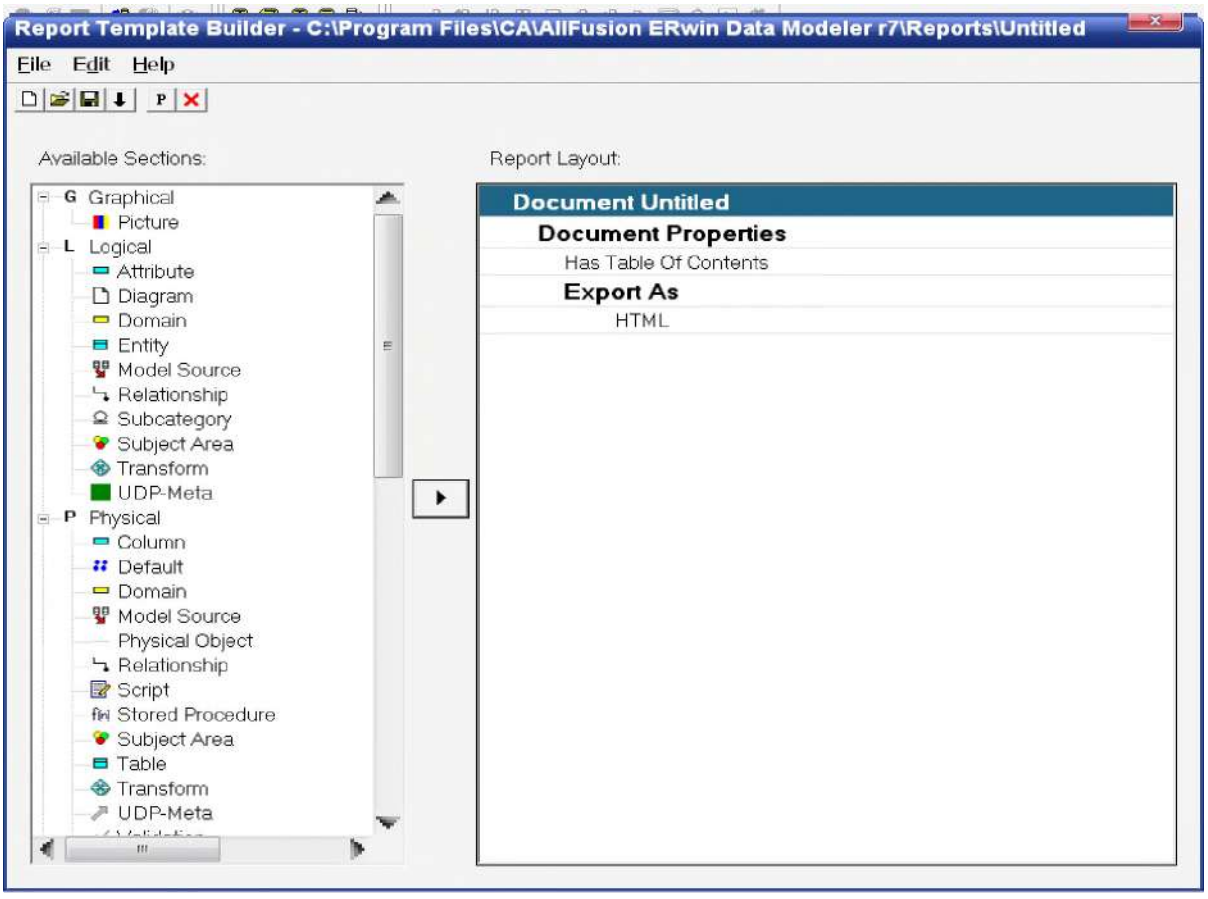


Рис. 5.57. Вікно конфігурування звіту *Report template builder*

Для конфігурування змісту звіту слід обрати з лівої частини вікна ***(Available sections***) із доступних для документування розділів потрібні та внести їх у звіт. Обраний розділ можна вилучити.

Далі у правій частині вікна (***Report Layout***) потрібно двічі клацнути на внесеній групі параметрів для безпосереднього виділення в дереві властивостей (***Property Tree***) тих параметрів, які відтворюватимуться у звіті.

Для виконання цього завдання лабораторної роботи слід у лівій частині вікна ***Report Template Builder*** обрати у групі ***Physical*** об'єкт ***Table*** та внести його у зміст звіту.

Далі необхідно виділити в обраному розділі ***Table*** у групі ***Filter*** усі основні параметри документування (рис. 5.58), опції ***Data Warehouse Rule*** та опції ***Column/Data Warehouse Source***.

Тут же, на вкладці ***Section***, можна задати оформлення обраного розділу звіту.

Для пуску звіту на генерацію потрібно натиснути кнопку запуску.

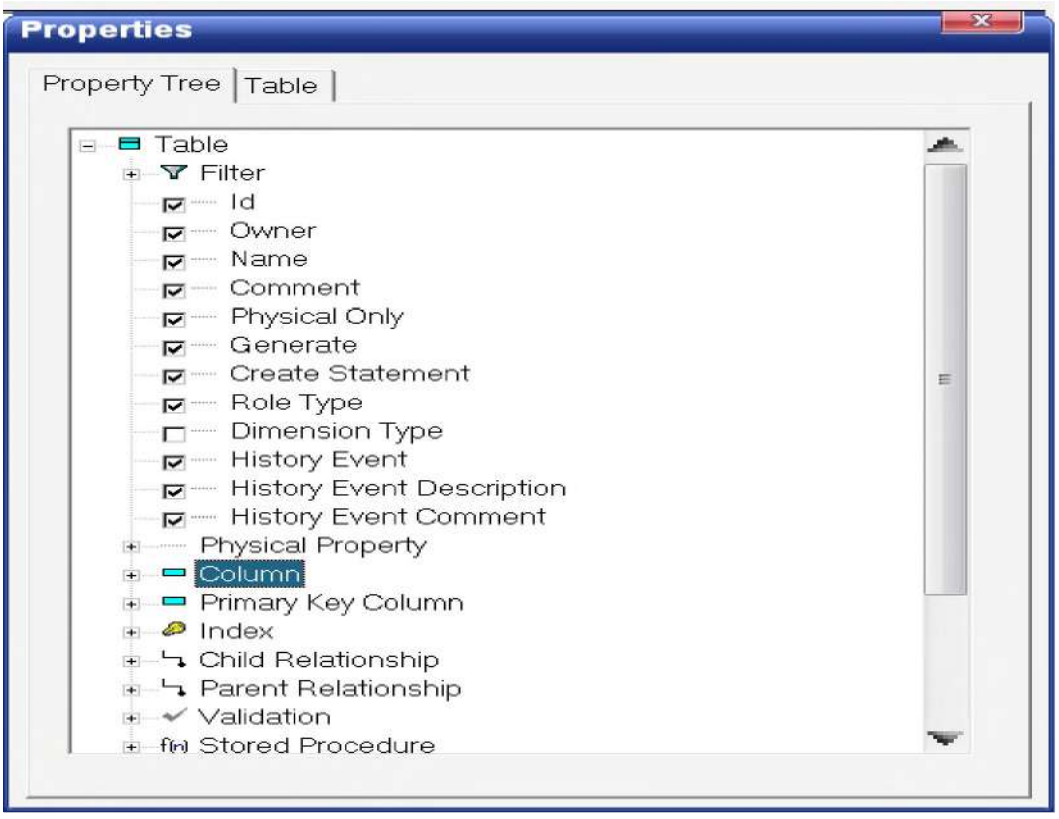


Рис. 5.58. Діалогове вікно конфігурування змісту звіту

5.8. Лабораторна робота 12

Тема. Створення проекту вимірної моделі СД на задачі-прототипі («Реалізація») за допомогою CASE-засобу Erwin Data Modeler пакета AllFussion Modeling Suite

Мета: вивчити концепцію та функціональні можливості сучасних CASE-засобів для проектування СД на прикладі з використанням CASE-засобу Erwin Data Modeler.

Завдання

1. На прикладі вимірної моделі СД задачі-прототипу («Реалізація») ознайомитися з особливостями проектування СД у середовищі CASE-засобу ERwin Data Modeler пакета AllFussion Modeling Suite:

- ✓ запуск пакета та створення нової моделі СД;
- ✓ створення сутностей та робота з моделлю;
- ✓ заповнення сутностей атрибутами та побудова зв'язків;
- ✓ встановлення параметрів таблиць сховища даних (*особливості вкладки Dimensional, вибір типу правил маніпулювання даними*);

✓ визначення метаданих *(документування інформації про джерела даних, внесення додаткової інформації про методи).*

2. Формування копії екранних вікон побудованої моделі прототипу сховища даних на фізичному рівні у середовищі CASE-засобу ERwin Data Modeler за допомогою меню ***File/Print Setup*** та ***File/Print*** (*обов'язково з наявним текстовим блоком, у якому вказане ПІП студента та номер групи*).

3. Формування електронної версії згенерованого звіту.

4. Відповіді на контрольні запитання.

Хід роботи

1. Проведення аналізу індивідуального об'єкта дослідження в контексті задачі-прототипу.

2. Виконання роботи для задачі-прототипу.

3. Виконати всі пункти, які передбачені у завданні.

4. Враховуючи індивідуальні особливості об'єкта дослідження, допускається розширення і поглиблення робіт «Індивідуального проекту» (може ускладнитись процес отримання зворотної інформації про виконання індивідуальних завдань та їхнього корегування).

Оформлення звіту

Звіт про виконану роботу оформляється згідно із «Загальними рекомендаціями до виконання лабораторних робіт та оформлення звіту» (додаток А: А.1, А.2).

Запитання на закріплення

1. Як здійснюється запуск CASE-засобу AllFusion ERwin DM?
2. Як здійснюється створення нової моделі СД у CASE-засобі AllFusion ERwin DM?
3. Як здійснюється створення сутностей та робота з моделлю СД у CASE-засобі AllFusion ERwin DM?
4. Як здійснюється заповнення сутностей атрибутами моделі СД у CASE-засобу AllFusion ERwin DM?
5. Як здійснюється побудова зв'язків моделі СД CASE-засобу AllFusion ERwin DM?
6. Як здійснюється встановлення параметрів таблиць моделі СД у CASE-засобу AllFusion ERwin DM?

7. Як здійснюється визначення метаданих моделі СД у CASE-засобу AllFusion ERwin DM?

8. Як здійснюється друк звіту в CASE-засобу AllFusion ERwin DM?

5.9. Лабораторна робота 13

Тема. Створення проекту у середовищі CASE-засобу Erwin Data Modeler вимірної моделі СД індивідуального об'єкта дослідження і розвиток «Наскрізного індивідуального проекту»

Мета роботи:

✓ закріпити навики створення проекту сховища даних у середовищі CASE-засобу Erwin Data Modeler пакета AllFussion Modeling Suite на основі вимірної моделі СД індивідуального об'єкта дослідження;

✓ розвиток «Наскрізного індивідуального проекту».

Завдання

1. Створити в CASE-засобі Erwin Data Modeler пакета AllFussion Modeling Suite новий проект сховища даних для OLAP-системи індивідуального об'єкта дослідження на основі ескізного концептуального проекту в процесі виконання таких етапів:

✓запуск пакета;

✓створення нової моделі сховища даних;

✓створення сутностей та робота з моделлю;

✓заповнення сутностей атрибутами;

✓побудова зв'язків;

✓встановлення параметрів таблиць сховища даних (*особливості вкладки Dimensional, вибір типу правил маніпулювання даними*);

✓визначення метаданих *(документування інформації про джерела даних, внесення додаткової інформації про методи).*

2. Формування копії екранних вікон побудованої моделі прототипу сховища даних на фізичному рівні у середовищі CASE-засобу ERwin Data Modeler за допомогою меню ***File/Print Setup та File/Print*** (*обов'язково з наявним текстовим блоком, у якому вказане ПІП студента та номер групи*).

3. Формування електронної версії згенерованого звіту.

4. Відповіді на контрольні питання.

Хід роботи

1. Проведення аналізу індивідуального об'єкта дослідження.

2. Виконання роботи для індивідуального об'єкта дослідження здійснюється відповідно до завдання-прототипу.

3. Виконати всі пункти, які передбачені у завданні.

4. Враховуючи індивідуальні особливості об'єкта дослідження, допускається розширення і поглиблення робіт «Індивідуального проекту» (може ускладнитись процес отримання зворотної інформації про виконання індивідуальних завдань та їхнього корегування).

Оформлення звіту

Звіт про виконану роботу оформляється згідно із «Загальними рекомендаціями до виконання лабораторних робіт та оформлення звіту» (додаток А: А.1, А.2).

Питання для самоперевірки

1. Охарактеризуйте відмінності між аналітичними і транс-акційними системами оброблення даних.

2. Надайте визначення СД та наведіть основні його відмінності від бази даних.

3. Надайте визначення корпоративного СД та вітрини (кіоску) даних.

4. Перелічіть види побудови архітектури сховищ даних.

5. Охарактеризуйте архітектуру СД на основі семантичної інтеграції предметних областей.

6. Охарактеризуйте архітектуру сховища даних із системою управління запитами до предметних областей.

7. Охарактеризуйте архітектуру монолітного сховища даних.

8. Охарактеризуйте архітектуру сховища даних на основі стандартного архіву даних.

9. Охарактеризуйте основні компоненти сховища даних.

10. Репозитарій метаданих, його роль та призначення.

Завдання для моделювання

1. Розробити аналітичні запити до СД вказаного об'єкта дослідження.

2. Розробити вимірну модель СД на створених аналітичних запитах.

3. Розробити SQL-запити на спроектованій багатовимірній моделі СД.

ВІДОМІСТЬ 1

розрахунку відрядної індивідуальної заробітної плати
щодо цеху _____ за _____ місяць 201__ р.
код та назва цеху

Таб.номер	Прізвище	Код деталі	Розряд	Професія		Кількість виготовлених деталей	Розцінка за деталь	Відроблений час	Сума заробітної плати
				Код	Назва				

ВІДОМІСТЬ 2

потреби в матеріалах на річну виробничу програму
за підприємством на 201__ р.

Код матеріалу	Назва матеріалу	Одиниці вимірювання	Код виробу	Назва виробу	Програма випуску виробів на рік	Норми витрат матеріалу на виріб	Потреба матеріалу на річну програму

ВІДОМІСТЬ 3

обліку руху матеріалів на складі ___ за ___ місяць
код та назва складу

Матеріал			Залишок вхідний		Надходження		Видатки		Залишок вихідний	
Код	Назва	Одиниця виміру	Кількість	Вартість	Кількість	Вартість	Кількість	Вартість	Кількість	Вартість

ВІДОМІСТЬ 4

обліку якості готової продукції щодо цеху ___ за __ місяць

код та назва цеху

Номер дільниці	Деталь		Операція		Вид браку		Причина (винуватець)		Кількість деталей	
	Код	Назва	Код	Назва	Код	Назва	Код	Назва	Придатних	Бракованих

ВІДОМІСТЬ 5

відвантаження готової продукції за _____ місяць

Код покупця	Назва покупця	Код продукції	Назва продукції	№ угоди на відвантаження	Кількість за угодою	Фактично відвантажено	Відхилення

РОЗДІЛ 6
АНАЛІЗ, УДОСКОНАЛЕННЯ ТА ІНТЕГРАЦІЯ МОДЕЛЕЙ ЗА ДОПОМОГОЮ CASE-ПАКЕТА ALLFUSSION MODELING SUITE CA

6.1. Структура та принцип роботи CASE-засобу AllFusion Data Model Validator (DMV) пакета AllFussion Modeling Suite CA

Для великих, тобто таких, що містять сотні таблиць, моделей даних істотною проблемою є пошук і виправлення помилок. Рішення цієї проблеми вручну досить трудомістке завдання, що може неприпустимо затягти виконання проекту.

AllFusion DMV – інструмент, що заснований на базі знань, дозволяє аналізувати структуру баз даних з метою виявлення недоліків і помилок проектування. AllFusion DMV доповнює функціональність AllFusion ERwin DM, автоматизуючи трудомістке завдання пошуку і виправлення помилок. AllFusion DMV може використовувати як джерело метаданих готову модель AllFusion ERwin DM, DDL – скрипт або провести зворотне проектування БД.

При проектуванні великих інформаційних систем (далі – ІС) ключовою проблемою є створення якісної документації за моделями. AllFusion PM і AllFusion ERwin DM дозволяють генерувати різноманітні звіти, які можуть бути використані для аналізу і документування моделей.

Звіти можуть бути експортовані в розповсюджені формати: текстовий, MS Office, HTML та ін.

Результати експорту можуть бути використані для створення звітів за допомогою засобів інших виробників, наприклад Crystal Reports.

AllFusion PM підтримує також експорт та імпорт моделі в текстовий файл формату IDL, що є стандартом для експорту й імпорту моделей IDEF0, дозволяє розробляти функціональні моделі одночасно інструментальними засобами різних виробників.

Створення сучасних ІС, заснованих на широкому використанні розподілених обчислень, об'єднанні традиційних і новітніх інформаційних технологій, вимагає тісної взаємодії всіх учасників проекту: менеджерів, бізнес-аналітиків і системних аналітиків, адміністраторів баз даних, розроблювачів. Для цього засоби моделювання і розробки, що використовуються на різних етапах і різними фахівцями, повинні бути об'єднані загальною системою організації спільної роботи.

Моделювання даних являє собою складне і відповідальне завдання, оскільки від якості моделі даних залежить у підсумку ефективність і продуктивність ІС.

Створення моделей для великих ІС вручну вже немислимо, для цього використовують CASE-засоби, такі як AllFusion ERwin DM тощо. Однак хоча застосування CASE-засобів і полегшує технічну роботу зі створення моделей, воно не оберігає від помилок і неточностей, які допускаються при моделюванні даних досить часто, особливо при створенні великих моделей.

Пошук і виправлення таких помилок без застосування спеціалізованих засобів за труднощами може перевершувати створення самої моделі, оскільки необхідно аналізувати модель, що містить тисячі таблиць і десятки тисяч колонок і зв'язків.

Для автоматизованого пошуку і виправлення помилок призначений інструмент, що входить у пакет AllFusion, AllFusion Data Model Validator (AllFusion DMV).

AllFusion DMV дозволяє автоматично аналізувати структуру баз даних з метою виявлення недоліків і помилок проектування. Принципова схема роботи AllFusion DMV показана на рис. 6.1.

AllFusion DMV може використовувати як джерело метаданих готову модель даних AllFusion ERwin DM, DDL-скрипт або провести зворотне проектування бази даних.

AllFusion DMV проводить діагностику імпортованої структури даних і пропонує рекомендації з її виправлення. Крім того, AllFusion DMV генерує скрипт, що дозволяє виправити деякі типи помилок.

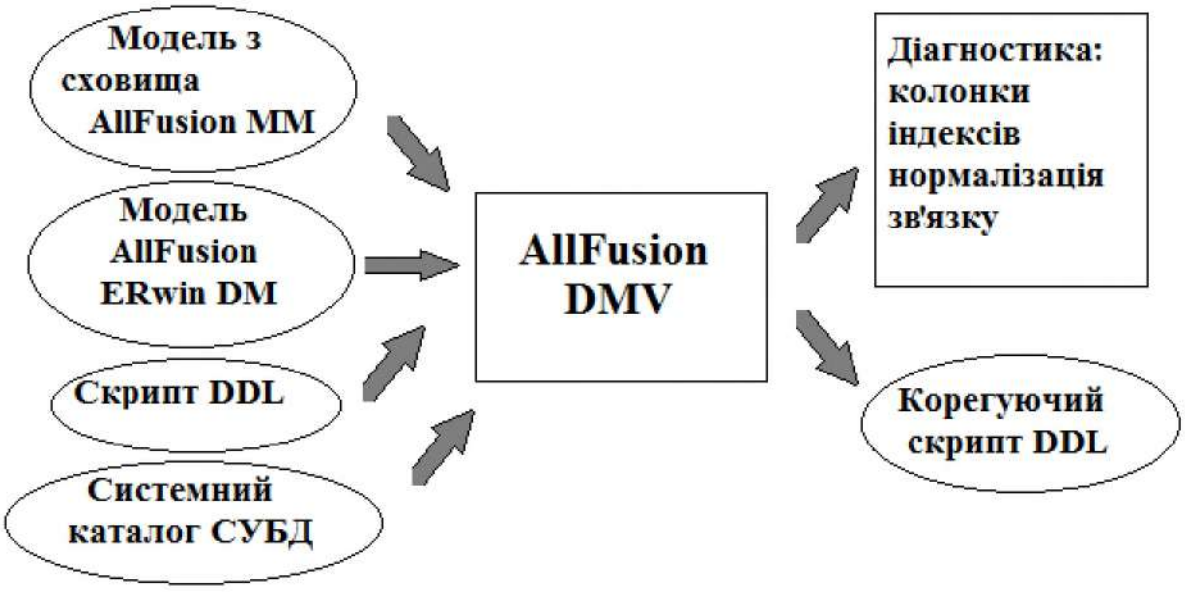


Рис. 6.1. Принципова схема роботи AllFusion DMV

6.2. Особливості помилок і недоліків моделювання колонок, ключів, індексів, обмежень, зв'язків таблиць БД

6.2.1. Помилки і недоліки моделювання колонок таблиць БД

Розглянемо помилки і недоліки моделювання даних, які дозволяє діагностувати AllFusion DMV.

СПІВРОБІТНИК

Номер співробітника: CHAR(18)
Прізвище:VARCHAR2(20) Ім'я:VARCHAR2(20) По батькові: CHAR(18) Номер_міста:VARCHAR2(20)

МІСТО

Номер міста: NUMBER
Назва:VARCHAR2(20)

Рис. 6.2. Приклад суперечливого визначення типу колонок

Суперечливе визначення типу. Назва колонки зустрічається в різних таблицях, але при цьому колонки (або їхні синоніми) мають різні визначення типу. Це може призвести до збереження правильних даних. Розглянемо приклад на рис. 6.2.

Тут колонки СПІВРОБІТНИК.Номер_міста і МІСТО. Номер_міста мають однакове найменування, але різне визначення типів – varchar2(20) і number; отже, колонки СПІВРОБІТНИК.

Номер_міста і МІСТО. Номер_міста можуть мати різні набори даних. Якщо не вживати спеціальних заходів, це може призвести до втрати посилкової цілісності, наприклад внесення співробітника з неіснуючого міста.

Суперечливе визначення груп колонок. Група колонок, що порушують першу нормальну форму, має суперечливе взаємне визначення (рис. 6.3).

ЗАМОВЛЕННЯ

Номер_замовлення: NUMBER
Виплата1: NUMBER Виплата2:VARCHAR(20) Виплата3: NUMBER

Рис. 6.3. Приклад суперечливого визначення групи колонок

Якщо вже в моделі є групи колонок, що порушують першу нормальну форму, то принаймні всі колонки групи повинні мати однакове визначення типу. Так, у прикладі на рис. 6.3 є група колонок Виплата1, Виплата2, Виплата3, причому тип колонки Виплата2 varchar(20) відрізняється від типу колонок Виплата1, Виплата3 (number). Для виправлення цієї помилки потрібно привести тип колонки Виплата2 у відповідність із типом інших колонок групи.

Таблиці з неунікальними іменами. Ім'я таблиці повинне бути унікальним у моделі. Для виправлення помилки необхідно видалити або перейменувати одну з таблиць.

Таблиця не має колонок. Таблиця повинна мати хоча б одну колонку.

Для колонки типу VARCHAR не визначена довжина (для СУБД Oracle). При генерації схеми СУБД Oracle не дозволить створити таку колонку.

Суперечливе значення за замовчуванням. Значення колонки за замовчуванням конфліктує з типом даних колонки або зі значенням, визначеному в обмеженні CHECK. Можлива також ситуація, коли значення колонки за замовчуванням конфліктує з обмеженням CHECK або з визначенням даних.

Колонки, які представляють ті самі дані, повинні мати ті самі типи даних. У прикладі на рис. 6.4 колонки КЛІЄНТ.Номер_клієнта і ЗАМОВЛЕННЯ.Номер_клієнта мають різні типи (CHAR і VARCHAR).

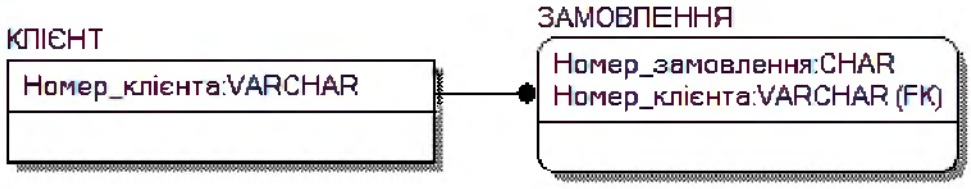


Рис. 6.4. Приклад нечіткого визначення типів колонок

Хоча така невідповідність не приведе до втрати посилкової цілісності, як у прикладі на рис. 6.4, колонки повинні бути приведені до однакового типу.

Імена колонок збігаються з резервованими словами ANSI SQL. Це може призвести до конфліктів при генерації схеми СУБД. Такі колонки повинні бути перейменовані.

Таблиця не має зв'язків. Іноді таблиця без зв'язків створюється навмисне, наприклад службові таблиці або спеціальні довідники. Але якщо такі таблиці відображають предметну область, вони повинні бути ретельно проаналізовані. У прикладі на рис. 6.5 таблиці КЛІЄНТ і ЗАМОВЛЕННЯ не пов'язані, але такий зв'язок передбачається на полях КЛІЄНТ.Номер_клієнта і ЗАМОВЛЕННЯ.Клієнт. Це помилка, і вона повинна бути виправлена, має бути встановлений зв'язок між КЛІЄНТ і ЗАМОВЛЕННЯ.

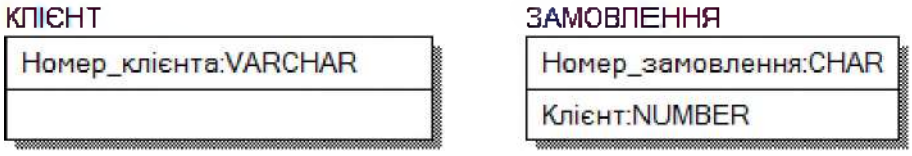


Рис. 6.5. Таблиці без зв'язків

6.2.2. Помилки і недоліки моделювання ключів

Колонки потенційних ключів допускають значення NULL. Оскільки кожний екземпляр сутності повинен однозначно ідентифікуватися, то не допускається значення NULL для первинних і альтернативних ключів.

Неправильно визначений альтернативний ключ. Таблиця має сурогатний ключ, тобто ключ, що не має відповідності в предметній області, наприклад порядковий номер послідовності. У цьому випадку доцільно створити альтернативний ключ.

У прикладі на рис. 6.6 ліва таблиця СПІВРОБІТНИК має сурогатний ключ Номер_співробітника, але не має альтернативного ключа. Це може призвести до внесення імені тієї самої людини з різними номерами. Для виключення такої ситуації доцільно створити альтернативний ключ на колонках Прізвище, Ім'я, По_батькові, Дата_народження (рис. 6.6, праворуч). AllFusion ERwin DM згенерує в цьому випадку на основі альтернативного ключа унікальний індекс.



Рис. 6.6. Визначення альтернативного ключа

Таблиці не мають потенційних ключів. Таблиці не мають потенційних ключів, тобто первинного ключа, унікального обмеження або унікального індексу, що гарантує унікальність рядка. Це може призвести до неправильного виконання запитів. Кожна таблиця повинна мати або первинний ключ, або унікальний індекс, або унікальне обмеження. Якщо потенційного ключа немає, то СУБД не в змозі перевірити в момент вставки запису, чи був внесений такий же рядок з тими ж значеннями колонок раніше.

Якщо таблиця не використовується для спеціальних цілей, наприклад для тестування або тимчасового зберігання звітів, для неї повинен бути створений первинний ключ.

Первинний ключ, що має колонку типу дійсного числа. Зазвичай колонки такого типу не застосовуються у складі первинного ключа, за винятком спеціальних випадків, наприклад додатків, що працюють із графікою.

Непотрібний зовнішній ключ. У прикладі на рис. 6.7 посилальна цілісність підтримується зв'язками R1 (КРАЇНА – РЕГІОН) і R2 (КРАЇНА – МІСТО), а зв'язок R3 (РЕГІОН – МІСТО)повинен бути вилучений.

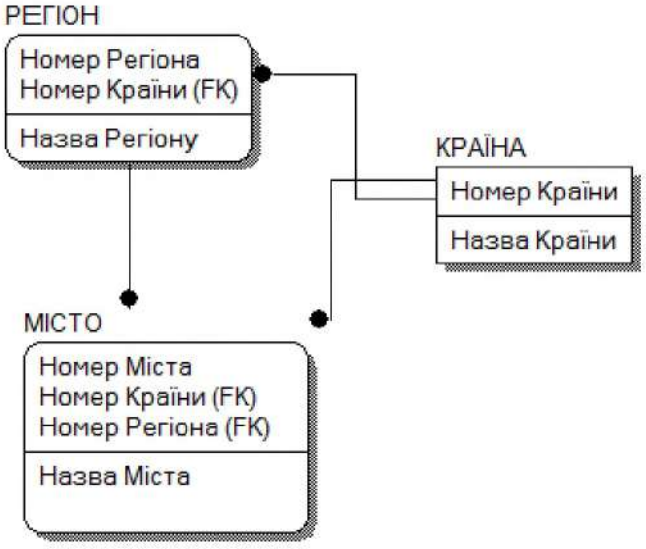


Рис. 6.7. Непотрібний зовнішній ключ

Довжина первинного ключа досить велика. Максимальна загальна довжина колонок, що входять у первинний ключ, перевершує значення, заздалегідь визначеної користувачем AllFusion DMV.

Первинні ключі з досить великою кількістю колонок. Такі ключі сповільнюють операції вставки (*update*) і вибірки (*select*) з об'єднанням таблиць. У цьому випадку доцільно замінити природний первинний ключ на сурогатний, як показано на рис. 6.8 і 6.9.

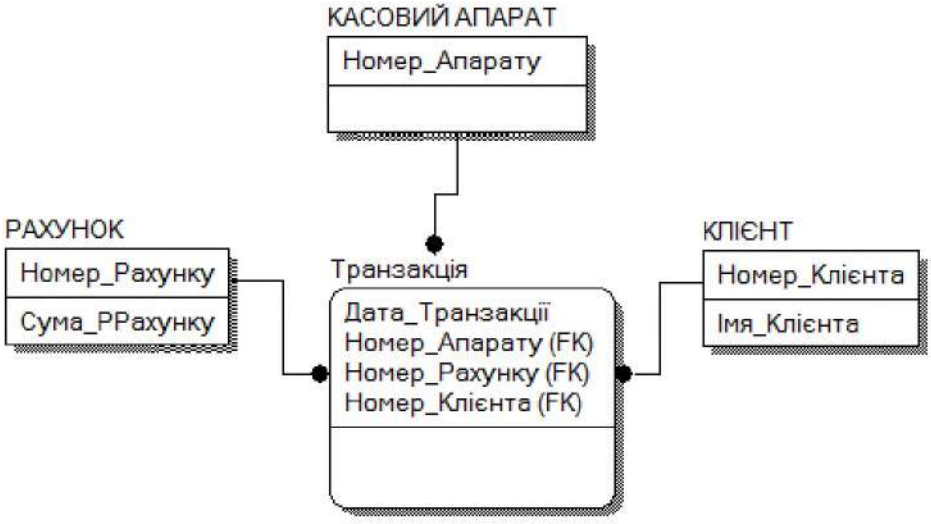


Рис. 6.8. Первинний ключ з досить великою кількістю колонок

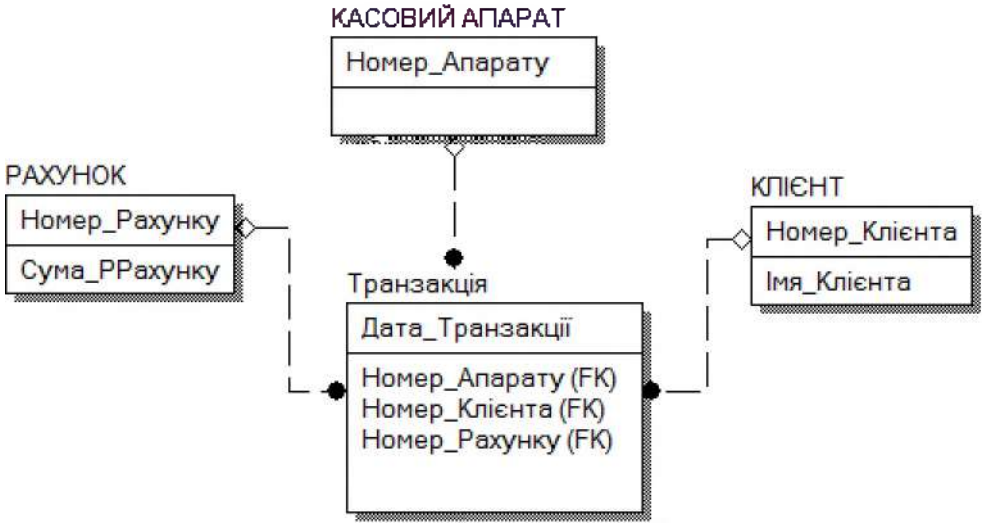


Рис. 6.9. Сурогатний первинний ключ

Альтернативні ключі з колонками, що допускають значення NULL. Згідно з реляційною теорією потенційні ключі не повинні містити колонок, що допускає значення NULL, тому альтернативні і первинні ключі не повинні мати таких колонок.

Помилка зовнішнього ключа. У моделі AllFusion ERwin DM на логічному рівні атрибут зовнішнього ключа позначений як *Logical Only*. У цьому випадку зовнішній ключ генеруватися не буде. Для виправлення помилки опція *Logical Only* повинна бути виключена.

6.2.3. Помилки і недоліки моделювання індексів

Аномалії у визначенні індексів. Виправлення цього типу помилки приводить до збільшення продуктивності ІС. Цей тип помилок може включати такі 3 різновиди.

Індекс функціонально еквівалентний первинному ключу. Неунікальний індекс еквівалентний первинному ключу, тобто містить ті ж самі колонки, але в іншому порядку. У цьому випадку або первинний ключ не гарантує унікальності рядка, або індекс оголошений як унікальний, що, можливо, не потрібно.

Оскільки первинний ключ унікальний, будь-яка перестановка колонок ключа повинна бути унікальна, тому індекс, функціонально еквівалентний первинному ключу, повинен бути унікальний і

ймовірно не потрібний, або первинний ключ визначений на колонках, які не забезпечують унікальності рядка. Розглянемо приклад.

```
create table РЕЙС
(
Аеропорт_вильоту varchar (3) not null,
Аеропорт_прибуття varchar (3) not null,
Номер_рейса number not null,
Час_прибуття number (6,2)
);

alter table РЕЙС
add constraint PK_ РЕЙС
primary key (Аеропорт_вильоту, Аеропорт_прибуття,
Номер_рейса);
create index NDX_РЕЙС
on РЕЙС (Аеропорт_прибуття, Аеропорт_вильоту, Номер_рейса);
```

Якщо сполучення колонок, що входять у первинний ключ (Аеропорт_вильоту, Аеропорт_прибуття, Номер_рейса), є унікальним, то і сполучення Аеропорт_прибуття, Аеропорт_вильоту, Номер_рейса повинно бути унікальним.

Для того щоб знайти таку помилку, потрібно переконатися, що колонки, що входять у первинний ключ, дійсно забезпечують унікальність рядка. Якщо це так, то індекс повинен бути унікальним.

Якщо індекс створений для того, щоб витягати рядки у визначеному порядку, можливо він містить більше колонок, ніж потрібно.

Якщо у прикладі варто витягати рядки, відсортовані за колонкою Аеропорт_прибуття, то колонки Аеропорт_вильоту і Номер_рейса в індексі NDX_РЕЙС зайві.

Якщо первинний ключ не забезпечує унікальності рядка таблиці, необхідно переглянути колонки-кандидати на первинний ключ і доповнити первинний ключ так, щоб він забезпечував унікальність.

Індекс містить супернабір первинного ключа. Індекс містить супернабір первинного ключа, тобто має всі колонки первинного ключа і принаймні ще одну колонку, але при цьому не є унікальним. Оскільки первинний ключ унікальний, будь-який супернабір повинен бути також унікальний. Або індекс повинен бути зроблений унікальним, або в дійсності первинний ключ не забезпечує унікальності й повинен бути переглянутий.

Розглянемо приклад.

create table ЗАМОВЛЕННЯ
(Номер_замовлення number not null,
Дата_замовлення date not null);
alter table ЗАМОВЛЕННЯ
add constraint PK_замовлення
primary key (Номер_замовлення);
create index NDX_замовлення
on ЗАМОВЛЕННЯ (Дата_замовлення, Номер_замовлення);

Якщо Номер_Замовлення унікальний, то сполучення колонок Дата_замовлення, Номер_Замовлення теж повинне бути унікальним.

Для того щоб виявити таку помилку, потрібно переконатися, що колонки, що входять у первинний ключ, дійсно забезпечують унікальність рядка. Якщо це так, то індекс повинен бути зроблений унікальним.

Можливо, такий індекс створений для сортування рядків за колонкою Дата_замовлення.

У цьому випадку колонка Номер_замовлення повинна бути вилучена з індексу.

Якщо ж індекс створений для сортування за датою замовлення, а потім за номером, то він побудований правильно.

Якщо індекс створений для забезпечення унікальності, то він повинен бути вилучений, оскільки первинний ключ уже забезпечує унікальність.

Якщо первинний ключ не забезпечує унікальності, необхідно переглянути склад первинного ключа.

Індекс містить супернабір іншого унікального індексу. Індекс містить супернабір іншого унікального індексу, тобто має

всі колонки унікального індексу і принаймні ще одну колонку або обмеження. Цей індекс повинен бути унікальним і повинен бути перевизначений як унікальний.

Розглянемо приклад.

create table КЛІЄНТ
(
Номер_Клієнта number not null primary key,
Ім'я_Клієнта varchar (80),
Місто_Клієнта varchar (80)
);
create unique index NDX_КЛІЄНТ_01
on КЛІЄНТ (Ім'я_Клієнта);
create index NDX_КЛІМЕНТ_02
on КЛІЄНТ (Місто_Клієнта, Ім'я_Клієнта);

Якщо Ім'я_Клієнта унікальне, то сполучення Місто_Клієнта, Ім'я_Клієнта теж повинне бути унікальним. Індекс NDX_КЛІНЕНТ_02 повинен бути перевизначений як унікальний.

Таблиці, що не мають кластеризованих індексів. Кластеризовані індекси істотно підвищують продуктивність при вибірці даних. У багатьох випадках доцільно створювати первинні ключі й унікальні індекси як кластеризовані. Однак потрібно пам'ятати, що при інтенсивній вставці даних наявність кластеризованого індексу може призвести до падіння продуктивності.

Непотрібні індекси. Непотрібні індекси призводять до падіння продуктивності ІС і повинні бути вилучені.

Індекс, побудований на колонці (*або групі колонок*), що приймає тільки одне значення. Такий індекс не буде ефективний ні при вибірці (витягатимуться усі рядки таблиці), ні при сортуванні і повинен бути вилучений.

Індекс, колонки якого цілком включені в інший індекс.

Розглянемо приклад.

create table ЗАМОВЛЕННЯ
(Номер_замовлення number not null primary key,
Номер_Клієнта number not null,
Дата_замовлення date not null);
create unique index NDX_замовлення_01

on ЗАМОВЛЕННЯ (Номер_Клієнта, Дата_замовлення);
create index NDX_замовлення__02
on ЗАМОВЛЕННЯ (Номер_Клієнта);

Якщо рядок ідентифікується за індексом NDX_замовлення_01, то він ідентифікується і за індексом NDX_замовлення_02; отже, індекс NDX_замовлення_02 повинен бути вилучений (якщо тільки він не використовується для спеціальних цілей).

Індекс, колонки якого цілком включені в первинний ключ. Індекс, колонки якого цілком включені в первинний ключ, повинен бути вилучений за тих самих причин, що й індекс, колонки якого цілком включені в інший індекс (*якщо тільки він не використовується для спеціальних цілей)*.

Відсутність індексів. Розглянемо приклад на рис. 6.10. Таблиці ЗАМОВЛЕННЯ і КЛІЄНТ денормалізовані.



Рис. 6.10. Відсутність індексів на зовнішніх ключах

Таблиця КЛІЄНТ пов'язана з таблицею ЗАМОВЛЕННЯ за колонкою Номер_Клієнта. Якщо в таблиці ЗАМОВЛЕННЯ на колонку Номер_Клієнта не буде створений індекс, то при видаленні рядка в таблиці КЛІЄНТ для забезпечення посилкової цілісності СУБД перевірятиме наявність рядків у таблиці ЗАМОВЛЕННЯ з відповідним значенням поля ЗАМОВЛЕННЯ.Номер_Клієнта. Якщо по цьому полю не побудований індекс, продуктивність системи може різко впасти.

З іншого боку, якщо дані в батьківській таблиці міняються рідко (наприклад, у таблиці МІСТО), відсутність індексу на зовнішньому ключі (КЛІЄНТ.Номер_Міста) припустимо.

Таблиці із занадто великою кількістю індексів. Велика кількість індексів сповільнює операції вставки і відновлення (*insert і update*). Конкретне число індексів, що припустимо для

цієї таблиці, залежить від багатьох параметрів і повинно визначатися автором моделі.

Індекси із занадто великою кількістю колонок. Такі індекси сповільнюють операції вставки (*update*). Припустиме число колонок в індексі також повинне визначатися автором моделі.

Довжина індексу досить велика. Максимальна загальна довжина колонок, що входять в індекс, перевершує значення, заздалегідь визначеного користувачем AllFusion DMV. Досить велика довжина індексу може призвести до його неприпустимого розміру.

Унікальний індекс еквівалентний первинному ключу. Унікальний індекс еквівалентний первинному ключу, тобто включає ті ж колонки, що й первинний ключ, але в іншому порядку. Якщо цей індекс створювався для забезпечення унікальності рядка, то він повинен бути вилучений, оскільки первинний ключ уже гарантує унікальність. Якщо такий індекс створювався для доступу до певного рядка або групи рядків чи для сортування рядків, він повинен бути модифікований. Досить імовірно, що з такого індексу можна виключити колонки. Це не тільки підвищить продуктивність при вибірці даних, але і заощадить дисковий простір.

Неправильний тип колонок, що входять в індекс. В індекс не можуть входити колонки типів BLOB, IMAGE, MEMO тощо.

6.2.4. Помилки і недоліки моделювання обмежень

Різні визначення обмеження CHECK. Ті самі колонки, що описані у двох або більше таблицях, мають різні обмеження CHECK. Це може призвести до неправильного виконання запитів, особливо при виконанні операції об'єднання (join). Якщо колонка має обмеження *CHECK* в одній таблиці, аналогічні колонки в інших таблицях повинні мати ті ж самі обмеження.

Непотрібні обмеження CHECK. Обмеження CHECK вважаються непотрібними, якщо вони або перевіряють тільки одну умову в логічному «або», або перевіряють зовнішній ключ у дочірній таблиці замість того, щоб перевіряти первинний ключ у батьківській.

Розглянемо приклад перевірки тільки однієї умови в логічному «або»:

CHECK (Статус IN ('ДОЗВОЛИТИ'))

Якщо у всіх рядках таблиці колонка Статус приймає тільки одне значення – «ДОЗВОЛИТИ», то або така колонка не потрібна і її варто видалити разом з обмеженням CHECK, або в дійсності колонка Статус може приймати й інші значення, тому обмеження CHECK потрібно перевизначити.

Інший приклад:

create table КАТЕГОРІЯ
(Категорія varchar (12) not null primary key);
create table КЛІЄНТ
(Номер_Клієнта number not null primary key,
Ім'я_Клієнта varchar (80),
Категорія varchar (12) not null
constraint VALID_Категорія check
(Категорія in ('Звичайний', 'Постійний', 'VIP')));

Припустимо, з таблиці КАТЕГОРІЯ видаляється рядок, наприклад значення 'Звичайний'.

При цьому в таблиці КЛІЄНТ рядки з таким значенням залишаються. Для забезпечення цілісності даних при кожному редагуванні таблиці КАТЕГОРІЯ необхідно редагувати безліч рядків таблиці КЛІЄНТ і змінювати обмеження VALID_Категорія.

Тому обмеження VALID_Категорія в дочірній таблиці на колонку КЛІЄНТ.Категорія повинне бути вилучено й аналогічне обмеження повинно бути створене в батьківській таблиці КАТЕГОРІЯ.Категорія.

6.2.5. Помилки і недоліки моделювання зв'язків

Некоректний ієрархічний рекурсивний зв'язок

Розглянемо приклад.

create table СПІВРОБІТНИК
(
Номер_Співробітника number not null primary key,
Ім'я_Співробітника varchar (80),
Номер_Керівника number not null
);

Тут інформація про керівника співробітника зберігається в тій же таблиці СПІВРОБІТНИК, що й інформація про співробітника.

За допомогою ієрархічної рекурсії будується піраміда підпорядкованості співробітників, і в цієї піраміди повинна бути вершина, тобто повинен знайтися співробітник, у якого немає керівника.

У цьому прикладі таку піраміду побудувати не можна, оскільки колонка Номер_Керівника має ознаку *not null*. Для виправлення помилки необхідно допустити для колонки Номер_Керівника значення NULL.

Невизначені зв'язки. Колонки в різних таблицях мають однакове ім'я і визначення, але таблиці не позв'язані одна з одною. Такі таблиці повинні бути ретельно вивчені, і якщо бізнес-правила вимагають встановити зв'язок між такими таблицями, він повинен бути створений.

Нескінченні цикли. Розглянемо приклад на рис. 6.11. Припустимо, для всіх зовнішніх ключів моделі встановлене правило Delete Cascade.

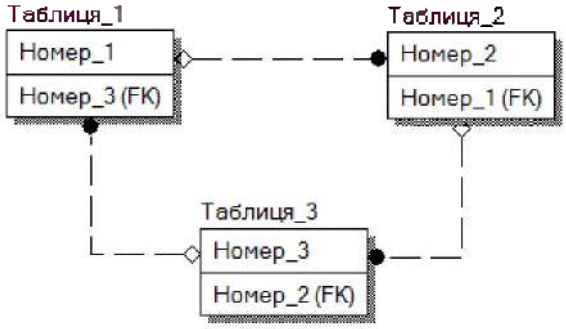


Рис. 6.11. Нескінченний цикл

Якщо видаляється рядок з таблиці Таблиця_1, повинні видалятися всі відповідні рядки з таблиці Таблиця_2, при видаленні рядка з Таблиця_2 видаляються всі відповідні рядки з Таблиця_3, при видаленні рядка з Таблиця_3 видаляються всі відповідні рядки з Таблиця_1 і так нескінченно. Спроба видалення лише одного рядка з будь-якої таблиці може призвести до знищення всієї інформації, що зберігається в базі даних.

Взаємно перетинаючі зв'язки. Розглянемо приклад на рис. 6.12.

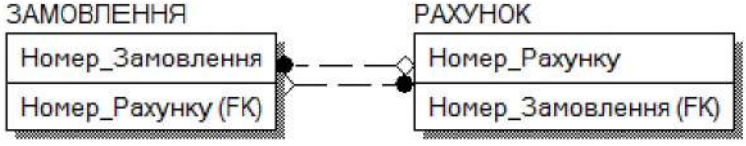


Рис. 6.12. Взаємно перетинаючі зв'язки

Припустимо, у таблицях немає жодного рядка. Потрібно виконати команду вставки:

Insert into ЗАМОВЛЕННЯ (Номер_замовлення, Номер_рахунку) values (1,1)

При вставці рядка в таблиці ЗАМОВЛЕННЯ СУБД перевірить, чи є відповідний рядок у таблиці РАХУНОК (зі значенням Номер_рахунку=1).

Але відповідний рядок у таблицю РАХУНОК вставити не можна, оскільки не існує відповідного рядка в таблиці ЗАМОВЛЕННЯ. У результаті в жодну із двох таблиць неможливо вставити дані.

Множинні шляхи. Розглянемо приклад на рис. 6.13.



Рис. 6.13. Множинні шляхи

Таблиця КЛІЄНТ пов'язана з таблицею ПОЗИЦІЯ_ЗАМОВЛЕННЯ як безпосередньо, так і через таблицю ЗАМОВЛЕННЯ.

Хоча структура даних на рис. 6.13 не може вважатися нормалізованою, такі зв'язки не обов'язково є помилкою.

Якщо необхідно одержати позиції замовлення цього клієнта, у запиті не потрібно позв'язувати таблицю ЗАМОВЛЕННЯ, що істотно скоротить час виконання запиту.

З іншого боку, оскільки таблиці ЗАМОВЛЕННЯ й ПОЗИЦІЯ_ЗАМОВЛЕННЯ не пов'язані за колонкою Номер_клієнта, можлива ситуація, коли в рядках, що відповідають тому самому замовленню в таблицях ЗАМОВЛЕННЯ і ПОЗИЦІЯ_ЗАМОВЛЕННЯ, будуть внесені різні значення колонки Номер_клієнта, що є порушенням посилкової цілісності.

Конфлікт правила посилкової цілісності Delete Cascade/Restrict. Розглянемо приклад на рис. 6.14.



Рис. 6.14. Конфлікт правила посилкової цілісності Delete Cascade/Restrict

У моделі створені два зовнішніх ключі:

FK1: Parent КЛІЄНТ Child ЗАМОВЛЕННЯ (Cascade)

FK2: Parent ЗАМОВЛЕННЯ Child ПОЗИЦІЯ_ЗАМОВЛЕННЯ (Restrict)

Тоді при видаленні рядка з таблиці КЛІЄНТ буде зроблена спроба видалення всіх відповідних рядків з таблиці ЗАМОВЛЕННЯ, водночас із таблиці ЗАМОВЛЕННЯ не можна буде видалити рядок, якщо їй відповідає хоча б один рядок у таблиці ПОЗИЦІЯ_ЗАМОВЛЕННЯ.

6.3. Технологія пошуку і виправлення помилок моделювання БД за допомогою AllFusion DMV

Інтерфейс AllFusion DMV. Інтерфейс AllFusion DMV досить простий і інтуїтивно зрозумілий. Робочий простір містить два вікна (рис. 6.15).

Праве вікно слугує для візуального відображення моделі даних, ліве містить три вкладки, дві з них (Relationships і Tables) показують ієрархічний список об'єктів моделі, третя (Diagnostics) показує список помилок моделювання.

У верхній частині робочого простору розташовується панель керування. Призначення кнопок панелі керування показане в табл. 6.1.

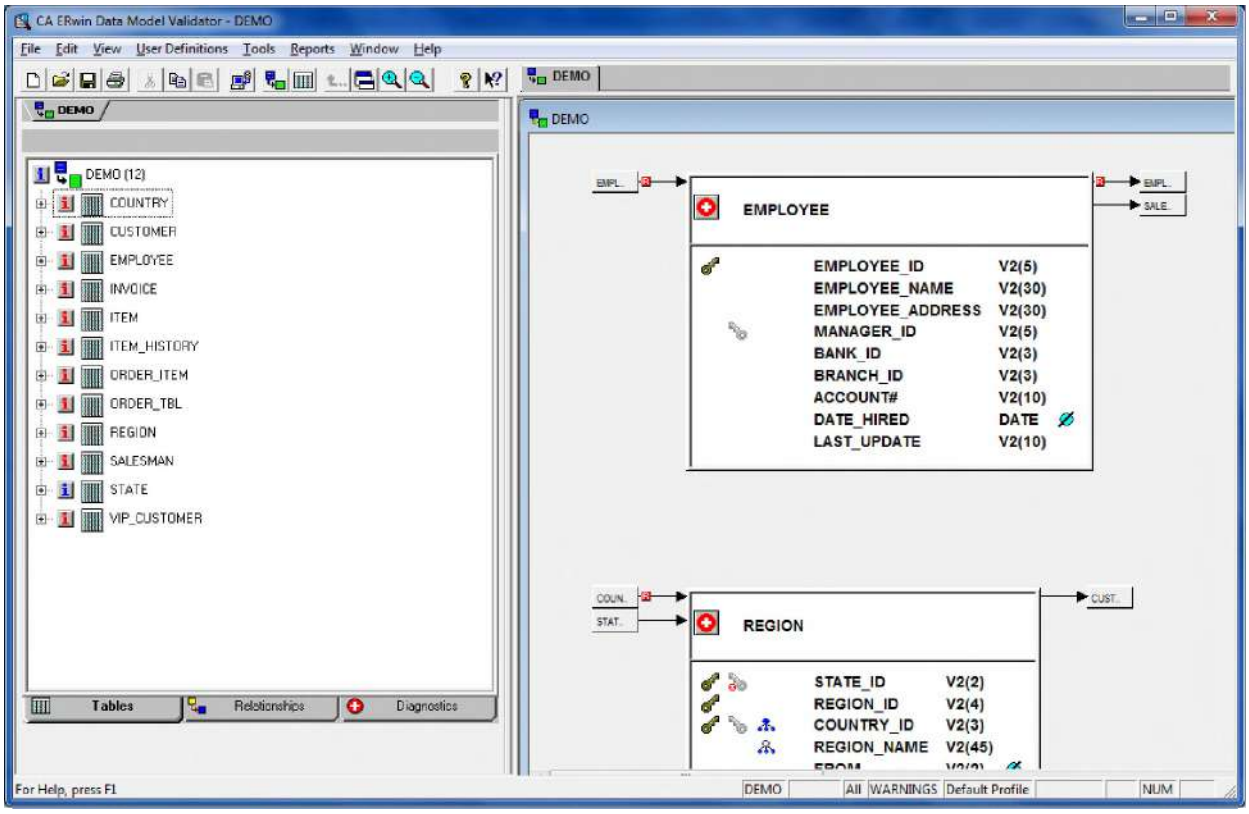


Рис. 6.15. Робочий простір AllFusion DMV

Таблиця 6.1

Панель інструментів AllFusion DMV

Кнопки *Призначення*

Створення, відкриття й збереження проекту ***AllFusion DMV***



Друк проекту або моделі

«Вирізати» дані у кліпборд

Копіювати дані у кліпборд

Вставити дані із кліпборда

Відновлення моделі

Перемикання правого вікна робочого простору на перегляд моделі

Перемикання правого вікна робочого простору на перегляд таблиць



Повернення до попереднього режиму перегляду

Перемикання між режимами перегляду «на повний екран» і звичайним

Збільшення масштабу перегляду

Зменшення масштабу перегляду

Створення проекту AllFusion DMV. Для створення нового проекту варто клацнути по кнопці створення проекту на панелі інструментів або обрати пункт меню File/New. З'являється діалог Select Project Type (рис. 6.16).

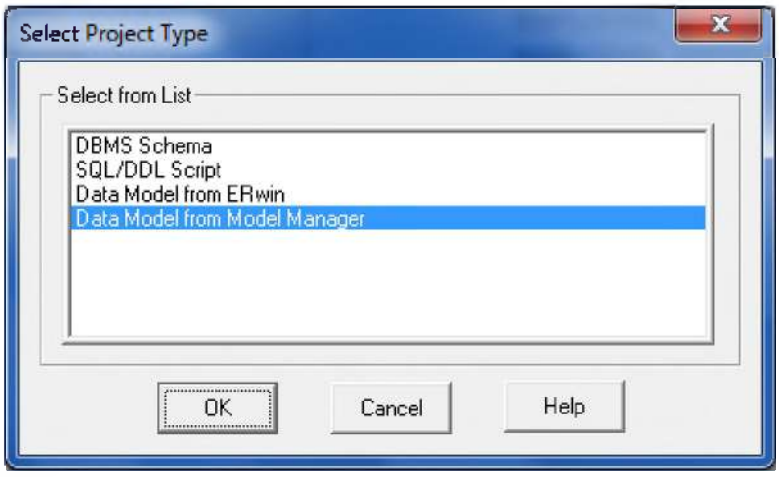


Рис. 6.16. Діалог Select Project Type AllFusion DMV

У діалозі Select Project Type можна обрати джерело метаданих майбутнього проекту. Як було зазначено вище, в AllFusion DMV можна використовувати 4 джерела метаданих:

1. Системний каталог СУБД.
2. DDL-скрипт
3. Модель AllFusion ERwin DM.
4. Модель AllFusion ERwin DM у сховище AllFusion MM.

CASE-засіб AllFusion DMV підтримує роботу з таким СУБД:

а) ORACLE; б) DB2 Mainframe DB2 UDB версій 4, 5, 6 і 7; в) DB2 Client/Server DB2 UDB версій 5, 6 і 7; г) Microsoft SQL Server; д) Sybase.

Це означає, зокрема, що, як джерело, можна використовувати тільки моделі AllFusion ERwin DM, на фізичному рівні яких як сервер зазначена одна з перерахованих СУБД. Зворотне проектування із системного каталогу або DDL-скрипта можливе теж тільки із цих СУБД.

Після вибору моделі даних з'являється діалог Select Tables for Model, у якому можна відібрати таблиці джерела для включення у проект AllFusion DMV.

Після імпорту моделі у вкладках Tables і Relationships відображаються об'єкти моделі. Вкладка Tables містить ієрархічний список (рис. 6.17), перший рівень якого містить список відкритих в AllFusion DMV моделей, другий – список таблиць, інші – список колонок і первинних ключів.

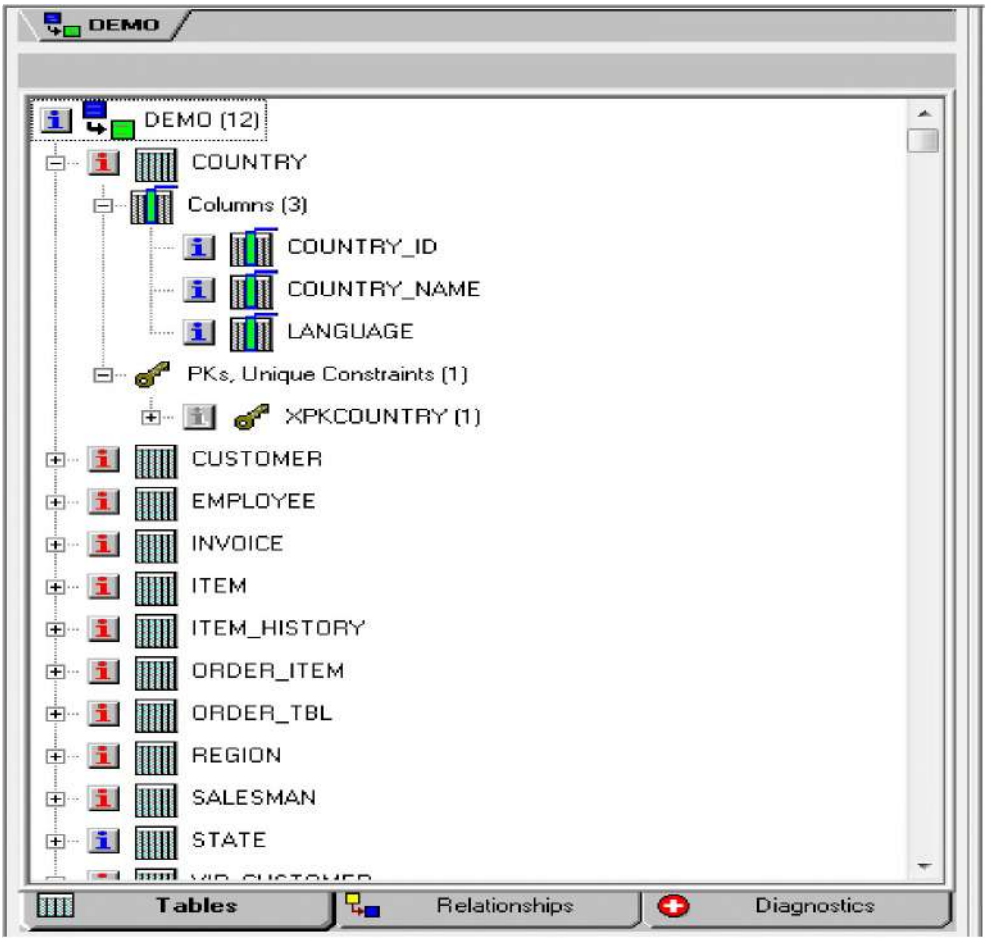


Рис. 6.17. Вкладка Tables

Кнопка, що розміщена ліворуч від імені колонки, викликає діалог з докладною інформацією про об'єкт. Докладний опис помилки можна одержати, клацнувши лівою кнопкою миші по кнопці ліворуч від імені помилки.

Вкладка Relationships містить інформацію про зв'язки (рис. 6.18). Перший рівень списку на вкладці Tables містить список моделей, другий – список таблиць, третій – список зв'язків «один до багатьох» (Parent to Child), четвертий – список ключів і зв'язків нижнього рівня.

Вкладка Diagnostics містить інформацію про помилки і недоліки моделі даних (рис. 6.19).

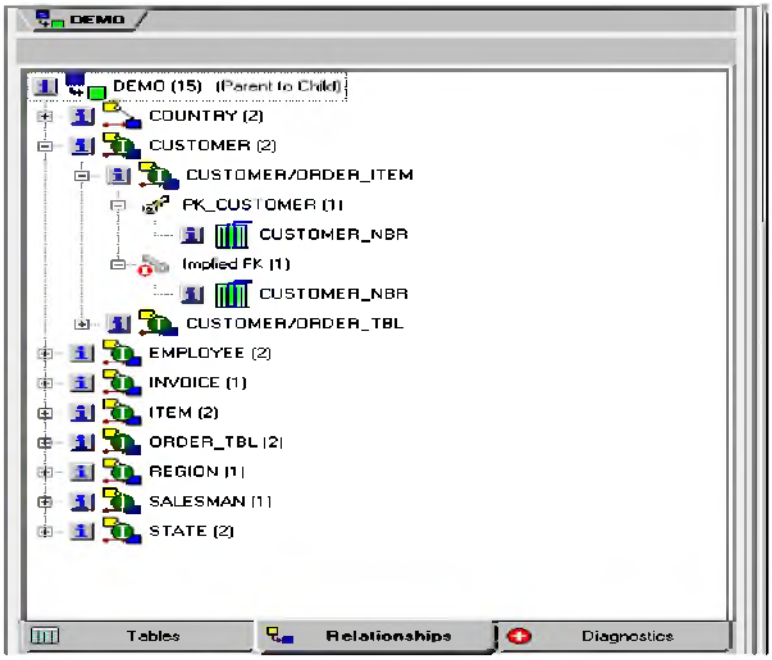


Рис. 6.18. Вкладка Relationships

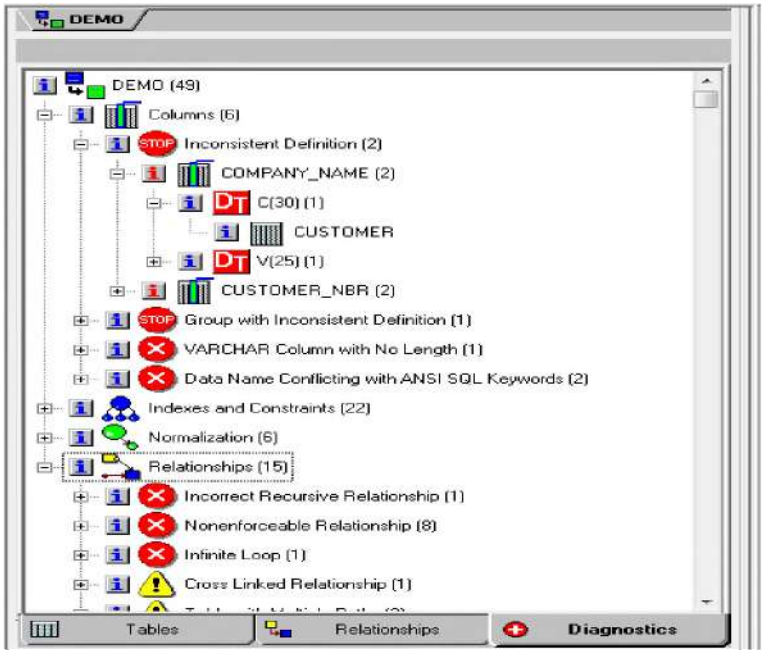


Рис. 6.19. Вкладка Diagnostics

Категорії помилок. Помилки об'єднані в 4 категорії. У першу категорію (Columns) входять помилки проектування колонок.

Друга категорія поєднує помилки проектування індексів і обмежень (Indexes and Constraints).

Третя категорія поєднує помилки нормалізації (Normalization). Перебувають некоректно визначені функціональні залежності і помилки першої, другої і третьої нормальних форм.

У четверту категорію входять помилки зв'язків (Relationships).

AllFusion DMV знаходить усі помилки, що розглянуті вище, причому існує можливість налаштовування діагностики. Налаштовування діагностики дозволяє включити в аналіз або виключити з аналізу ті чи інші помилки.

Так, наприклад, при перевірці структури сховищ даних перевірку нормалізації можна відключити.

Для завдання параметрів налаштовування варто перейти в меню Tools/Preferences. З'являється діалог User Preferences. Вкладка Diagnostic Setting (рис. 6.19) містить, крім загальних опцій діагностики, кнопку Select Individual Diagnostics. Клацання по цій кнопці викликає діалог Diagnostic Messages (рис. 6.20).

Діалог Diagnostic Messages містить 4 вкладки, що відповідають чотирьом категоріям помилок. На кожній вкладці втримується список діагностованих AllFusion DMV помилок і вікна вибору, що дозволяють відключити розпізнавання тієї або іншої помилки при діагностиці (рис. 6.21).

У правому вікні AllFusion DMV відображаються звіти, у тому числі звіт про помилки моделі або графічне подання моделі (рис. 6.22). За замовчуванням зв'язок таблиць показується в нетрадиційному синтаксисі у вигляді посилань, однак існує можливість подання зв'язків у більш звичному стилі – у вигляді ліній.

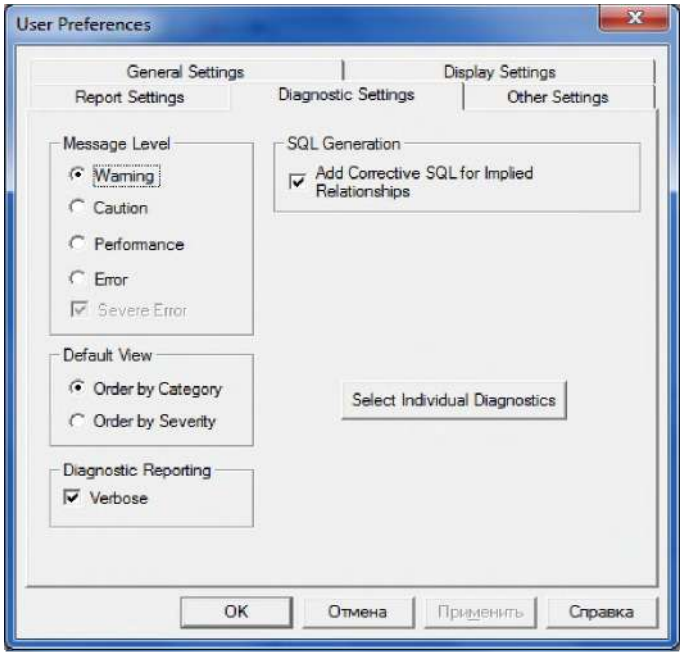


Рис. 6.20. Діалог User Preferences

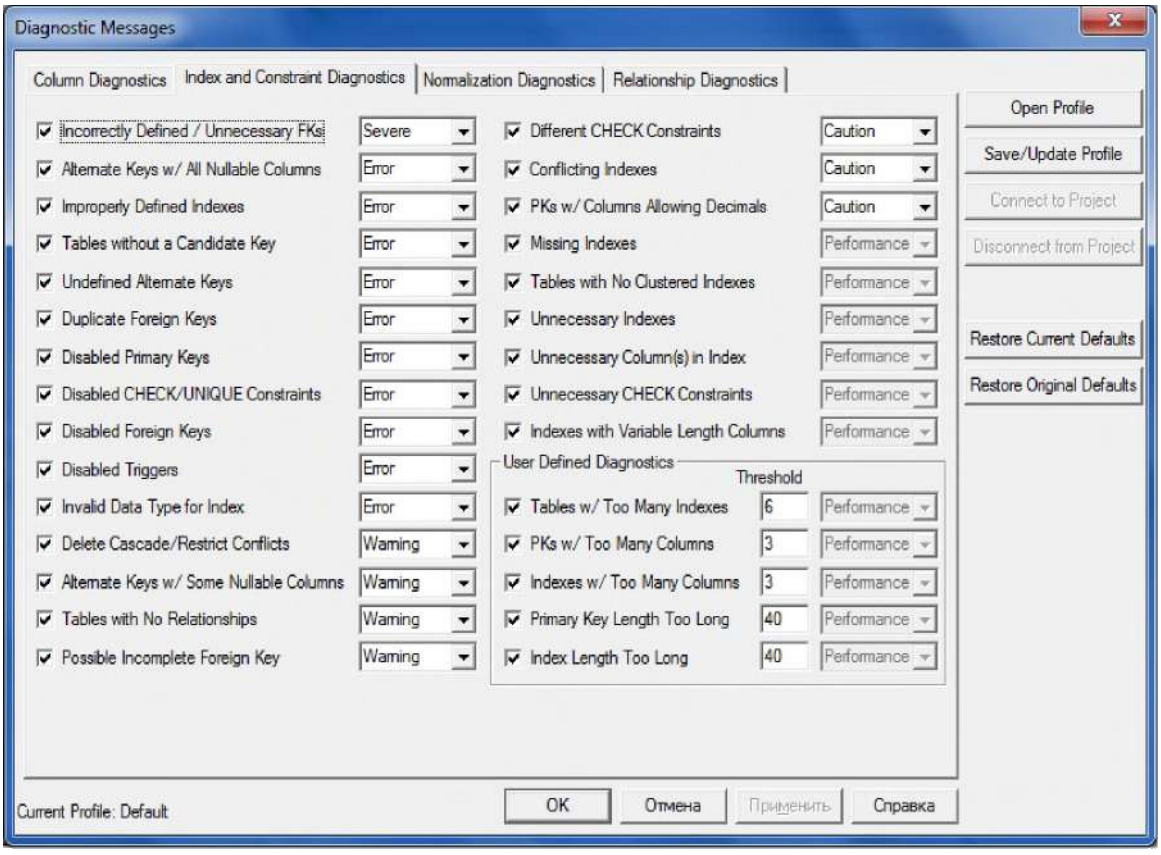


Рис. 6.21. Діалог Diagnostics Messages

Посилання на діаграмі EMPL → слугують для навігації – клацання по посиланню перемикає вікно на дочірню або батьківську таблицю.

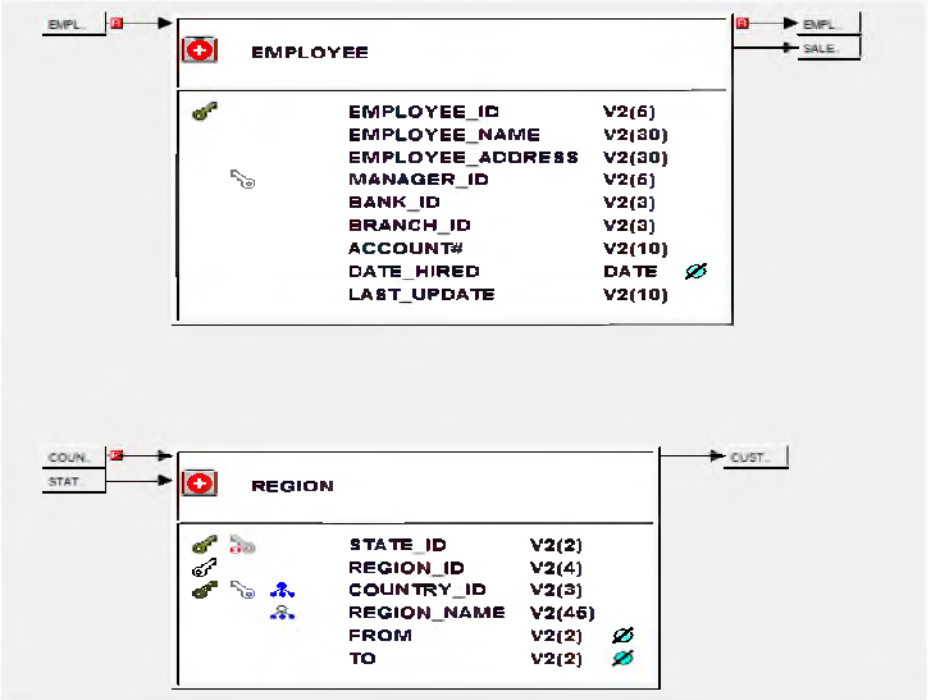


Рис. 6.22. Графічне подання моделі в AllFusion DMV

Діаграма таблиць досить інформативна – у графічному вигляді показуються не тільки таблиці й колонки, але й обмеження, індекси, знайдені помилки.

Можна викликати діалог Diagnostics (рис. 6.21), у якому показуються помилки моделі, пов'язані з таблицею.

З контекстного меню (Legend) можна легко викликати довідку з системи позначень діаграми (рис. 6.23).

Результатом діагностики помилок може стати звіт або SQL-скрипт, що коректує помилки моделювання. Для створення текстового файла скрипта необхідно перейти в меню File/SQL Generation/New File.

Нижче наведено фрагмент коригувального коду, згенерованого для Oracle.

---iCorrective Oracle SQL Scripts for model: DEMO

---iIncluding implied foreign keys.

---iWed Jun 06 15:25:13 2007

---i Transform implied relationship between CUSTOMER and ORDER_TBL to physical foreign key

ALTER TABLE ORDER_TBL ADD (CONSTRAINT FK63ORDER_TBL_CUSTOMER FOREIGN KEY

(CUSTOMER_NBR)

REFERENCES CUSTOMER

(CUSTOMER_NBR));

---i Transform implied relationship between EMPLOYEE and SALESMAN to physical foreign key

ALTER TABLE SALESMAN ADD (CONSTRAINT FK79SALESMAN_EMPLOYEE FOREIGN KEY

(EMPLOYEE_ID)

REFERENCES EMPLOYEE

(EMPLOYEE_ID));

---i Transform implied relationship between ITEM and ITEM_HISTORY to physical foreign key

ALTER TABLE ITEM_HISTORY ADD (CONSTRAINT. FK40ITEM_HISTORY_ITEM FOREIGN KEY

(ITEM_ID)

REFERENCES ITEM

(ITEM_ID));

6.4. Навчальна опція з моделювання даних в AllFusion DMV

Ключовою можливістю AllFusion DMV є навчання моделюванню даних. При виклику опису помилки (кнопка – ліворуч від імені помилки у вкладці Diagnostics) з'являється діалог з описом помилки, що містить кнопку Teach Me. Клацання по цій кнопці викликає довідку з цієї проблеми, включаючи приклади й опис шляхів її вирішення.

Отже, користувачі ***AllFusion DMV*** навчаються, у першу чергу, за тими темами, які вони погано знають.

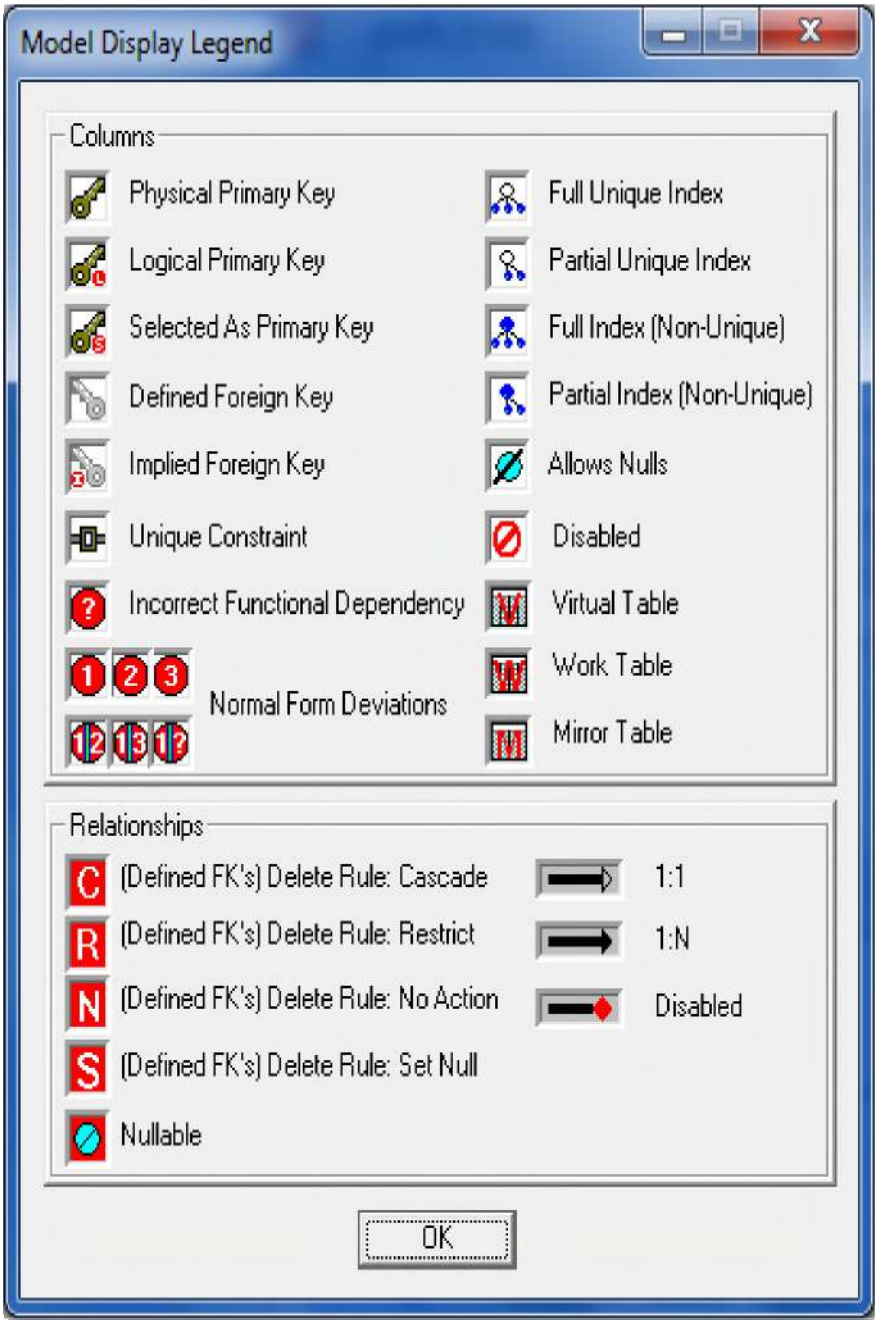


Рис. 6.23. Довідка за системою позначень AllFusion DMV

6.5. Порівняння моделі даних і об'єднання DDL-скриптів

Крім виявлення помилок, AllFusion DMV дозволяє також порівнювати моделі даних і поєднувати DDL-скрипти. Для роботи з великими моделями передбачена зручна навігація за моделлю і робота з підмоделями, причому діагностика може бути проведена в рамках окремої підмоделі.

Для порівняння моделей варто перейти в меню File/Compare. З'являється діалог Select Models To Compare (рис. 6.24).

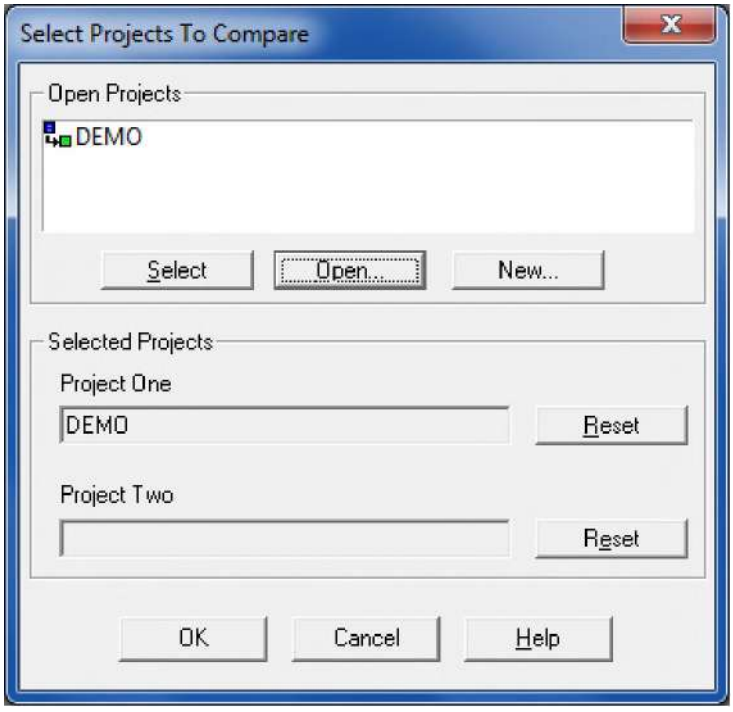


Рис. 6.24. Діалог Select Models To Compare

Кнопка Open дозволяє обрати модель для включення у список Open Models у верхній частині діалогу. Кнопка Select призначена для вибору пари моделей для порівняння. Клацання по кнопці ОК викликає діалог Model Comparison.

Діалог Model Comparison містить два вікна, кожне з яких відображає ієрархічний список об'єктів порівнюваних моделей (*таблиці і колонки відображаються в різних вкладках*).

Включення опції Show Differences Only дозволяє відобразити тільки розбіжності між моделями.

6.6. Лабораторна робота 14

Тема. Аналіз і вдосконалення моделі даних «Наскрізний індивідуальний проект» в CASE-засобі Data Modeler Valadator (DMV) пакета AllFussion Modeling Suite

Мета роботи:

✓ вивчення основних принципів та технології роботи в середовищі AllFusion DMV;

✓ вивчення основних помилок і недоліків при моделюванні колонок, ключів, індексів, обмежень і зв'язків моделі даних;

✓ засвоєння технології з пошуку і виправлення помилок за допомогою AllFusion DMV;

✓ ознайомлення з іншими можливостями засобу AllFusion DMV (*навчальний та оцінюючий сегменти*);

✓ удосконалення моделі даних «Наскрізного індивідуального проекту» з використанням засобу AllFusion DMV.

Завдання

1. Засвоїти технологію з пошуку і виправлення помилок за допомогою AllFusion DMV.

2. Удосконалити власний проект з використанням засобу AllFusion DMV шляхом проведення аналізу моделі власного проекту бази даних у процесі виявлення помилок за категоріями:

✓*категорія* Columns – помилки проектування колонок;

✓*категорія* Indexes and Constraints – помилки проектування індексів і обмежень;

✓*категорія* Normalization – помилки нормалізації. Виявляються некоректно визначенні функціональні залежності і помилки першої, другої та третьої нормальних форм *(слід врахувати ту обставину, що при перевірці структури сховищ даних перевірку нормалізації можна відключити)*;

✓*категорія* – помилки зв'язків (Relationships).

Хід виконання роботи

1. Вивчити основні принципи роботи засобу AllFusion DMV.

2. Вивчити основні помилки і недоліки при моделюванні колонок, ключів, індексів, обмежень і зв'язків моделі даних.

3. Засвоїти технологію з пошуку і виправлення помилок за допомогою AllFusion DMV.

4. Ознайомитися з іншими можливостями засобу AllFusion DMV (*навчальний та оцінюючий сегменти*).

5. Удосконалити «Наскрізний індивідуальний проект» з використанням засобу AllFusion DMV шляхом проведення аналізу колонок, ключів, індексів, обмежень і зв'язків моделі даних.

Оформлення звіту

Звіт про виконану роботу оформляється згідно із «Загальними рекомендаціями до виконання лабораторних робіт та оформлення звіту» (додаток А: А.1, А.2).

Контрольні запитання

1. Суперечливе визначення типу, які наслідки можуть бути і як можна виправити цю помилку?

2. Суперечливе визначення груп колонок, які наслідки можуть бути і як можна виправити цю помилку?

3. Таблиці з неунікальними іменами, які наслідки можуть бути і як можна виправити цю помилку?

4. Таблиця не має колонок, які наслідки можуть бути і як можна виправити цю помилку?

5. Суперечливе значення за замовчуванням, які наслідки можуть бути і як можна виправити цю помилку?

6. Імена колонок збігаються з резервованими словами ANSI SQL, які наслідки можуть бути і як можна виправити цю помилку?

7. Таблиця не має зв'язків, які наслідки можуть бути і як можна виправити цю помилку?

8. Колонки потенційних ключів допускають значення NULL, які наслідки можуть бути і як можна виправити цю помилку?

9. Довжина первинного ключа досить велика, які наслідки можуть бути і як можна виправити цю помилку?

10. Первинні ключі із досить великою кількістю колонок, які наслідки можуть бути і як можна виправити цю помилку?

11. Помилка зовнішнього ключа, які наслідки можуть бути, як можна виправити цю помилку?

12. Невизначені зв'язки, які наслідки можуть бути і як можна виправити цю помилку?

13. Нескінченні цикли, які наслідки можуть бути і як можна виправити цю помилку?

6.7. Інтегративні можливості CASE-пакета структурного аналізу і проектування AllFussion Modeling Suite CA

Модель даних і її відповідність моделі процесів. Функціональна модель AllFusion PM є основою для побудови моделі даних. Дійсно, не маючи інформації про те, як працює підприємство, недоцільно будувати модель даних. Для побудови моделі даних зручно використовувати спеціалізований засіб фірми Computer Associates – AllFusion ERwin Data Modeler (DM).

Необхідно зазначити, що процес перетворення моделі AllFusion PM у модель даних погано формалізується і тому неавтоматизований.

Модель даних, як правило, створюється вручну в середовищі AllFusion ERwin DM, при цьому функціональна модель використовується як проектна документація.

Після розробки моделі даних її варто пов'язати з моделлю процесів. Такий зв'язок гарантує завершеність аналізу та наявність джерела даних як сутність для всіх потреб даних, а саме робіт. Зв'язки об'єктів сприяють погодженості, коректності та завершеності аналізу.

Стрілки в моделі процесів позначають деяку інформацію, що використовується в модельованій системі. У методології IDEF1X AllFusion ERwin DM на логічному рівні моделі даних інформація відображається у вигляді сутностей (*відповідають таблицям на фізичному рівні*), що складаються з атрибутів сутностей (*відповідають колонкам таблиці*).

На діаграмах AllFusion ERwin DM сутності подаються у вигляді прямокутників. Є кілька рівнів подання моделі. На рівні сутностей ім'я сутності вказується усередині прямокутника (рис. 6.25).

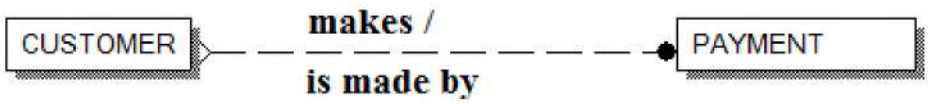


Рис. 6.25. Фрагмент моделі даних в IDEF1X (рівень сутностей)

На рівні атрибутів ім'я сутності вказується над прямокутником, атрибути сутності вказуються у вигляді списку усередині прямокутника (рис. 6.26).

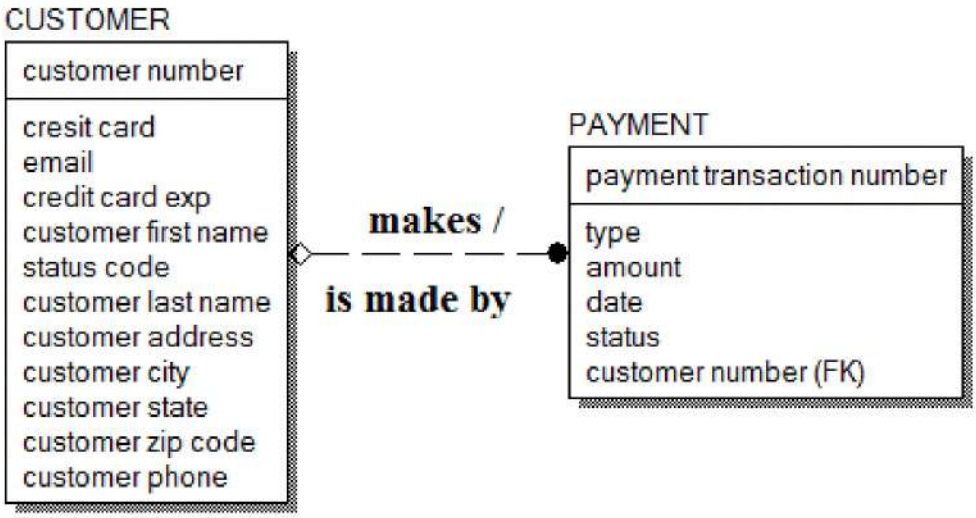


Рис. 6.26. Фрагмент моделі даних в IDEF1X (рівень атрибутів)

Сутності складаються із сукупності окремих записів – екземплярів сутностей (*відповідають записам у таблиці*). До моделі даних пред'являються певні вимоги (*нормалізація даних*), які покликані забезпечити компактність і несуперечність зберігання даних.

Основна ідея нормалізації даних – кожний факт повинен зберігатися в одному місці. Це приводить до того, що змодельована інформація у вигляді однієї стрілки в моделі процесів може зберігатися в декількох сутностях і атрибутах у моделі даних. Отже, одній і тій же стрілці в моделі процесів можуть відповідати кілька сутностей у моделі даних і, навпаки, одній сутності може відповідати кілька стрілок.

Стрілці в моделі процесів (діаграми «Виготовлення запчастин» та «Збір запчастин») може відповідати окрема сутність у

моделі даних (сутності «Запчастини», «Продукт» та «Працівник» (рис. 6.27). Так, стрілці «*Запчастини*» відповідає сутність «*Запчастини*», стрілці «*Кінцеві продукти*» – сутність «*Продукт*». Інформація про стрілку може міститися тільки в декількох атрибутах сутності. Різним атрибутам однієї і тієї ж сутності можуть відповідати різні стрілки.

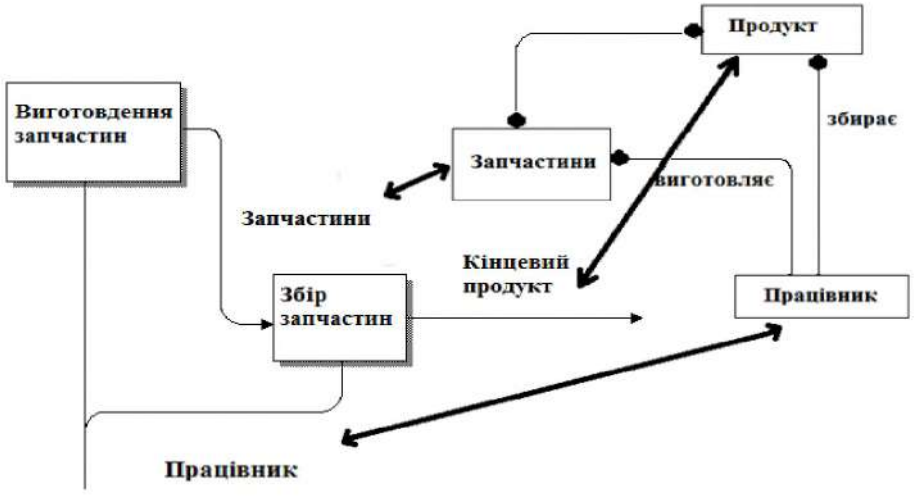


Рис. 6.27. Перетворення стрілки в сутність

На рис. 6.28 стрілка «*Нова запчастина*» відповідає атрибутам «*Номер запчастини*» і «*Назва запчастини*», стрілка «*Наявність кількості*» – атрибуту «*Кількість*».

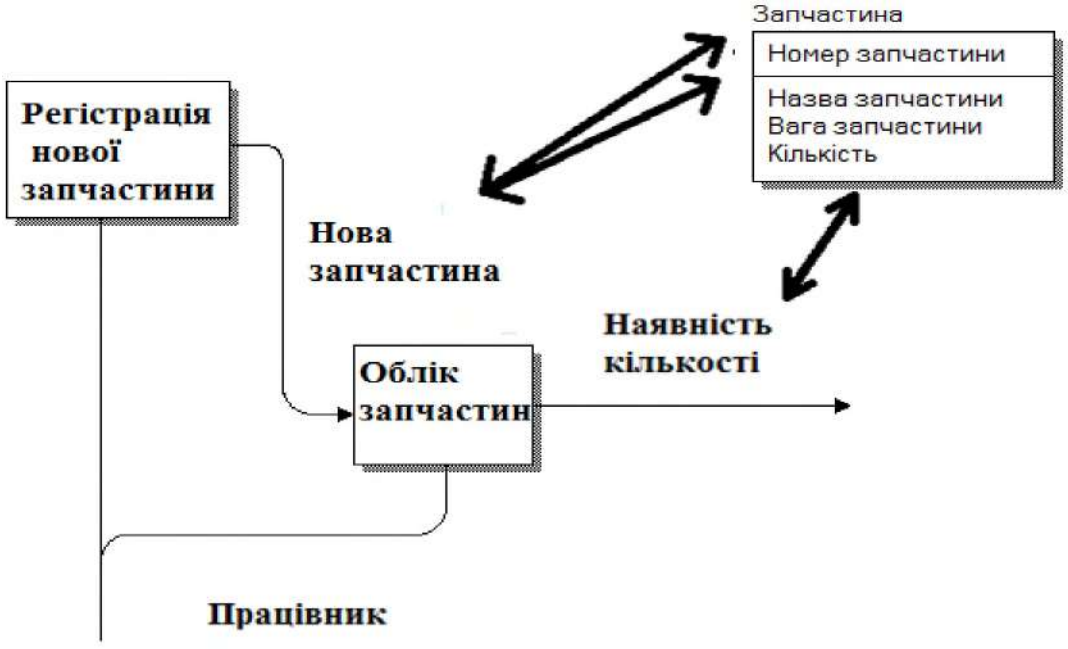


Рис. 6.28. Перетворення стрілки в атрибут

Роботи в моделі процесів можуть створювати або змінювати дані, які відповідають вхідним або вихідним стрілкам. Вони можуть впливати повністю на сутності, створюючи або модифікуючи екземпляри сутності (рис. 6.29), або на окремі атрибути сутності (рис. 6.30).

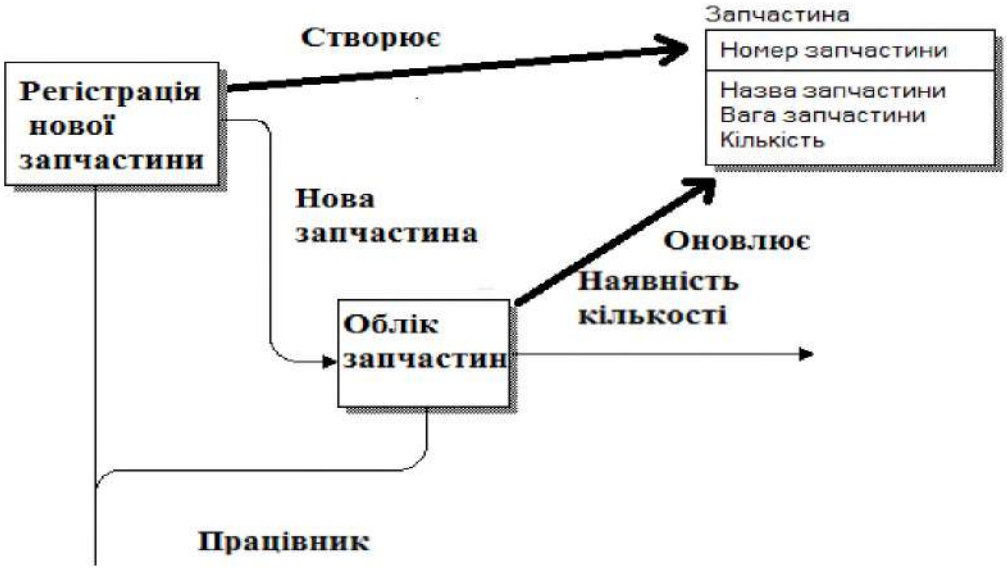


Рис. 6.29. Вплив роботи функціональної моделі на сутність моделі даних

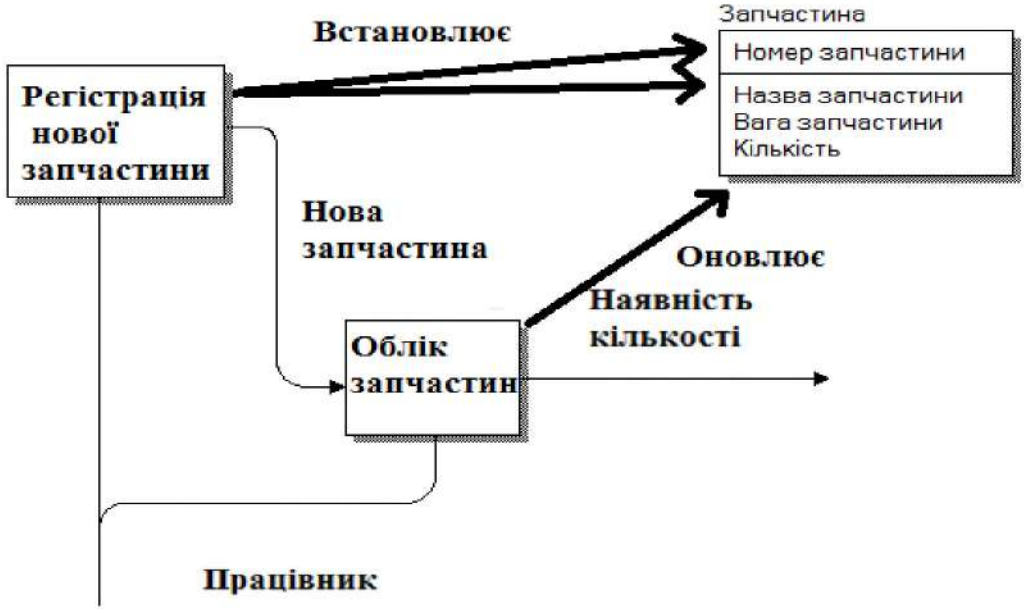


Рис. 6.30. Вплив роботи функціональної моделі на атрибут моделі даних

6.8. Експорт моделі даних з AllFusion ERwin DM в AllFusion PM

AllFusion PM дозволяє пов'язувати елементи моделі даних, створеної за допомогою AllFusion ERwin DM, документувати вплив робіт на дані і тим самим дозволяє створити специфікації на права доступу до даних для кожного процесу. Існує два способи пов'язування об'єктів моделі даних і моделі процесів:

1. Експорт і імпорт через файли формату ***.eax – .bpx***.
2. Синхронізація моделей, що зберігаються в AllFusion Model Manager.

Першим кроком пов'язування моделі даних і моделі процесів є експорт даних з AllFusion ERwin DM в AllFusion PM. Для успішного пов'язування моделей необхідно, щоб версії засобів AllFusion ERwin DM і AllFusion PM відповідали один одному.

Експорт і імпорт через файли формату .eax – .bpx. Для експорту моделі даних з AllFusion ERwin DM в AllFusion PM необхідно в AllFusion ERwin DM відкрити модель (рис. 6.31) і обрати пункт меню File/Export/ AllFusion PM. У діалозі, що з'явився, необхідно обрати ім'я файла *.eax і натиснути ОК. Потім в AllFusion PM потрібно відкрити модель процесу, обрати в меню пункт File/Import/ERwin (eax), ім'я файла і натиснути ОК.

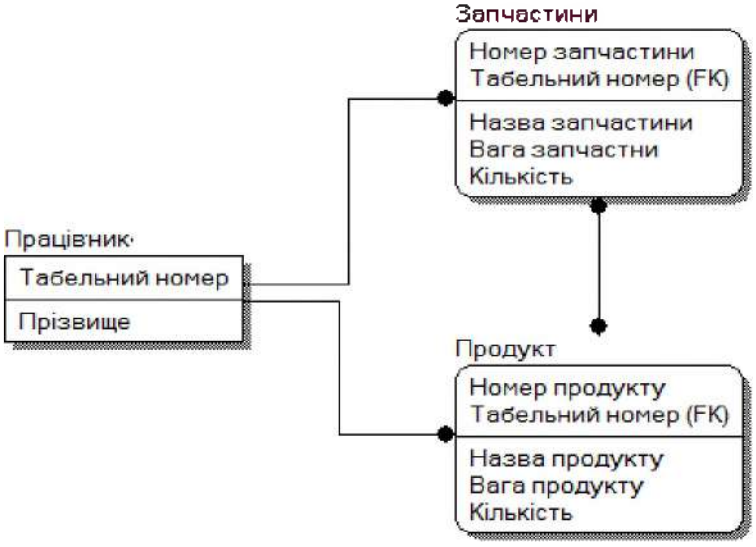


Рис. 6.31. Модель даних, відкрита в AllFusion ERwin DM

З'явиться діалог Import Differences Preview, у якому показується протокол імпорту (рис. 6.32).

Для внесення даних у модель процесу потрібно клацнути по кнопці Accept. Кнопка Cancel скасовує імпорт.

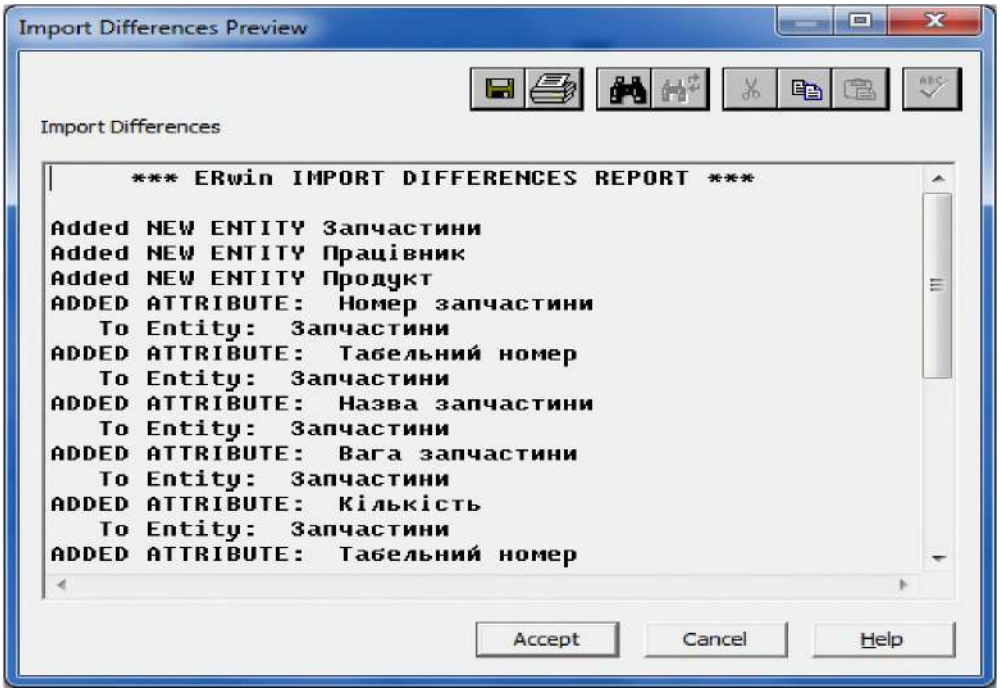


Рис. 6.32. Діалог Import Differences Preview

Після внесення даних у модель процесів можна пов'язати сутності й атрибути зі стрілками. Правою кнопкою миші потрібно клацнути по стрілці й обрати в контекстному меню Arrow Data. З'являється вкладка Arrow Data діалогу Arrow Properties (рис. 6.33).

Для пов'язування атрибута зі стрілкою достатньо клацнути по іконці вибору атрибута в їхньому ієрархічному списку. При цьому сутність автоматично пов'язується зі стрілкою. Кожна стрілка в моделі процесів може бути пов'язана з декількома атрибутами різних сутностей.

Кнопка Copy In дозволяє копіювати пов'язані дані з іншої стрілки.

Кнопка Clear дозволяє копіювати всі зв'язки стрілки з даними.

Кнопка Migrate викликає діалог Changes to Arrow Data Associations, у якому відображаються дані, що мігрують від дочірніх до батьківських стрілок (*для стрілок, що розгалужуються і зливаються*). При міграції можливі зміни пов'язування даних:

1. *Deletions* – якщо дані пов'язані з батьківською стрілкою, але не пов'язані з дочірньою, зв'язки з батьківською стрілкою видаляються.

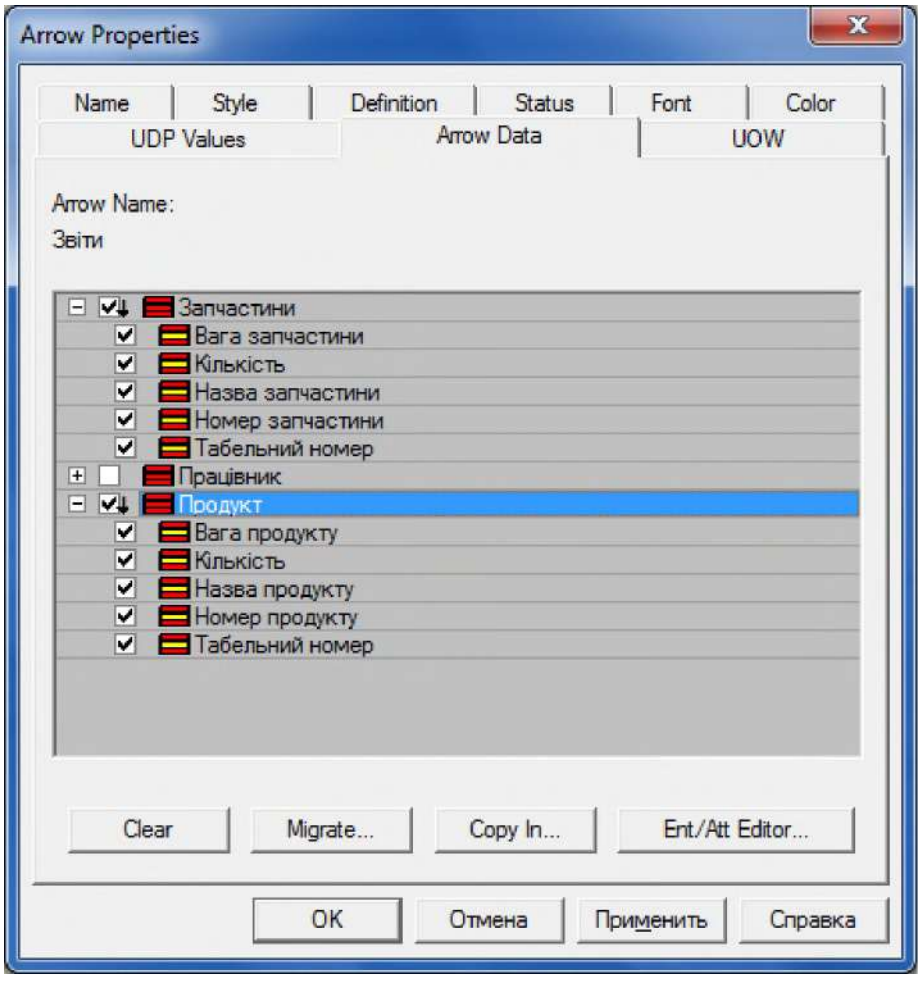


Рис. 6.33. Вкладка Arrow Data діалогу Arrow Property

2. *Additions* – якщо дані пов'язані з дочірньою стрілкою і не пов'язані з батьківською, додається зв'язок з батьківською стрілкою.

Для підтвердження змін у діалозі Changes to Arrow Data Associations потрібно клацнути по кнопці OK. Міграція можлива тільки в моделях IDEF0 і DFD.

Документування впливу «роботи» на дані. Як було зазначено вище, роботи можуть впливати на дані. Для документування такого впливу необхідно клацнути правою кнопкою миші по роботі й обрати пункт меню Data Usage Editor (рис. 6.34).

У діалозі, що з'явився, Data Usage Editor у вигляді ієрархічного списку показуються всі роботи моделі, стрілки, які стосуються робіт, сутності й атрибути, які були пов'язані зі стрілками.

У верхньому списку потрібно клацнути по імені стрілки, з якою були пов'язані сутності й атрибути. Для задання асоціації достатньо клацнути по вікну атрибута в їхньому ієрархічному списку.

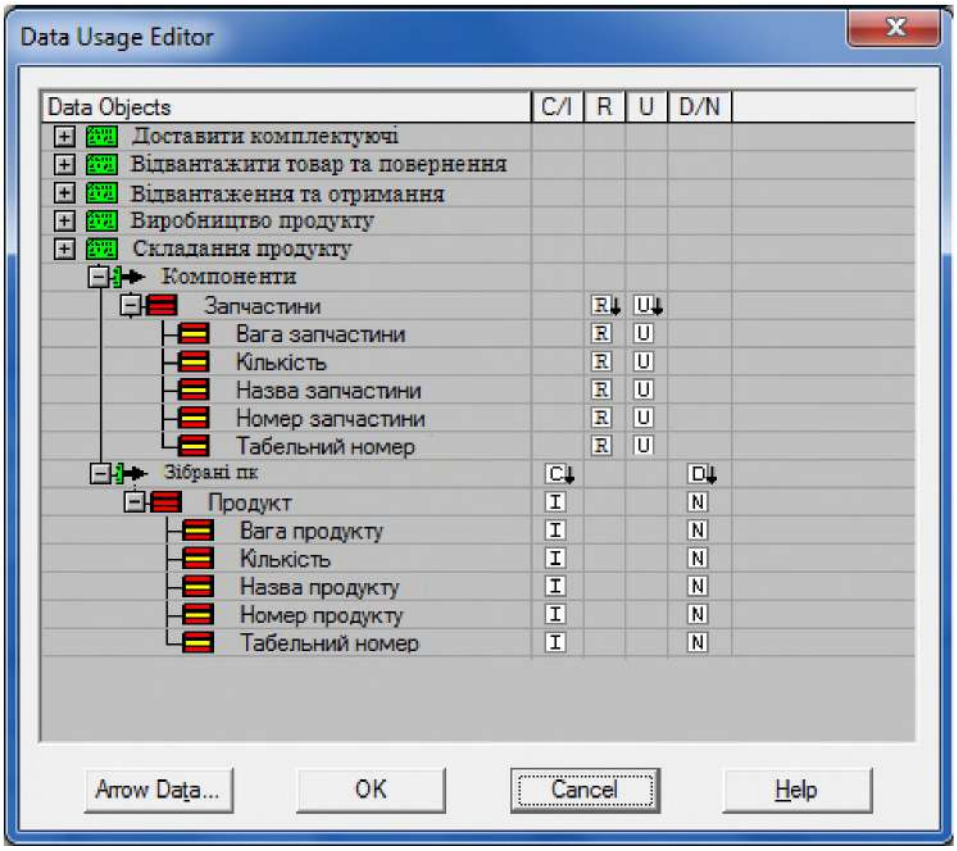


Рис. 6.34. Діалог AllFusion PM Data Usage Editor

Для сутностей задається асоціація CRUD (*Create, Read, Update, Delete*), для атрибутів – IRUN (*Insert, Read, Update, Nullify*).

Асоціації CRUD і IRUN – це правила використання сутностей і атрибутів «роботами», тобто те, що можуть здійснювати *«роботи»* із вхідними або вихідними даними. Дані не можуть використовуватися *«роботами»* довільно.

Стрілки входу представляють дані, які *«робота»* перетворить у вихід або використовує. Такі дані можуть бути оновлені (*Update*) або прочитані (*Read*), але не можуть бути створені (*Create, Insert*) або вилучені (*Delete, Nullify*).

Дані, що пов'язані зі стрілками управління, можуть бути тільки прочитані (*Read*), але не можуть бути змінені – процедури і стратегії не можуть змінюватися в роботі.

Дані, що пов'язані зі стрілками виходу, можуть бути оновлені (*якщо їм відповідають дані стрілок входу*), вилучені (*Delete, Nullify*) або створені (*Create, Insert*).

Для стрілок механізму асоціації не встановлюються. Результат пов'язування об'єктів моделі процесів можна відобразити у звіті Data Usage Report (*меню Report/Data Usage Report*).

Нижче наведений приклад такого звіту.

Arrow Name	*Entity Name*	*C R U D*	*Attribute Name I R U N*
Деталі	*Запчастина*	*U D*	*Вага Запчастини U N*
		U D	*Кількість U N*
		U D	*Назва Запчастини U*
		U D	*Номер Запчастини U*

6.9. Створення сутностей і атрибутів в AllFusion PM та їх експорт в AllFusion ERwin DM

Створення сутностей і атрибутів AllFusion PM. Якщо в процесі пов'язування стрілок з об'єктами моделі даних виявиться, що будь-яких сутностей або атрибутів не вистачає, їх можна додати прямо в AllFusion PM, а потім експортувати в AllFusion ERwin DM.

Для редагування сутностей потрібно обрати пункт меню Dictionary/Entity. З'являється діалог Entity Dictionary (рис. 6.35) – словник сутностей. Інтерфейс словника сутностей повністю аналогічний інтерфейсу словника стрілок. Для експорту в AllFusion ERwin DM у словнику Entity Dictionary потрібно створити нову сутність (наприклад, фірма), яка може бути використана для асоціації зі стрілками одразу ж після створення (до експорту в AllFusion ERwin DM).

Для редагування атрибутів попередньо створених сутностей слугує словник атрибутів (пункт меню Dictionary/Entity/ Attribute).

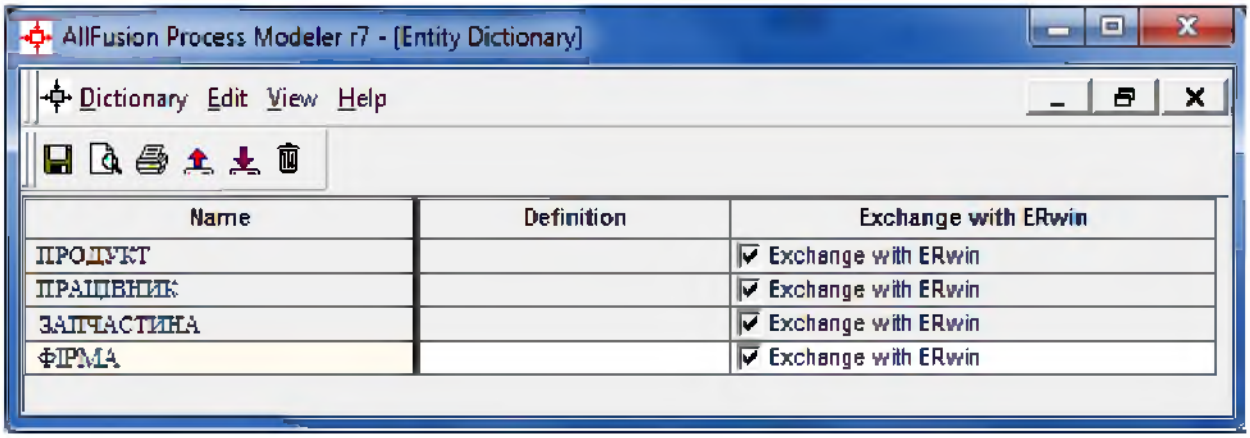


Рис. 6.35. Діалог Entity Dictionary

Колонка Entity діалогу Attribute Dictionary слугує для пов'язування створеного атрибуту із сутністю (рис. 6.36).

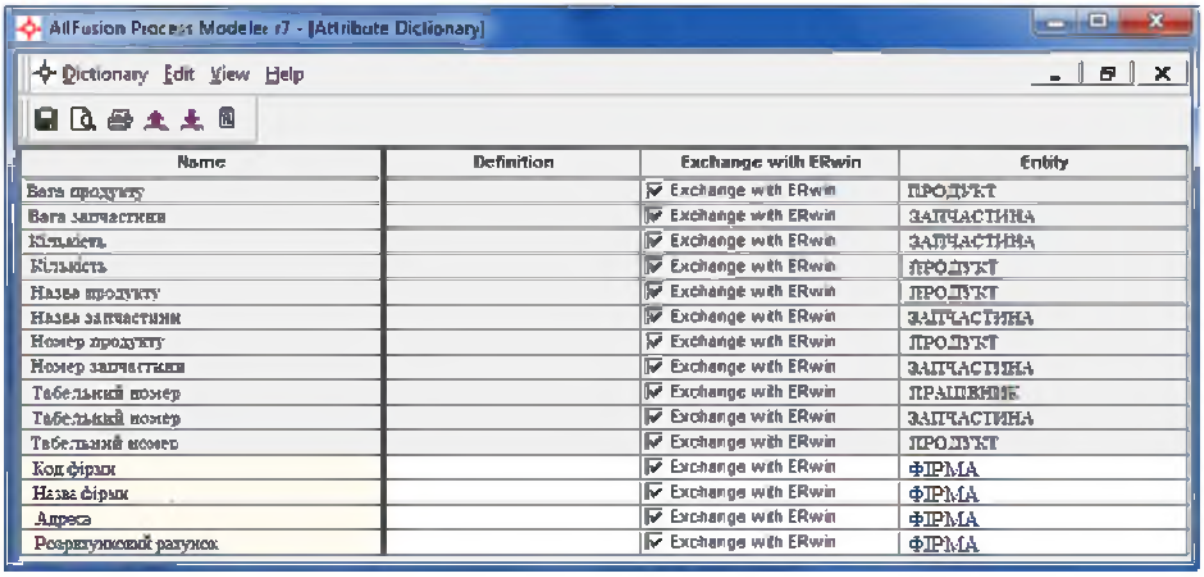


Рис. 6.36. Діалог Attribute Dictionary

У розкриваючому списку, що з'являється, коли фокус установлений на поле Entity таблиці, показуються тільки ті сутності, які створені в діалозі Entity Dictionary або імпортовані з AllFusion ERwin DM.

Після опису сутностей або атрибутів потрібно зберегти дані і вийти зі словника.

Експорт в AllFusion ERwin DM сутностей і атрибутів, які створені в AllFusion PM

Для експорту даних в AllFusion PM потрібно обрати меню File/Export/ ERwin (BPX) і вказати файл, у який буде вивантажена інформація про модель. У AllFusion ERwin DM потрібно обрати меню File/Import/AllFusion PM і вказати файл BPX, у який була вивантажена інформація про модель.

Виникає діалог ERwin/AllFusion PM Import (рис. 6.37), у якому відображаються:

✓ сутності й атрибути, наявні у BPX-файлі, але відсутні в моделі AllFusion ERwin DM (*верхнє вікно – Entities/Attributes available to be imperted)*;

✓ імена робіт, асоційованих із сутностями й атрибутами, на основі яких будуть створені предметні області (Subject Area) моделі даних.

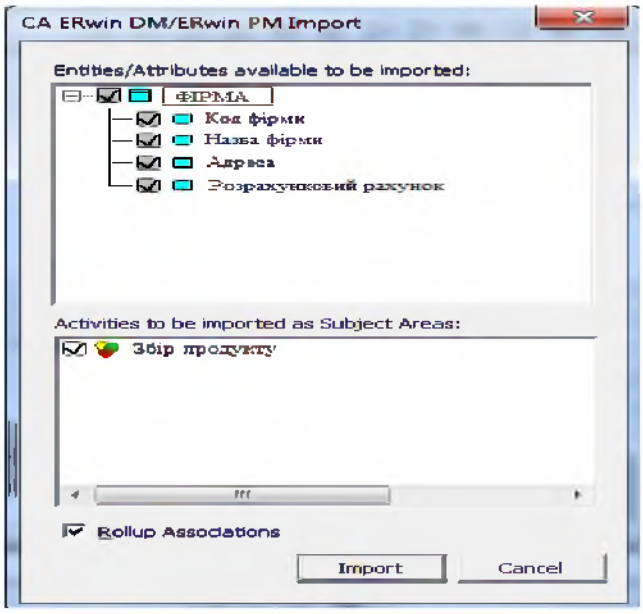


Рис. 6.37. Імпорт сутності «Фірма» із BPX-файла

У прикладі на рис. 6.37 сутність «Фірма» з атрибутами («Прізвище», «Ім'я», «Адреса») буде імпортована із BPX-файла в модель AllFusion ERwin DM.

Після клацання по кнопці Import запускається процес імпорту BPX-файла. Імпортована сутність «Фірма» розміщується в лівому верхньому куті діаграми AllFusion ERwin DM (рис. 6.38).

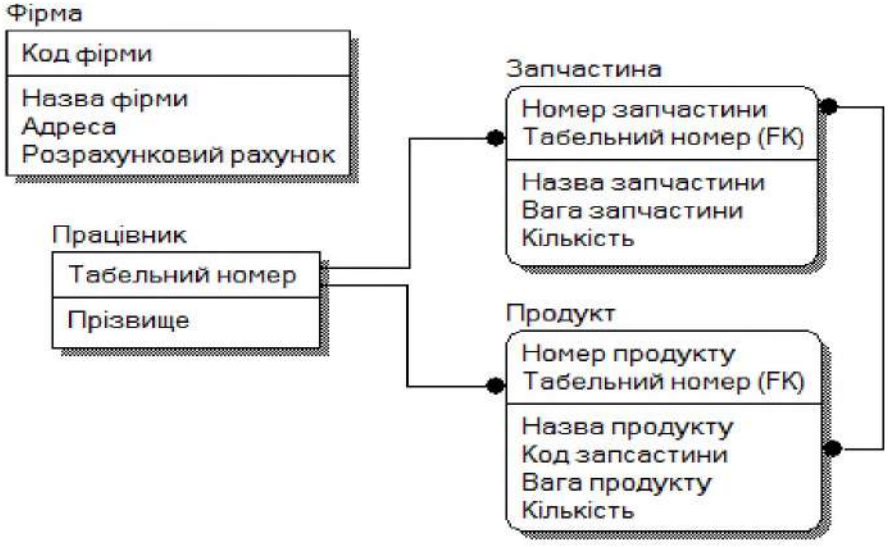


Рис. 6.38. Модель даних після імпорту сутності «Фірма»

Завершення моделювання даних в AllFusion ERwin DM. Імпортована сутність у моделі даних не має первинного ключа і

не пов'язана з іншими сутностями. Назначення атрибута «Код фірми» первинним ключем і пов'язування сутностей можна провести тільки засобами AllFusion ERwin DM (рис. 6.39).



Рис. 6.39. Модель даних після назначення атрибутів первинним ключем і пов'язування сутності «Фірма»

Таким чином, можна зробити висновок: сутності й атрибути, що створені в AllFusion PM і потім імпортовані в AllFusion ERwin DM, можна розглядати як заготовки для створення повноцінної моделі даних, а не як готову модель.

Контрольні запитання

1. Як здійснюється експорт та імпорт моделей процесів і даних?
2. Чого досягається в процесі інтеграції моделей, що зберігаються в AllFusion Model Manager?
3. Як можуть використовуватися сутності й атрибути, що створені в AllFusion PM і потім імпортовані в AllFusion ERwin DM?
4. Чи можуть використовуватися сутності й атрибути, що створені в AllFusion PM і потім імпортовані в AllFusion ERwin DM як готову модель?
5. Чи може функціональна модель AllFusion PM бути основою для побудови моделі даних?

6. Чи добре формалізується процес перетворення моделі AllFusion PM у модель даних?

7. Чому процес перетворення моделі AllFusion PM у модель даних неавтоматизований?

8. Наскільки автоматизовано створюється модель даних у середовищі AllFusion ERwin DM?

9. Яке місце функціональної моделі у процесі створення моделі даних у середовищі AllFusion ERwin DM?

10. Для чого модель даних потрібно пов'язати з моделлю процесів?

11. Яка цінність стрілок у моделі процесів для модельованої системи?

6.10. Лабораторна робота 15

Тема. Інтеграція об'єктів моделі процесів і моделі даних реалізацією механізму двонаправленого зв'язку AllFusion PM – AllFusion ERwin DM

Мета роботи:

✓ вивчення основних принципів інтеграції CASE-пакета AllFussion Modeling Suite для вдосконалення «Наскрізного індивідуального проекту»;

✓ вивчення основних принципів відповідності моделі даних її моделі процесів, що створені за допомогою CASE-пакета AllFussion Modeling Suite;

✓ засвоєння технології пов'язування об'єктів моделі даних зі стрілками і роботами шляхом експорту даних з AllFusion ERwin DM в AllFusion PM;

✓ засвоєння технології створення сутностей і атрибутів AllFusion PM та їхній експорт і актуалізація в AllFusion ERwin DM.

Завдання

1. На модельному прикладі бізнес-завдання «Виготовлення запасних частин та кінцевого продукту» засвоїти основні принципи інтеграції CASE-пакета AllFussion Modeling Suite пов'язуванням моделі процесів і моделі даних.

2. Удосконалити «Наскрізний індивідуальний проект» з використанням технології пов'язування об'єктів моделі процесів і

моделі даних у середовищі CASE-пакета AllFussion Modeling Suite для власного індивідуального об'єкта дослідження:

✓ виконати експорт моделі даних з AllFusion ERwin DM в AllFusion PM CASE-пакета AllFussion Modeling Suite;

✓ пов'язати сутності й атрибути зі стрілками після внесення даних у модель процесів;

✓ здійснити пов'язування атрибута зі стрілкою;

✓ виконати документування впливу «роботи» на дані;

✓ створити необхідну сутність і відповідні атрибути в AllFusion PM;

✓ здійснити експорт в AllFusion ERwin DM сутностей і атрибутів, які створені в AllFusion PM;

✓ завершити моделювання даних в AllFusion ERwin DM.

Хід виконання

1. Поновлення «Наскрізного індивідуального проекту» у вигляді моделі даних і моделі процесів, створених у середовищі AllFussion Erwin DM і AllFussion PM CASE-пакета AllFussion Modeling Suite для власного індивідуального об'єкта дослідження.

2. Вивчення основних принципів інтеграції CASE-пакета AllFussion Modeling Suite.

3. Вивчення основних принципів відповідності моделі даних її моделі процесів, що створені за допомогою CASE-пакета AllFussion Modeling Suite.

4. Засвоєння технології пов'язування об'єктів моделі даних зі стрілками і роботами шляхом експорту даних з AllFusion ERwin DM в AllFusion PM на прикладі «Наскрізного індивідуального проекту».

5. Засвоєння технології створення сутностей і атрибутів AllFusion PM, а також їхній експорт і актуалізація в AllFusion ERwin DM на прикладі «Наскрізного індивідуального проекту».

6. Виконати всі пункти, які передбачені у Завданні для «Наскрізного індивідуального проекту».

7. Враховуючи індивідуальні особливості об'єкта дослідження, допускається розширення і поглиблення робіт «Індивідуального проекту» (може ускладнитися процес отримання зворотної інформації про виконання індивідуальних завдань та їхнього корегування).

Оформлення звіту

Звіт про виконану роботу оформляється згідно із «Загальними рекомендаціями до виконання лабораторних робіт та оформлення звіту» (додаток А: А.1, А.2).

Контрольні запитання

1. Як виконати експорт моделі даних з AllFusion ERwin DM в AllFusion PM CASE-пакета AllFussion Modeling Suite?
2. Як пов'язати сутності й атрибути зі стрілками після внесення даних у модель процесів?
3. Як здійснити пов'язування атрибута зі стрілкою?
4. Як виконати документування впливу «роботи» на дані?
5. Як створити необхідну сутність і відповідні атрибути в AllFusion PM?
6. Як здійснити експорт в AllFusion ERwin DM сутностей і атрибутів, які створені в AllFusion PM CASE-пакета AllFussion Modeling Suite?
7. Як завершити моделювання даних в AllFusion ERwin DM?
8. Описати алгоритм пов'язування об'єктів моделі даних і моделі процесів власного проекту шляхом експорту/імпорту через файли формату .eax – .bpx.
9. Описати алгоритм пов'язування об'єктів моделі даних і моделі процесів власного проекту шляхом синхронізації моделей, що зберігаються в AllFusion Model Manager.

ТЕСТИ

1. Яке з цих визначень відповідає поняттю «програмне забезпечення»:

а) комплекс технічних засобів, який включає ЕОМ: зовнішні пристрої, термінали, абонентські пункти тощо, які необхідні для функціонування тієї чи іншої системи;

б) загальне поняття, що вказує на набір кодованих інструкцій (програм) для керування процесором комп'ютера;

в) зовнішні пристрої, термінали, абонентські пункти, які необхідні для функціонування тієї чи іншої системи;

г) набір кодованих інструкцій (програм) для керування процесором комп'ютера та комплекс технічних засобів.

2. Як називається життєвий цикл розробки програмного продукту (ПП):

а) послідовність фаз або стадії від задуму до його втілення в ПП, експлуатацію, вилучення/перепроектування;

б) структура, що складається із процесів, робіт та задач, які включають у себе розробку, експлуатацію і супровід ПП;

в) удосконалення застарілого ПП шляхом його реорганізації або реструктуризації, а також перепрограмування окремих елементів або настроювання параметрів на іншу платформу, середовище виконання зі збереженням зручності його супроводу;

г) сукупність властивостей, які повинен мати ПП.

3. Який правильний порядок етапів життєвого циклу програмних продуктів (ПП):

а) проектування→розробка ТЗ→вимоги→реалізація→впровадження→супровід→зняття з експлуатації/перепроектування;

б) розробка ТЗ→вимоги→проектування→реалізація→впровадження→супровід→зняття з експлуатації/перепроектування;

в) вимоги→розробка ТЗ→проектування→реалізація→-впровадження→супровід → зняття з експлуатації/перепроектування;

г) проектування→вимоги→реалізація→розробка ТЗ→впровадження→супровід→зняття з експлуатації/перепроектування.

4. Яка сутність специфікації вимог до програмного продукту (ПП):

а) спостереження за бізнес-процесом, що підлягають автоматизації;

б) високорівневий опис рішення відповідно до загальновизнаних стандартів розробки бізнес-аналітичних документів;

в) можливість планування робіт і виконання подальших дій з розробки рішення;

г) повний опис поведінки системи, що розробляється.

5. Що таке специфікація вимог до програмного продукту (ПП) у відповідності до системи знань SWEBOK:

а) спостереження за бізнес-процесом, що підлягають автоматизації;

б) опис рішення відповідно до загальновизнаних стандартів розробки бізнес-аналітичних документів;

в) можливість планування робіт і виконання подальших дій з розробки рішення;

г) повний опис поведінки ПП, що розробляється.

6. Що таке стандарт:

а) легкість зміни системи з метою використання у середовищах, на які вона не була орієнтована безпосередньо;

б) можливий ступінь виконання блокового і системного тестування програми та перевірки її відповідності вимогам;

в) здатність ПЗ бути перенесеним з одного середовища (оточення) в інше, зокрема з одного комп'ютера на інший;

г) нормативно-технічний документ, який встановлює комплекс норм, правил, вимог об'єкта стандартизації та затверджений компетентним органом.

7. Що таке ANSI:

а) міжнародна організація зі стандартизації;

б) американський національний інститут стандартів;
в) інститут інженерів з електроніки та електротехніки;
г) міжнародна комісія з електротехніки.

8. Що таке ISO:
а) міжнародна організація зі стандартизації;
б) інститут інженерів з електроніки та електротехніки;
в) міжнародна комісія з електротехніки;
г) американський національний інститут стандартів.

9. Які існують всесвітні стандарти якості програмних продуктів:
а) Programs World Standard (PWS); ISO 9000;
б) ISO 1000; Programs World Standard (PWS);
в) ISO 9000; Malcolm Baldrige National Quality Award (NQA);
г) System Award (SA); ISO 9000.

10. Які спеціальні процеси входять до стандарту ISO/IEC 12207 щодо забезпечення якості ПЗ:
а) валідація; акредитація;
б) планування; верифікація;
в) планування; акредитація;
г) валідація; верифікація.

11. Які практики застосовуються для досягнення того, щоб стандарти відповідали за створення правильного програмного продукту:
а) прототипування; затвердження моделі; приймальні тести;
б) первинний аналіз вимог; документування; приймальні тести;
в) первинний аналіз вимог; затвердження моделі; приймальні тести;
г) документування; затвердження моделі; приймальні тести.

12. Які розділи входять у склад області знань «Вимоги до ПЗ» :

а) інженерія вимог; аналіз вимог; валідація вимог;

б) архітектура вимог; інженерія вимог; проектування вимог;

в) архітектура вимог; аналіз вимог; проектування вимог;

г) архітектура вимог; проектування вимог; валідація вимог.

13. Яку сукупність властивостей повинно мати програмне забезпечення:

а) характеристики;

б) вимоги;

в) функції;

г) параметри.

14. Яка техніка виявлення та подання вимог:

а) планування; спостереження; моделювання;

б) опитування; спостереження; моделювання;

в) консультація зі спеціалістом; планування; моделювання;

г) консультація зі спеціалістом; спостереження; моделювання.

15. Які стадії етапу накопичення вимог:

а) збір первинної інформації і вимог; документування; первинний аналіз вимог; маркетинг дослідження;

б) збір первинної інформації і вимог; документування; первинний аналіз вимог; перевірка;

в) маркетинг дослідження; документування; первинний аналіз вимог; перевірка;

г) вивчення первинної інформації; документування; первинний аналіз вимог; перевірка.

16. Що таке якість ПЗ:

а) нормативно-технічний документ, який встановлює комплекс норм, правил, вимог об'єкта стандартизації та затверджений компетентним органом;

б) це міра, що дозволяє отримати числове значення деяких властивостей програмного забезпечення та його специфікацій;

в) сукупність його властивостей і характеристик, що впливають на його здатність задовольняти задані потреби користувачів;

г) можливий ступінь виконання блокового і системного тестування програми та перевірки її відповідності вимогам.

17. Що таке функціональність ПЗ:

а) здатність ПЗ виконувати набір функцій, які задовольняють потреби користувачів. Набір зазначених функцій визначається в зовнішньому описі ПЗ;

б) здатність ПЗ бути перенесеним з одного середовища (оточення) в інше, зокрема з одного комп'ютера на інший;

в) можливість використання системи без її зміни в тих галузях або середовищах, на які вона не була орієнтована безпосередньо;

г) здатність ПЗ безвідмовно виконувати визначені функції при заданих умовах упродовж заданого періоду часу з досить великою імовірністю.

18. Що таке практичність ПЗ:

а) здатність ПЗ бути перенесеним з одного середовища (оточення) в інше, зокрема з одного комп'ютера на інший;

б) здатність ПЗ безвідмовно виконувати визначені функції при заданих умовах протягом заданого періоду часу з досить великою імовірністю;

в) можливість використання системи без її зміни в тих галузях або середовищах, на які вона не була орієнтована безпосередньо;

г) легкість вивчення і використання системи.

19. Що таке надійність ПЗ:

а) відношення рівня послуг, які надає ПЗ користувачу при заданих умовах, до обсягу використовуваних ресурсів;

б) здатність ПЗ безвідмовно виконувати визначені функції при заданих умовах упродовж заданого періоду часу з досить великою імовірністю;

в) можливість використання системи без її зміни в тих галузях або середовищах, на які вона не була орієнтована безпосередньо;

г) можливий ступінь виконання блокового і системного тестування програми та перевірки її відповідності вимогам.

20. Що таке супровід ПЗ:

а) здатність ПЗ безвідмовно виконувати визначені функції при заданих умовах упродовж заданого періоду часу з досить великою імовірністю;

б) відсутність/наявність дефектів у специфікації, проекті та реалізації системи;

в) характеристики ПЗ, що дозволяють мінімізувати зусилля щодо внесення змін для усунення в ньому помилок і з його модифікації відповідно до потреб користувачів;

г) сукупність властивостей і характеристик, що впливають на його здатність задовольняти задані потреби користувачів.

21. Що таке мобільність ПЗ:

а) здатність ПЗ бути перенесеним з одного середовища (оточення) в інше, зокрема з одного комп'ютера на інший;

б) сукупність властивостей і характеристик, що впливають на його здатність задовольняти задані потреби користувачів;

в) можливий ступінь виконання блокового і системного тестування програми та перевірки її відповідності вимогам;

г) здатність ПЗ безвідмовно виконувати визначені функції при заданих умовах упродовж заданого періоду часу з досить великою імовірністю.

22. Що таке легкість застосування ПЗ:

а) здатність ПЗ безвідмовно виконувати визначені функції при заданих умовах упродовж заданого періоду часу з досить великою імовірністю;

б) характеристики ПЗ, що дозволяють мінімізувати зусилля користувача з підготовки вхідних даних, застосування ПЗ і оцінки отриманих результатів;

в) здатність ПЗ бути перенесеним з одного середовища (оточення) в інше, зокрема з одного комп'ютера на інший;

г) легкість вивчення і використання системи.

23. Що таке зручність супроводу ПЗ:

а) здатність ПЗ бути перенесеним з одного середовища (оточення) в інше, зокрема з одного комп'ютера на інший;

б) характеристики ПЗ, що дозволяють мінімізувати зусилля користувача з підготовки вхідних даних, застосування ПЗ і оцінки отриманих результатів;

в) легкість зміни програмної системи з метою реалізації додаткових можливостей, підвищення швидкодії, виправлення дефектів тощо;

г) здатність ПЗ безвідмовно виконувати визначені функції при заданих умовах упродовж заданого періоду часу з досить великою імовірністю.

24. Що таке гнучкість ПЗ:

а) можливий масштаб зміни системи з метою використання її в тих галузях або середовищах, на які вона не була безпосередньо орієнтована;

б) здатність ПЗ безвідмовно виконувати визначені функції при заданих умовах упродовж заданого періоду часу з досить великою імовірністю;

в) легкість зміни програмної системи з метою реалізації додаткових можливостей, підвищення швидкодії, виправлення дефектів тощо;

г) здатність ПЗ бути перенесеним з одного середовища (оточення) в інше, зокрема з одного комп'ютера на інший.

25. Що таке портованість ПЗ:

а) здатність ПЗ бути перенесеним з одного середовища (оточення) в інше, зокрема з одного комп'ютера на інший;

б) легкість зміни програмної системи з метою реалізації додаткових можливостей, підвищення швидкодії, виправлення дефектів тощо;

в) здатність ПЗ безвідмовно виконувати визначені функції при заданих умовах упродовж заданого періоду часу з досить великою імовірністю;

г) легкість зміни системи з метою використання у середовищах, на які вона не була орієнтована безпосередньо.

26. Який процес розробки ПЗ надає відповідь на запитання, чи правильно створюється система:

а) верифікація;

б) валідація;

в) інспекція;

г) атестація.

27. Що таке «успішний тест» у процесі технічного дослідження ПЗ:

а) тест, результати якого відповідають потребам замовника;

б) тест, який виявляє раніше невідомі проблеми;

в) тест із успішним результатом;

г) тест, який не має негативних результатів.

28. Що включає в себе тестування ПЗ як одна із технік контролю якості:

а) планування робіт; проектування тестів; виконання тестування; аналіз отриманих результатів;

б) планування робіт; проектування тестів; виконання тестування; виправлення помилок;

в) проектування тестів; виконання тестування; аналіз отриманих результатів; виправлення помилок;

г) планування робіт; проектування тестів; аналіз отриманих результатів; виправлення помилок.

29. Що таке «якість ПЗ» :

а) дії, які мають визначити, якою мірою ПЗ відповідає своєму призначенню;

б) сукупність властивостей, що визначають спроможність задовольнити запити замовника, які він висловив у вигляді вимог до розроблень;

в) коректність, повнота, безпечність;

г) характеристика програмного забезпечення.

30. Які дії не належать до процесу «супровід програмного продукту (ПП)» :

а) сукупність дій із забезпечення роботи ПП;

б) внесення змін при виявленні помилок, адаптація ПП до нового середовища функціонування;

в) підвищення продуктивності або поліпшення деяких характеристик ПП;

г) сукупність дій із розповсюдження ПП.

31. За допомогою яких технологій здійснюється супровід ПП як його еволюційна розробка:

а) реінженерія; реверсна інженерія; рефакторинг;

б) реінженерія; маркетинг; рефакторинг;

в) реінженерія; моніторинг; рефакторинг;

г) маркетинг; моніторинг; реверсна інженерія.

32. З якою метою проводиться супровід програмного продукту (ПП) відповідно до стандартів ISO/IEC 12207 і ISO/IEC 14764:

а) продаж і модифікація ПП за умови збереження його цілісності;

б) розповсюдження ПП з метою дослідження його ефективності;

в) виконання і модифікація ПП у процесі експлуатації за умови збереження його цілісності;

г) маркетинг ПП у процесі його ефективних продаж.

33. Що таке реінженерія програмного продукту (ПП):

а) удосконалення застарілого ПП шляхом його реорганізації або реструктуризації, а також перепрограмування окремих еле-

ментів або настроювання параметрів на іншу платформу, середовище виконання зі збереженням зручності його супроводу;

б) відновлення специфікації (графічних нотацій, потоків даних та ін.) за отриманим кодом системи для її аналізу на більш високому рівні;

в) реорганізація коду для поліпшення характеристик і показників якості об'єктно-орієнтованих і компонентних програм без зміни їхньої поведінки;

г) наповнення ПП сигнальною інформацією для зручності сприйняття супровідної документації.

34. Що таке реверсна інженерія програмного продукту (ПП):

а) удосконалення застарілого ПП шляхом його реорганізації або реструктуризації; перепрограмування окремих елементів або настроювання параметрів на іншу платформу, середовище виконання;

б) відновлення специфікації (графічних нотацій, потоків даних та ін.) за отриманим кодом системи для її аналізу на більш високому рівні;

в) реорганізація коду для поліпшення характеристик і показників якості об'єктно-орієнтованих і компонентних програм без зміни їхньої поведінки;

г) наповнення ПП сигнальною інформацією для зручності сприйняття супровідної документації.

35. Що таке рефакторинг програмного продукту (ПП):

а) удосконалення застарілого ПП шляхом його реорганізації або реструктуризації, а також перепрограмування окремих елементів або настроювання параметрів на іншу платформу, середовище виконання зі збереженням зручності його супроводу;

б) відновлення специфікації (графічних нотацій, потоків даних та ін.) за отриманим кодом системи для її аналізу на більш високому рівні;

в) реорганізація коду для поліпшення характеристик і показників якості об'єктно-орієнтованих і компонентних програм без зміни їхньої поведінки;

г) наповнення ПП сигнальною інформацією для зручності сприйняття супровідної документації.

36. Що являє собою база даних:

а) поіменована, структурована сукупність взаємопов'язаних даних, які забезпечують зв'язок користувачів та прикладних програм;

б) поіменована, структурована сукупність взаємопов'язаних даних, які характеризують окрему предметну область і перебувають під управлінням СУБД;

в) сховище незалежних даних, яке призначене для використання багатьма споживачами і забезпечує незалежність даних від прикладних програм;

г) неіменована, неструктурована сукупність незалежних даних, які характеризують конкретну предметну область.

37. Що являє собою система управління базою даних (СУБД) :

а) комплекс програмних і мовних засобів, необхідних для створення БД, підтримки її в актуальному стані, маніпулювання даними, організації доступу до них різних користувачів чи прикладних програм в умовах чинної технології обробки даних;

б) комплекс програмних і мовних засобів, необхідних для створення БД та організації доступу до неї різних користувачів чи прикладних програм в умовах прийнятої технології обробки даних;

в) комплекс програмних і мовних засобів загального та спеціального призначення, необхідних для управління БД;

г) комплекс програмних і мовних засобів, необхідних для підготовки та контролю даних, забезпечення їхньої цілісності та секретності, перетворення даних у форму, зручну для подальшого використання, ведення зв'язку з користувачем.

38. Що таке сховище даних:

а) предметно-орієнтований, інтегрований, незмінний набір даних, що підтримує хронологію і здатний бути комплексним

джерелом достовірної інформації для оперативного аналізу та прийняття рішень;

б) впорядкований набір логічно взаємопов'язаних даних, що використовуються спільно та призначені для задоволення інформаційних потреб користувачів;

в) інформація для всіх СППР, яку можна використовувати для заповнення сховищ окремих предметних областей і вітрин даних;

г) певна підмножина корпоративних даних, які характеризують конкретний аспект діяльності корпорації, наприклад роботу конкретного підрозділу.

39. Як називаються таблиці у моделі СД (типу «сніжинка»), що приєднуються до таблиць вимірів:

а) таблицями-нащадками;

б) батьківськими;

в) користувальницькими;

г) консольними.

40. Що передбачає багатовимірна модель СД типу «сніжинка»:

а) нормалізацію таблиць вимірів;

б) нормалізацію таблиць фактів;

в) ненормалізовану таблицю вимірів;

г) ненормалізовану таблицю фактів.

41. Що є основними елементами даних, які зберігаються в СД:

а) факти, міри та індикатори продуктивності;

б) продаж, прогнози продажу, фінансові дані;

в) фінансові дані, їхні виміри та ієрархія фактів;

г) показники (змінні), виміри та їхня ієрархія, факти.

42. Які рівні включає розширена архітектура подання даних інформаційної системи:

а) зовнішній, інфологічний, концептуальний, внутрішній, фізичний;

б) зовнішні, концептуальні і фізичні рівні;

в) фізичні рівні і рівні зовнішніх моделей;

г) зовнішній, інфологічний, концептуальний, внутрішній, реалізації.

43. Що являє собою зовнішній рівень подання даних інформаційної системи:

а) процеси і документи предметної області;

б) процеси предметної області і моделі даних СУБД;

в) документи предметної області і моделі даних СУБД;

г) процеси і документи предметної області, моделі даних СУБД.

44. Що визначає інфологічний рівень подання даних інформаційної системи:

а) інформаційно-логічна модель предметної області, в якій відображено інформаційні особливості об'єкта управління з урахуванням особливостей і специфіки конкретної СУБД;

б) інформаційно-логічна модель предметної області, в якій відображено інформаційні особливості об'єкта управління без урахування особливостей і специфіки конкретної СУБД;

в) інформаційно-логічна модель предметної області, в якій відображено інформаційні особливості об'єкта управління з урахуванням особливостей технології обробки даних;

г) інформаційно-логічна модель предметної області, в якій відображено інформаційні особливості об'єкта управління з урахуванням особливостей доступу до даних.

45. Що являє собою концептуальний рівень БД інформаційної системи:

а) концептуальна модель даних, яка містить структури зберігання даних у пам'яті ЕОМ;

б) структурована модель предметної області, яка підтримується засобами програмного забезпечення ЕОМ;

в) концептуальна модель даних, яка відповідає особливостям і обмеженням конкретної СУБД;

г) структурована модель предметної області, яка містить характеристики об'єктів і зв'язки між ними.

46. Що являє собою внутрішній рівень бази даних інформаційної системи:

а) опис форматів у записі даних; фізичне упорядкування даних; розміщення даних за типами пристроїв; шляхи доступу до даних;

б) структури зберігання бази даних у пам'яті ЕОМ; фізичне упорядкування даних; характеристики даних; шляхи доступу до даних;

в) опис форматів даних; розміщення даних за типами пристроїв; фізичне упорядкування даних; опис атрибутів даних;

г) структури представлення бази даних, що формуються засобами опису, та маніпулювання СУБД.

47. Що являє собою рівень реалізації бази даних як системи її фізичної організації:

а) дані, розташовані у файлах або сторінкових структурах, розташованих на зовнішніх носіях інформації під управлінням безпосередньо засобами збереження СУБД;

б) дані, розташовані у файлах або сторінкових структурах, розташованих на зовнішніх носіях інформації під управлінням безпосередньо самою ОС;

в) дані, розташовані у файлах або сторінкових структурах, розташованих в оперативній пам'яті під управлінням безпосередньо засобами збереження СУБД;

г) дані, розташовані у файлах або сторінкових структурах, розташованих в оперативній пам'яті під управлінням безпосередньо самою ОС.

48. Як може бути представлена інфологічна модель БД:

а) теоретико-графовими моделями;

б) моделями сутність – зв'язок (ER – моделі);

в) теоретико-множинними моделями;

г) графово-множинними моделями.

49. Що являє собою первинний ключ таблиці БД:

а) стовбець або мінімальна підмножина стовбців, які унікально і єдиним чином визначають рядки;

б) стовбець, який є частиною зовнішнього ключа таблиці;

в) стовбець, який є зовнішнім ключем таблиці;

г) стовбець, значення якого унікальне для таблиці.

50. Які з перерахованих засобів є системами управління базами даних (СУБД):

а) MS Access, Oracle, MySQL, C++ Builder;

б) MS Word, Oracle, MySQL, MS Access;

в) MS SQL Server, Oracle, MySQL, MS Access;

г) C++ Builder, Oracle, MySQL, MS Access.

51. Які з поданих нижче типів є об'єктами реляційної моделі даних:

а) n-арні нормалізовані відношення;

б) таблиці;

в) агрегати даних і групові відношення;

г) множини.

52. Що являє собою реквізит:

а) елемент даних, який не має самостійного змісту;

б) інформаційна сукупність найнижчого рангу, яка не підлягає поділу на одиниці інформації;

в) структурна одиниця, яка характеризує будь-який конкретний об'єкт управління з кількісного та якісного боку;

г) сукупність даних, об'єднаних за змістовим принципом.

53. На основі якої теорії заснована технологія проектування реляційної БД:

а) теорії нормалізації;

б) теорії чисел;

в) теорії детермінації;

г) теорії відносин.

54. Що є основною структурою даних у реляційній моделі:

а) відношення;

б) набір даних;

в) агрегат даних;

г) запис.

55. Як називається технологія автоматизованого проектування інформаційних систем:

а) OLAP-технологія;

б) технологія Data Mining;

в) CASE-технологія;

г) технологія OLTP.

56. Яке з наведених визначень відповідає поняттю CASE-засоби:

а) методологія проектування ІС, а також набір інструментальних засобів, що дозволяють у наочній формі моделювати тільки предметну область;

б) методологія проектування ІС, а також набір інструментальних засобів, що дозволяють аналізувати представлену модель на всіх етапах розробки і супроводу;

в) методологія проектування ІС, а також набір інструментальних засобів, за допомогою яких можливо лише розробити програми відповідно до інформаційних потреб користувачів;

г) методологія проектування ІС, а також набір інструментальних засобів, що дозволяють у наочній формі моделювати предметну область, аналізувати цю модель на всіх етапах розробки і супроводу ІС та розробляти програми відповідно до інформаційних потреб користувачів.

57. На якому етапі (етапах) розробки ІС CASE-засоби забезпечують якість прийнятих технічних рішень та підготовку проектної документації:

а) етап проектування, етап аналізу;

б) етап проектування, етап реалізації;

в) етап реалізації, етап впровадження;

г) етап впровадження, етап супроводу.

58. Що являє собою методологія структурного аналізу та проектування:

а) методи і засоби функціонального аналізу, опису специфікацій вимог та процесів системи;

б) методи обробки статистичних даних проектованої системи, опису специфікацій вимог;

в) аналіз стратегічного розподілу ресурсів організації у процесі проектування програмних систем;

г) методи обробки та аналізу даних, що надходять від діяльності організації.

59. Що є основою взаємозв'язку між структурним і об'єктно-орієнтованим підходами:

а) спільність ряду категорій і понять обох підходів (процес і варіант використання, сутність і клас);

б) методи реалізації програмних продуктів, форма прояву взаємозв'язку;

в) засоби реалізації великомасштабних програмних продуктів;

г) принципи реалізації, супроводу та зняття з експлуатації програмних продуктів.

60. Для чого призначена функціональна модель:

а) вирішення проблем з ресурсним забезпеченням підприємства;

б) вирішення проблем з функціонуванням підприємства;

в) опис існуючих бізнес-процесів та передачі інформації про конкретну систему;

г) вирішення проблем з інформаційного супроводу підприємства.

61. Які властивості графічної мови моделювання систем:

а) просте вивчення і розуміння полегшує взаємодію та взаєморозуміння системних аналітиків, розробників та персоналу досліджуваного об'єкта;

б) мова не генерується набором інструментальних засобів машинної графіки, а є текстовим представленням контекстної моделі;

в) досить громіздка, потрібно уміння, щоб лаконічно описати процес, який моделюється;

г) має багато функціональних можливостей для представлення процесів у часі.

62. Яка із запропонованих нижче нотацій не використовується для графічного представлення вимог до програмного продукту:

а) IDEF0;

б) DFD;

в) UML;

г) SQL.

63. На яких методологіях засновано CASE-засоби:

а) структурного й об'єктно-орієнтованого аналізу і проектування;

б) подієвого й об'єктно-орієнтованого аналізу і проектування;

в) подієвого і структурного аналізу та проектування;

г) подієвого і статистичного аналізу.

64. Як називаються інструментальні засоби, що підтримують процеси формулювання й аналізу вимог, проектування, створення і супроводу програмних продуктів:

а) ВЕБ-засоби;

б) CASE-засоби;

в) інформаційні засоби;

г) програмні засоби.

65. За яким принципом працює програмне забезпечення AllFussion Modeling Suite від корпорації Computer Associates:

а) комунікабельності та інтерпретації;

б) комплексності та взаємної інтеграції;

в) довговічності і надійності;

г) взаємопов'язаності й ефективності.

66. Які відомі інструменти для моделювання, аналізу, документування та оптимізації бізнес-процесів:

а) All Fussion Process Modeler CA, Rational Software Architect;

б) системне програмне забезпечення;

в) Microsoft Office;

г) CASE-технології.

67. Який засіб CASE-пакета структурного аналізу і проектування AllFussion Modeling Suite дозволяє аналізувати структуру БД з метою виявлення недоліків і помилок проектування:

а) AllFusion ERwin Data Modeler;

б) AllFusion Data Model Validator;

в) AllFusion Model Manager;

г) AllFusion Process Modeler.

68. Який засіб CASE-пакета структурного аналізу і проектування AllFussion Modeling Suite дозволяє проведення аналізу і реорганізації бізнес-процесів:

а) AllFusion ERwin Data Modeler;

б) AllFusion Data Model Validator;

в) AllFusion Model Manager;

г) AllFusion Process Modeler.

69. Який засіб CASE-пакета структурного аналізу і проектування AllFussion Modeling Suite використовується для проектування та документування БД:

а) AllFusion ERwin Data Modeler;

б) AllFusion Data Model Validator;

в) AllFusion Model Manager;

г) AllFusion Process Modeler.

70. Що являє собою IDEF0:

а) методологія функціонального моделювання і графічного описання процесів з метою формалізації та опису бізнес-процесів;

б) методологія опису системи в цілому як множина взаємозалежних дій або функцій;

в) методологія опису системи з детальним поглибленням у процеси взаємодії елементів системи;

г) методологія, що проводить реструктуризацію системи та згодом її перебудову.

71. Які компоненти синтаксису графічної мови IDEF0:

а) контекстна діаграма, зв'язки, тунелі;

б) блоки, стрілки, синтаксичні правила;

в) діаграми, зв'язки, тунелі;

г) контекстна діаграма, зв'язки, синтаксичні правила.

72. Яку назву мають стрілки IDEF0-діаграм:

а) вхід, правила, вихід, механізм виконання;

б) вхід, управління, вихід, механізм виконання;

в) вхід, управління, результат, механізм виконання;

г) вхід, правила, результат, механізм виконання.

73. Які синтаксичні правила для блоків:

а) кожна сторона функціонального блоку повинна мати стандартне відношення блок/стрілки;

б) ім'я блоку має бути дієсловом або віддієслівним іменником;

в) блоки можуть розташовуватися у довільному порядку;

г) блок повинен бути прямокутником.

74. Які синтаксичні правила для стрілок:

а) ламані стрілки змінюють напрямок тільки під кутом 60°, при цьому сегменти стрілок повинні позначатися іменником або оборотом іменника;

б) стрілки повинні бути намальовані розривними лініями різної товщини;

в) стрілки можуть складатися тільки з вертикальних або горизонтальних відрізків (відхилення не допускаються) і позначаються іменником або оборотом іменника;

г) стрілки можуть торкатися внутрішнього кордону функціонального блока, можуть його перетинати, повинні обов'язково бути названими.

75. Що являє собою декомпозиція:

а) дослідження процесів у функціональній моделі та формування відповідної документації;

б) виділення додаткових блоків у роботі та формування відповідної документації;

в) дослідження зав'язків між блоками та формування відповідної документації;

г) розбиття складного процесу на складові, реалізується в процесі поступового представлення моделі системи у вигляді ієрархічної структури окремих діаграм.

76. Що таке контекстна діаграма:

а) діаграма, що відображає декомпозицію функціонального блоку;

б) діаграма, що відображає систему в цілому;

в) діаграма, що відображає процеси, які протікають у системі;

г) контекстна діаграма в поєднанні з переліком системних вимог відображає завдання «Що робить система».

77. У чому сутність методології IDEF3:

а) методологія опису процесів, що дозволяє подати ситуацію як упорядковану послідовність подій з одночасним описом об'єктів, що мають безпосереднє відношення до процесу;

б) технологія моделювання бізнес-процесів із жорсткими синтаксичними або семантичними обмеженнями, які роблять зручним опис неповних або нецілісних систем;

в) технологія, яка добре пристосована для збору та формалізації даних, потрібних для проведення структурного аналізу системи;

г) технологія, яка може бути використана як метод проектування бізнес-процесів.

78. Що виділяє сценарій бізнес-процесу в моделі IDEF3:

а) послідовність дій або підпроцесів аналізованої системи;

б) тільки можливість створення нових процесів;

в) послідовність дій та потоків даних;

г) тільки послідовність дій.

79. Для чого використовуються в діаграмах IDEF3 перехрестя (з'єднання):

а) показати взаємини робіт і участь деякого об'єкта в двох або більше роботах;

б) показати розгалуження логічної схеми модельованого процесу й альтернативні шляхи його розвитку;

в) показати зв'язок між двома роботами або між роботою й об'єктом посилання;

г) показати участь деякого об'єкта в двох або більше роботах.

80. Що є основною одиницею роботи в моделі IDEF3:

а) операція;

б) перехід;

в) крок;

г) дія.

81. Як ініціюється асинхронне «АБО» у випадку злиття стрілок:

а) один або декілька наступних процесів повинні бути запущені;

б) один або декілька попередніх процесів повинні бути завершені;

в) один або декілька наступних процесів повинні бути запущені одночасно;

г) один або декілька попередніх процесів повинні бути завершені одночасно.

82. Як ініціюється асинхронне «І» у випадку розгалуження стрілок:

а) усі наступні процеси повинні бути запущені;

б) усі попередні процеси повинні бути завершені;

в) усі наступні процеси повинні бути запущені одночасно;

г) усі попередні процеси повинні бути завершені одночасно.

83. Як ініціюється виключне «АБО» у випадку злиття стрілок:

а) тільки один наступний процес запускається;

б) тільки один попередній процес завершений;

в) один або декілька наступних процесів запускається;

г) один або декілька попередніх процесів завершені.

84. Що являє собою діаграма потоків даних:

а) інфологічна модель БД в інформаційній системі;

б) концептуальна модель БД в інформаційній системі;

в) модель проектування, графічне представлення «потоків» даних в інформаційній системі;

г) модель проектування бізнес-процесів.

85. Які поняття лежать в основі DFD-моделі:

а) зовнішньої сутності, процесу, сховища (накопичувача) даних і потоку даних;

б) процесу, потоку даних, стрілок;

в) зовнішньої сутності, процесу, стрілок, потоку даних;

г) входи, внутрішні потоки даних і виходи системи.

86. Які види нотацій використовують для зображення діаграм потоків даних:

а) нотації Йордана і Гейна-Сарсона;

б) нотації Сарсона та Сейнса;
в) нотації Лоуренса та Йордана;
г) нотації Кона та Сейнса.

87. З чого складається контекстна DFD-діаграма:
а) двох функціональних блоків;
б) декількох зовнішніх сутностей;
в) одного функціонального блоку та кількох сутностей;
г) багатьох блоків і однієї сутності.

88. Чим відрізняються функціональні блоки DFD від функціональних блоків IDEF0:
а) не мають управління і механізму виконання;
б) мають входи і виходи;
в) моделюються як ресурси і їх потоки;
г) мають стрілки зв'язку між блоками.

89. Що являє собою зовнішня сутність у DFD:
а) матеріальний об'єкт, організацію або фізичну особу, яка є джерелом або приймачем інформації;
б) об'єкт, який надає інформацію клієнтам;
в) об'єкт, який приймає інформацію від замовника;
г) об'єкт або система як зовнішня сутність, яка знаходиться за межами аналізованої інформаційної системи.

90. Як позначається зовнішня сутність у DFD:
а) прямокутником із заокругленими краями;
б) прямокутником;
в) ромбом;
г) овалом.

91. Що визначає «потік даних» у DFD:
а) інформацію, яка передається від входу до виходу;
б) інформацію, передану через довільне з'єднання від джерела до приймача;

в) інформацію, що передається кабелем між двома пристроями;

г) інформацію, що пересилаються поштою листами.

92. Для чого слугують сховища даних у DFD:

а) для опису даних, що тимчасово не використовуються, знаходяться в незмінному, нерухомому стані, зберігаються деякий час;

б) для даних, що можуть використовуватись для внутрішньої обробки;

в) для оброблення даних і можливого надання;

г) для зберігання й аналізу інформації, що не використовується в цей час.

93. У яких областях спостерігається стійке становище успішного застосування CASE-технологій:

а) розважального програмного забезпечення;

б) ділового, комерційного і системного програмного забезпечення;

в) апаратного забезпечення;

г) прикладного програмного забезпечення.

94. Які компоненти не містить CASE-засіб:

а) засоби статистичної обробки даних;

б) засоби моделювання;

в) засоби реінжинірингу;

г) засоби аналізу.

95. Які з типів CASE-засобів є основними:

а) засоби планування й управління проектом;

б) засоби аналізу, проектування, розробки додатків, реінжинірингу;

в) засоби конфігураційного управління;

г) засоби тестування.

96. Яка компонента є найбільш актуальною для ефективного використання CASE-засобів із розробки ІС:

а) велика кількість часу;

б) дорога матеріальна база;

в) велика кількість коштів;

г) висока кваліфікація фахівців.

97. Що являє собою CASE-технологія:

а) методологія проектування ІС, використання інструментальних засобів для моделювання предметної області, аналізу моделей на всіх етапах розробки і супроводу ІС;

б) використання інструментальних CASE-засобів у процесі програмування;

в) методологія проектування ІС на принципах програмної інженерії;

г) використання інструментальних CASE-засобів у процесі програмування відповідно до інформаційних потреб користувачів.

98. Які основні напрями має розвиток програмного забезпечення (ПЗ):

а) удосконалення наявного та розробка принципово нового ПЗ;

б) зменшення витрат на впровадження нового ПЗ;

в) гарантування великої конкуренції на ринку;

г) розвиток вільного ПЗ.

99. Які CASE-інструменти є типовими:

а) інструменти для побудови діаграм, аналізу, проектування, генерування та редагування програмного коду;

б) інструменти створення технічної документації;

в) інструменти створення програмного коду;

г) інструменти обробки даних.

100. За яким принципом працює програмне забезпечення All Fussion Suite від корпорації Computer Associates:

а) комунікабельності та інтерпретації;

б) комплексності та взаємної інтеграції;

в) довговічності і надійності;

г) взаємопов'язаності й ефективності.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Вендров А. М. CASE-технологии. Современные методы и средства проектирования информационных систем / А. М. Вендров. – М., 2004. – 346 с.

2. Гайджарги В. І. Основи проектування та використання баз даних : навч. посібн. / В. І. Гайджарги, О. А. Дацюк. – 2-е вид. – К. : ІВЦ «Видавництво Політехніка», 2004. – 256 с.

3. Галузевий стандарт вищої освіти України з напряму підготовки 6.050101 «Комп'ютерні науки» : збірник нормативних документів вищої освіти. – К. : Видавнича група BHV, 2011. – 85 с.

4. Грабер М. SQL / М. Грабер ; пер. с англ. – М. : Лори, 2003. – 644 с.

5. Калянов Г. Н. CASE-технологии. Колсалтинг в автоматизации бизнес процессов / Г. Н. Калянов. – 3-е изд. – М. : Горячая линия-Телеком, 2002. – 3260 с. : [ил.].

6. Ковалюк Т. В. Професійний стандарт. Фахівець з розробки програмного забезпечення [Електронний ресурс] / Т. В. Ковалюк, О. А. Орєхов , О. П. Сирота. – Режим доступу : net/documents/standarts.pdf.

7. Конноли Т. Базы данных. Проектирование, реализация и сопровождение. Теория и практика / Томас Конноли, Каролин Бегг. – 3-е изд.; пер. с англ. – М. : Вильямс, 2003. – 1440 с.

8. Кравченко В. Г. Проектування автоматизованих інформаційних систем : навч. посіб. / В. Г. Кравченко. – К. : КНЕУ, 2008. – 360 с.

9. Лавріщева К. М. Програмна інженерія : підручник / К. М. Лавріщева. – К. : Академперіодика НАН України, 2008. – 319 с.

10. Маклаков С. В. Создание информационных систем с AllFusion Modeling Suite / С. В. Маклаков. – 2-е изд., испр. и дополн. – М. : Издательство Диалог-МИФИ, 2007. – 400 с.

11. Основи програмної інженерії (по SWEBOK) [Електронний ресурс]. – Режим доступу : www.swebok.sorlik.ru/software_engineering.html.

12. Пасічник В. В. Сховища даних / В. В. Пасічник. – Львів, 2013. –416 с.

13. Постіл С. Д. Інтеграція дисциплін у процесі міждисциплінарного інформаційного моделювання / С. Д. Постіл // Науковий вісник Національного університету державної податкової служби України. – 2014. – № 3 (61). – С. 68–76.

14. Постіл С. Д. Інформаційні системи і технології обліку : навч. посібник / Т. С. Іванова, С. Д. Постіл, І. О. Сафронова ; за ред. С. П. Ріппи. – Ірпінь : НУДПС України, 2010. – 266 с.

15. Проектування інформаційних систем : посібник / за ред. В. С. Пономаренка. – К. : Академія, 2002. – 486 с.

16. Ситник Н. В. Проектування баз і сховищ даних : навч. посібник / Н. В. Ситник. – К. : КНЕУ, 2004. – 348 с.

17. Ситник Н. В. Проектування баз і сховищ даних : навч.-метод. посіб. для самост. вивч. дисц. / Н. В. Ситник, М. Т. Краснюк. – К. : КНЕУ, 2005. – 264 с.

18. Федотова Д. Э. CASE-технологии : практикум / Д. Э. Федотова, Ю. Д. Семенов, К. Н. Чижик. – М. : Горячая линия–Телеком, 2003. – 160 с.

19. Curricula Recommendations [Електронний ресурс]. – Режим доступу : www.acm.org/education/education/curricula-recommendations

ТЕРМІНОЛОГІЧНИЙ СЛОВНИК

Автоматизована інформаційна технологія – системно організована для розв'язання задач управління сукупність методів і засобів реалізації операцій збору, реєстрації, передавання, нагромадження, пошуку, обробки та захисту інформації на основі застосування сучасного програмного забезпечення, використовуючи засоби обчислювальної техніки та зв'язку, а також засоби, за допомогою яких інформація пропонується клієнтам.

Багатовимірне сховище даних – підтримує багатовимірну модель даних на концептуальному рівні та призначено для інтерактивної аналітичної обробки агрегованих даних в OLAP-системах.

База даних – пойменована сукупність структурованих даних, що належать до певної предметної області.

База даних реляційна – база даних, реалізована згідно з реляційною моделлю даних.

Дані – інформація, подана у формалізованому вигляді, придатному для переміщення, інтерпретації або обробки за участю людини чи автоматичними засобами.

Життєвий цикл розробки складної програмної системи складається з етапів аналізу, проектування, програмування і тестування, впровадження і супроводу, які виконуються послідовно.

Зв'язок – функціональна залежність між таблицями бази даних.

Інформаційно-логічна або інфологічна модель – відображає інформаційні сутності, їхні властивості та зв'язки між ними без вказівки способів фізичного зберігання інформації (*модель предметної області*).

Інформаційна система – програмно-апаратний комплекс, призначений для зберігання й обробки інформації будь-якої предметної області.

Ключовий елемент таблиці – таке її поле (простий ключ) або конкатенація декількох полів (складений ключ).

Код – це знак чи сукупність знаків, прийнятих для позначення класифікаційного групування чи об'єкта класифікації.

Користувачі ІС: а) бізнес-аналітики, що вивчають і моделюють бізнес-процеси предметної області; б) системні аналітики й архітектори, що проектують архітектуру рішення, додатків і даних; в) автори коду додатків; г) фахівці з тестування й оцінки якості; д) автори документації; е) автори дистрибутивів; є) фахівці із впровадження.

Методологія документування процесів (IDEF3), що відбуваються в системі, дозволяє описати сценарій і послідовність операцій для кожного процесу.

Методологія проектування реляційних баз даних (IDEF1X) дозволяє побудувати моделі даних типу «сутність-зв'язок» у нотації цього стандарту.

Методологія функціонального моделювання складних систем (IDEF0) за допомогою наочної графічної мови представляє досліджувану систему у вигляді набору взаємозалежних функціональних блоків.

Мова SQL (Structured Query Language) – універсальна мова роботи з базами даних, що включає можливості її створення, модифікації структури, відбору даних за запитами, модифікації інформації в базі та інші операції маніпулювання базою даних.

Первинний ключ – головний ключовий елемент, що однозначно ідентифікує рядок у таблиці.

Предметна область – деяка частина реально існуючої системи, що функціонує як самостійна одиниця.

Реквізит – найпростіша структурна одиниця інформації, яка є неподільною на смисловому рівні. Він буває двох видів: а) реквізит-ознака, що містить якісну характеристику процесу або документа; б) реквізит-основа, що містить кількісну характеристику процесу або документа.

Реляційна БД – основний тип сучасних баз даних. Складається з таблиць, між якими можуть існувати зв'язки за ключовим значенням.

Система управління базами даних – комплекс програмних і мовних засобів, необхідних для створення й модифікації бази

даних, додавання, модифікації, видалення, пошуку й відбору інформації, подання інформації на екрані й у друкованому виді, розмежування прав доступу до інформації, виконання інших операцій з базою.

Сутність – відособлені об'єкти або події, інформацію про які необхідно зберігати, що мають певні набори властивостей-реквізитів.

Таблиця бази даних – регулярна структура, що складається з однотипних рядків (записів), розбитих на стовпці (поля). У теорії реляційних баз даних синонім таблиці – відношення, у якому рядок називається кортежем, а стовпець – доменом.

Універсальна мова моделювання (UML) для специфікації, візуалізації, конструювання й документування складних інформаційно-насичених об'єктних систем з використанням діаграм.

Файл – це сукупність однорідної інформації за складом і послідовністю полів, записаної на магнітному диску з присвоєнням імені.

Формалізація даних – зведення даних, що надходять з різних джерел, до однакової форми, щоб зробити їх зіставними між собою.

ДОДАТКИ

Додаток А

ЗАГАЛЬНІ РЕКОМЕНДАЦІЇ ДО ВИКОНАННЯ ЛАБОРАТОРНИХ РОБІТ ТА ОФОРМЛЕННЯ ЗВІТУ

Додаток А.1

І. Особливості проведення лабораторних робіт

1. Виконання лабораторної роботи супроводжується методичними рекомендаціями, з якими необхідно ознайомитися заздалегідь.

2. Теоретичний матеріал за темою роботи поданий необхідною мірою в цьому посібнику; за потреби звернутись до інших джерел (бібліотека університету чи Інтернет) для вироблення пошукового навику та належної підготовки роботи з літературою.

3. Впроваджена диференційована форма організації лабораторної роботи на основі індивідуальних завдань.

4. Завдання визначається предметом дослідження, який відповідає темі лабораторної роботи навчальної дисципліни, для індивідуального об'єкта дослідження (*додаток Г*).

5. Індивідуальний об'єкт дослідження формується таким чином:

✓підбирається викладачем на початку вивчення дисципліни з урахуванням здібностей студента (*темп та ритм навчальної діяльності, рівень сформованості умінь і навичок самостійної роботи, теоретичної підготовленості*);

✓узгоджується зі студентом;

✓розміщується в методичній розробці відповідно до порядкового номера студента в журналі;

✓при вивченні наступних дисциплін не змінюється.

5. У процесі виконання лабораторної роботи переважно використовують такі методи, як аналіз, синтез, порівняння, діагностування, висновки.

ІІ. Вимоги до оформлення звіту

1. Оформити звіт за допомогою MS Word, де повинні найти відображення:

а) титульний листок;

б) прізвище, ініціали і номер групи студента;

в) тема і мета роботи;

г) результати відповідей на зауваження викладача:

✓у 1-й версії звіту – пропуск;

✓у наступних версіях звіту – копія зауваження викладача;

✓зміни до звіту, які були внесені відповідно до попередньої версії;

✓остання версія звіту.

д) результати виконання всіх пунктів «Завдання» з детальним описом та представленням екранних вікон *(за наявності)*;

є) відповідь на контрольні запитання;

ж) висновок щодо:

✓досягнення мети;

✓виконання завдання;

✓зауваження та рекомендації.

2. Робота повинна бути представлена в електронному вигляді з іменем файла (текстового кирилицею та архівного довжиною до 40 символів):

«Прізвище_Ініціали_Тип-роботи_№Роботи_Група_Предмет_Версія-файла».

3. Звіт з роботи, який не відповідає всім вимогам до його змісту і вимог оформлення, буде відхилений.

4. Розроблений проект захистити і представити для оцінювання.

Додаток Б

Додаток Б.1

№ з/п	1	2	3	4	5	6	7	8	9	10
Код освітньої спеціальності	7(8).05010101	7(8).05010102	7(8).05010103	7(8).05010104	7(8).05010201	7(8).05010202	7(8).05010203	7(8).05010301	7(8).05010302	7(8).04030202

№ з/п	11	12	13	14	15	16	17	18	19	20
Код освітньої спеціальності	7(8).04030204	6.050101	6.050102	6.050103	6.40303	6.40302	5.5010101	5.5010102	5.5010201	5.5010301

Додаток Б.2

№ з/п	1	2	3	4	5	6	7	8	9	10
Код завдання діяльності	УР1	УР2	УР3	УР4	УР5	УР6	УР7	УР8	УР9	УР10

№ з/п	11	12	13	14	15	16	17	18	19	20
Код завдання діяльності	УР11	РВ1	РВ2	ПФ1	СП1	СП2				

Додаток Б.3

№ з/п	1	2	3	4	5	6	7	8	9	10	11	12
Код трудової функції	УР 1.1	УР 1.2	УР 1.3	УР 2.1	УР 2.2	УР 2.3	УР 2.4	УР 3.1	УР 4.1	УР 5.1	УР 5.2	УР 5.3

№ з/п	13	14	15	16	17	18	19	20	21	22	23	24
Код трудової функції	УР 5.4	УР 6.1	УР 6.2	УР 7.1	УР 8.1	УР 8.2	УР 9.1	РВ 1.1	РВ 1.2	ПФ 1.1	ПФ 1.2	ПФ 1.3

Додаток В

БАЗА ДАНИХ (БД)

STUDENT

складається з таблиць ***OSOBA, VYDCTRL, FACULTET, USPISHNIST.***

ТАБЛИЦЯ 1 OSOBA містить інформацію про особові справи студентів і має поля:

***ID_OSOBA, PRIZ, NAME, POBAT, DNAROD, ID_FACULT, GRP*:**

1. Поле **ID_OSOBA** має тип ***INT*** і містить ідентифікаційний код.
2. Поле **PRIZ** має тип ***CHAR[50]*** і містить прізвище студента.
3. Поле **NAME** має тип ***CHAR[50]***і містить ім'я студента.
4. Поле **POBAT** має тип ***CHAR[50*** і містить по батькові студента.
5. Поле **DNAROD** має тип ***DATE*** і містить дату народження.
6. Поле **ID_FACULT** має тип ***INT*** і містить ідентифікаційний код інституту, в якому він навчається.
7. Поле **GRP** має тип ***CHAR[10]*** і містить назву групи.

ТАБЛИЦЯ 2 VYDCTRL містить інформацію про види контрольних заходів і має поля ***ID_VYDCTRL*** і ***NAZVA:***

1. Поле **ID_VYDCTRL** має тип ***INT*** і містить код типу контрольного заходу.
2. Поле **NAZVA** має тип ***CHAR[50]*** і містить назву типу контрольного заходу (*модульна* контрольна робота, *підсумкова* контрольна робота, *акредитаційна* контрольна робота, *залік, екзамен*, тощо).

ТАБЛИЦЯ 3 **FACULTET** має поля ***ID_FACULT*** і ***NAZVA***:

1. Поле **ID_FACULT** має тип ***INT*** і містить ідентифікаційний код інституту (факультету).
2. Поле **NAZVA** має тип ***CHAR[50]*** і містить повну назву інституту (факультету).

ТАБЛИЦЯ 4 USPISHNIST містить інформацію про поточну успішність студентів і має такі поля:

ID_OSOBA, DTCONROL, ID_VYDCTRL, OCINKA, PRESENT:

1. Поле **ID_OSOBA** має тип ***INT*** і містить ідентифікаційний код.
2. Поле **DTCONROL** має тип ***DATE*** і містить дату проведення заходу.
3. Поле **ID_VYDCTRL** має тип ***INT*** і містить код типу контрольного заходу (*модульна* контрольна робота, *підсумкова* контрольна робота, *акредитаційна* контрольна робота, *залік, екзамен* тощо).
4. Поле **OCINKA** має тип ***INT*** і містить оцінку (1, 2,..5) за відповідний контрольний захід;
5. Поле **PRESENT** має тип ***INT*** і приймає два значення:
   0 – студент був відсутнім на контрольному заході і
   1 – був присутнім на контрольному заході.

Додаток Г

Варіанти завдань 1–23

1. Об'єкт досліджування, БД якого дозволяє отримати звітну відомість і реалізувати запити:

Абонент							
Квартира			Телефон			Прізвище	Адреса
Сума оплати	Сума субсидій	Борг	Сума оплати	Борг	Період неплатежів		

1) визначити прізвища й адреси боржників з максимальною сумою боргу з квартплати і за телефон;

2) вивести загальну суму субсидій за зазначеними адресами;

3) визначити телефонних боржників, у яких максимальний період неплатежів.

2. Об'єкт досліджування, БД якого дозволяє отримати звітну відомість і реалізувати запити:

Бібліотечний каталог											
Книги			Журнали					Газети			
Автор	Назва	Рік	Назва	Номер	Рік	Автор	Стаття	Автор	Назва	Стаття	Дата

1) вивести список робіт уведеного автора;

2) визначити кількість статей кожного автора, які друкуються в зазначеному журналі протягом зазначеного періоду років;

3) визначити газети, що друкувались до 2013 року.

3. Об'єкт досліджування, БД якого дозволяє отримати звітну відомість і реалізувати запити:

Будинок						
Місцевість	Приватний			Багатоквартирний		
	Кількість кімнат	Кількість мешканців	Розмір ділянки	Кількість поверхів	Кількість квартир на поверсі	Прибудови (магазин тощо)

1) визначити кількість мешканців у всіх приватних будинках у заданій місцевості;

2) визначити кількість квартир у всіх багатоквартирних будинках у заданій місцевості;

3) визначити приватні будинки з ділянкою менше 13 соток.

4. Об'єкт досліджування, БД якого дозволяє отримати звітну відомість і реалізувати запити:

Вид транспорту								
Місцевість	Залізниця		Автомобільна дорога		Авіа		Довжина	Пропускна здатність
	Тип поїзда	Тепловоз, паровоз, електровоз	Вид транспорту	Тип покриття	Тип літака	Повітряний коридор		

1) визначити сумарну довжину залізниці й автодороги в заданій місцевості;

2) вивести список доріг із пропускною здатністю більше заданої;

3) визначити можливість використання літаків заданого типу в заданій місцевості.

5. Об'єкт досліджування, БД якого дозволяє отримати звітну відомість і реалізувати запити:

Викладач (навантаження)							
Професор		Доцент		Асистент		Дисциплі на	Посадова ставка
Кількість аспірантів	Кількість лекційних годин	Кількість лекцій-них годин	Кількість практичних годин	Кількість практичних годин	Виробнича практика		

1)визначити дисципліни з максимальним лекційним навантаженням, де викладають професори;

2) вивести заробітну плату викладача з урахуванням того, що посадова ставка нараховується за 800 годин. Якщо є перенавантаження викладача, то нараховується надбавка;

3) визначити, чи викладають зазначену дисципліну і скільки годин призначено на неї за різними видами занять (лекція, практичні, практика).

6. Об'єкт досліджування, БД якого дозволяє отримати звітну відомість і реалізувати запити:

Виступи									
Дата	Назва театру	Кількість глядачів	Ціна	Драмтеатр		Оперний		Естрада	
				Комедія	Трагедія	Балет	Опера	Рок група	Сольний концерт

1)визначити сумарну ціну квитків, проданих на зазначену дату на балет у зазначений театр;

2) визначити спектаклі, якщо в зазначений день естрадний концерт не відбувався;

3) визначити кількість комедій і трагедій у певному театрі за зазначений місяць.

7. Об'єкт досліджування, БД якого дозволяє отримати звітну відомість і реалізувати запити:

Кадри ВНЗ								
Викладач					Студент			
ПІП	Учений ступінь	Ставка	Термін контракту	Підвищення каліфікації	ПІП	Група	Середній бал	Спеціалізація

1) визначити викладачів-доцентів, у яких контракт закінчується в 2019 році;

2) визначити викладачів, які потребують підвищення кваліфікації;

3) визначити студентів із середнім балом більше 3.8.

8. Об'єкт досліджування, БД якого дозволяє отримати звітну відомість і реалізувати запити:

Кадровий працівник							
ПІП	Стать	Інженерно-технічний співробітник (ІТС)				Допоміжний персонал	
		Стаж	Освіта	Кваліфікація	Доступ до інформації	Професія	Розряд

1) вивести список працівників-чоловіків, у яких зарплата менше уведеної і стаж більше трьох років;

2) визначити кількість ІТС із заданою кваліфікацією і рівнем таємності – 1;

3) визначити працівників з розрядом рівним 3 і з неповною вищою освітою.

9. Об'єкт досліджування, БД якого дозволяє отримати звітну відомість і реалізувати запити:

Канцелярські товари								
Зошити		Папір		Олівці		Дата поставки	Кількість	Ціна
Кількість аркушів	Формат	Формат	Якість	Кольори корпусу	Кольори стержня			

1)визначити сумарну вартість товару, який поставлений у певну дату;

2) Визначити список зошитів з кількістю аркушів від 12 до 96, поставлених у період з 01.05.2013 по 01.07.2013;

3) визначити список паперу вищої (першої) якості.

10. Об'єкт досліджування, БД якого дозволяє отримати звітну відомість і реалізувати запити:

Каталог ігор								
Гра			Фірма		Оцінка (10-бальна)			
Назва	Жанр	Рік виходу	Видавець	Розроблювач	Сюжет	Інтелект	Графіка	Звук

1)визначити список ігор та їхній жанр, що вийшли за зазначений період заданої фірми-розроблювача;

2) вивести списки ігор вищого балу за критеріями: сюжету, інтелекту, графіки і звуку;

3) визначити гри із сюжетом нижче 3 балів.

11. Об'єкт досліджування, БД якого дозволяє отримати звітну відомість і реалізувати запити:

Комп'ютерні товари										
Дата	Ціна	Комп'ютери			Принтери			Монітори		Кількість
		Apple	Athlon	Pentium	Матричний	Струменевий	Лазерний	LCD	TFT	

1)визначити сумарну вартість комп'ютерів, доставлених на період з 01.01.2017 по 01.11.2017;

2) вивести список принтерів, закуплених за зазначений місяць 2017 року;

3) визначити дати покупки і кількості комп'ютерів зазначеного типу.

12. Об'єкт досліджування, БД якого дозволяє отримати звітну відомість і реалізувати запити:

Лікарські препарати						
Наукова медицина			Народна медицина		Назва препарату	Ціна
Хвороба	Побічні ефекти	Відсоток ускладнень	Хвороба	Відсоток успішних результатів		

1)визначити найдешевший препарат, що надає мінімальну кількість ускладнень для зазначеної хвороби.

2) вивести список ліків, що мають мінімальну кількість побічних ефектів і відповідають межі зазначеної ціни.

3) визначити список хвороб, для яких застосовується зазначений препарат, що забезпечує перевищення 83 відсотків успішних результатів.

13. Об'єкт досліджування, БД якого дозволяє отримати звітну відомість і реалізувати запити:

Навчальна програма									
Факультет	Курс	Кількість студентів	Програмування			Математика			Прізвище студента
			Java	C++	C#	Економетрія	Мат. аналіз	Теорія алгоритмів	

1)визначити факультет, на якому викладається 'Java' на першому курсі;

2) визначити кількість студентів першого курсу, що слухають дисципліну 'Теорія алгоритмів;

3) вивести список студентів факультету (інституту) «Економіка і оподаткування», що вивчають "C++'.

14. Об'єкт досліджування, БД якого дозволяє отримати звітну відомість і реалізувати запити:

Навчальний заклад								
Ліцей		ВНЗ державний		ВНЗ приватний		На з ва	Кількість учнів (студентів)	Місто
Навчальний напрям	Методика викладання	Кількість факультетів	Рівень акредитації	Оплата навчання	Ліцензія			

1) визначити середню кількість учнів і студентів у навчальних закладах зазначеного міста;

2) визначити вузи, що не мають ліцензії на викладання;

3) вивести список ВНЗ з кількістю студентів не менше 3 000 і факультетів (інститутів) більше 10.

15. Об'єкт досліджування, БД якого дозволяє отримати звітну відомість і реалізувати запити:

Науково-дослідні роботи						
Керівник				Виконавець		
ПІП	Вчена ступінь	Зарплата	Тривалість контракту	ПІП	Посадова ставка	Кількість тем

1) визначити керівників докторів наук, у яких контракт закінчується в 2018 році;

2) визначити виконавців з мінімальною посадовою ставкою;

3) визначити виконавців, які мають не менше 2-х тем під керівництвом однієї особи.

16. Об'єкт досліджування, БД якого дозволяє отримати звітну відомість і реалізувати запити:

Персональний комп'ютер (ПК)							
Фірма	Ціна	Апаратне забезпечення			Програмне забезпечення		
		Тип процесора	RAM	Периферія	Операційна система	Середовище програму-вання	Утиліти

1) вивести список комп'ютерів (процесор, RAM, периферія), що поставлені зазначеною фірмою;

2) визначити сумарну вартість програмних продуктів, що поставляються заданою фірмою;

3) визначити комп'ютери з мінімальною RAM.

17. Об'єкт досліджування, БД якого дозволяє отримати звітну відомість і реалізувати запити:

Погода							
Місце-вість	Опади		Вітер		Темпера-тура		Дата
	Кіль-кість	Інтенсив-ність	Швид-кість	Напря-мок	Удень	Уно чі	

1) визначити середню кількість опадів у зазначеній місцевості з 01.06.2017 по 08.10.2017;

2) вивести список місцевостей, у яких у зазначений день температура була $+10^0$ (удень або вночі; указати, коли це було);

3) визначити напрямок вітру в заданій місцевості на 10.09.2017.

18. Об'єкт досліджування, БД якого дозволяє отримати звітну відомість і реалізувати запити:

Працівник підприємства							
Прізвище	Стаж	Підрозділ	Посадова ставка	Посада			
				Фахівець		Бакалавр	
				Кваліфікація	Відділ	Вид роботи	Професія

1)визначити всіх фахівців з посадовою ставкою більше 1 000 грн.;

2) вивести прізвища бакалаврів зі стажем не менше 3 років за професією «Комп'ютерні науки»;

3) визначити фахівців зі стажем менше 2-х роки.

19. Об'єкт досліджування, БД якого дозволяє отримати звітну відомість і реалізувати запити:

Розвиткові товари								
Дата	Фірма	Ціна	Кількість	Музичні інструменти			Спортивні	
				Струнні	Духові	Клавішні	Інвентар	Одежа

1) визначити сумарну вартість товарів, поставлених зазначеною фірмою;

2) вивести список фірм, що поставляють товари за мінімальною ціною;

3) визначити найбільш уживаний музичний інструмент.

20. Об'єкт досліджування, БД якого дозволяє отримати звітну відомість і реалізувати запити:

Спортивні споруди								
Місто	Стадіон				Басейн		Спортзал	
	Тип	Доріжка	Змагання	Місткість	Розмір	Види змагань	Тип	Види змагань

1. вивести список змагань у зазначеному місті на стадіонах із трав'яним покриттям і місткістю до 30 000;

2) визначити список міст із басейнами на 25 м, у яких заплановані змагання із плавання батерфляєм;

3) визначити кількість стадіонів у кожному місті.

21. Об'єкт досліджування, БД якого дозволяє отримати звітну відомість і реалізувати запити:

Освітній процес (суб'єкти)						
ППП	Викладач			Студент		
	Учена ступінь	Кількість наукових праць	Заробітна плата	Стипендія	Середній бал	Форма навчання

1) визначити кількість викладачів із зарплатою менше 4000 грн.;

2) визначити студентів з денною формою навчання;

3) визначити студентів із середнім балом менше 3.3;

4) в якому журналі була надрукована перша стаття викладача, який має найбільшу кількість наукових праць.

22. Об'єкт досліджування, БД якого дозволяє отримати звітну відомість і реалізувати запити:

Офісні товари								
Стілець м'який			Стілець твердий		Стілець на колесах			Ціна
Основа	Оббивка	Кольори	Матеріал	Кольори	Основа	Оббивка	Кількість коліс	

1)вивести вартість всіх стільців;

2) вивести список м'яких стільців з оббивкою червоних кольорів і ціною від 1 до 250 грн;

3) визначити найменш затребувані стільці.

23. Об'єкт досліджування, БД якого дозволяє отримати звітну відомість і реалізувати запити:

Художні твори							
Проза				Поезія		Картини	
Автор	Назва	Жанр	Обсяг	Жанр	Віршований розмір	Напрям	Жанр

1. визначити авторів, які працювали в жанрах прози, поезії і картин;
2. визначити кількість образотворчих здобутків у жанрі портрету уведеного автора;
3. визначити авторів, що писали лише оповідання.

Навчальне видання

СЕРІЯ «НА ДОПОМОГУ СТУДЕНТУ УДФСУ»

Постіл Степан Дмитрович

CASE-ТЕХНОЛОГІЇ. МІЖДИСЦИПЛІНАРНЕ ІНФОРМАЦІЙНЕ МОДЕЛЮВАННЯ

Навчальний посібник

Редактор *Л. Б. Дьомена*

Форматування та
комп'ютерна верстка *Н. М. Шамардак*

Здано до друку 28.03.2018. Формат 60×84/16.
Папір офсетний № 1. Гарнітура «Times New Roman».
Друк. арк. 17.5.
Наклад 300 прим. Замовлення № 693.

Підготовлено до друку Видавничо-поліграфічним центром
Університету ДФС України
08200, вул. Університетська, 31, м. Ірпінь, Київська обл., Україна

Свідоцтво про внесення суб'єкта видавничої справи
до державного реєстру видавців, виготовлювачів і
розповсюджувачів видавничої продукції
Серія ДК № 5104 від 20.05.2016