Географічні
Інформаційні
Системи
в аграрних
університетах
(ГІСАУ)

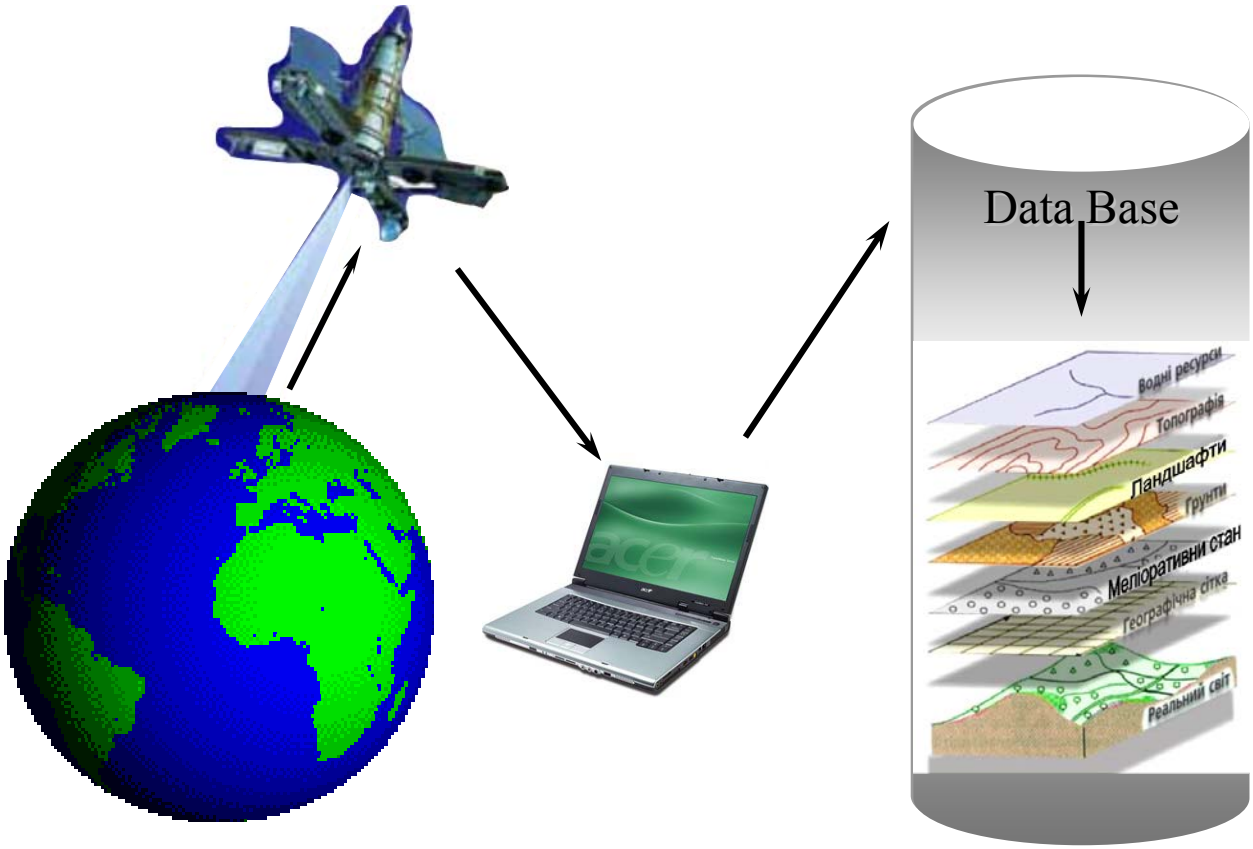
Education and Culture
TEMPUS

Geographic
Information
Systems in
Agrarian
Universities
(GISAU)

Tempus Project

Д.О. Ладичук, В.І. Пічура

Бази геоінформаційних даних



Херсон – 2007

Географічні Інформаційні Системи в аграрних університетах (ГІСАУ)

Geographic Information Systems in Agrarian Universities (GISAU)

Education and Culture
TEMPUS

Tempus Project

Д.О. Ладичук, В.І. Пічура

Бази геоінформаційних даних

Навчальний посібник
За редакцією професора В.В. Морозова

Партнери:

- Херсонський державний аграрний університет (Україна)
- Glasgow Caledonian University (Великобританія)
- University of Gavle (Sweden)
- Херсонський державний університет (Україна)

Проект фінансується за підтримки Європейської Комісії.

Херсон – 2007

Географічні Інформаційні Системи в аграрних університетах (ГІСАУ)

Geographic Information Systems in Agrarian Universities (GISAU)

Tempus Project

D.O. Ladychuk, V.I. Pichura

Geoinformation Databases

Textbook

Edited by professor V.V. Morozov

Partners:

- Kherson State Agrarian University (UKR)
- Glasgow Caledonian University (United Kingdom)
- University of Gavle (Sweden)
- Kherson State University (UKR)

This project has been funded with support from the European Commission.

This publication reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Kherson – 2007

УДК 0046

ББК 32.97

Рецензенти:

ЛИСОГОРОВ К.С. – доктор сільськогосподарських наук, ст. науковий співробітник, завідуючий лабораторією автоматизованих систем управління Інституту землеробства південного регіону УААН;
МАРАСАНОВ В.В. – доктор технічних наук, професор, завідуючий кафедрою економічної кібернетики Херсонського державного аграрного університету.

Ладичук Д.О., Пічура В.І. Бази геоінформаційних даних / За ред. професора В.В. Морозова – Херсон: Вид-во ХДУ, 2007. - 103 с.

Навчальний посібник призначений для фахівців і студентів, які займаються створенням бази даних для геоінформаційних систем та систем управління базами даних і є складовою частиною дисциплін спеціалізації "Геоінформаційні системи і технології в управлінні водними і земельними ресурсами"

Ладычук Д.А., Пичура В.И. Базы геоинформационных данных / Под ред. профессора В.В. Морозова – Херсон: Из-во ХГУ, 2007. - 103с.

Учебное пособие предназначено для специалистов и студентов, которые занимаются созданием базы данных для геоинформационных систем и систем управления базами данных и являются составной частью дисциплин специализации "Геоинформационные системы и технологии в управлении водными и земельными ресурсами"

Ladychuk D.O., Pichura V. I. Data Geoinformation bases / Edited by professor V.V. Morozov - Kherson: Kherson State University, 2007 – 103 p.

The textbook is intended for specialists and students engaged in creating databases for geoinformation systems and databases control systems. This study area is part of the university course “Geoinformation systems and technologies in water and land management”.

Рекомендовано до друку вченою радою Херсонського державного аграрного університету 24 січня 2007р. (протокол № 5)

ISBN 966

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ

АБД –адміністратор баз даних
АД – адміністратор даних
БД – бази даних
БЗ – база знань
БнД – банки даних
ВК – вторинний ключ
ГІС (GIS) –геоінформаційні системи
ДОБ – достовірна обчислювальна база
ЕОМ – електронно – обчислювальна машина
ЕС – експертна система
ІБ – інформаційна база
ІТ – інформаційна технологія
КП – ключ первинний
ММД – мова маніпулювання даними
МОД – мова опису даних
НСД – несанкціонований доступ
НРБ – нормативно регламентуюча база
ПАМС – природно – агромеліоративна геосистема
ПІК – програмно - інформаційний комплекс
ПК – персональний комп’ютер
ПС – підсистема
ПСППР – просторові системи підтримки прийняття рішень
СППР – система підтримки прийняття рішень
СУБД – система управління базами даних
СУРБД – система управління розподіленою базою даних
ШІ – штучний інтелект
CD-ROM – компакт – диск
DDE (Dynamic Data Exchange) - динамічний обмін даними
DLL – динамічні бібліотеки
ODBC (Open Database Connectivity) – відкрите з'єднання баз даних
OLE (Object Linking and Embedding) - зв'язування й впровадження об'єктів
SQL (Structured Query Language)- мова структурованих запитів

ЗМІСТ

ВСТУП

Завдяки широкому застосуванню ГІС в усіх сферах професійної та громадської діяльності зростає роль географічної інформації як багатогалузевого та загальносуспільного предмета споживання.

Створення інфраструктур геоінформаційних та геопросторових даних будь – якого рівня ґрунтується на загальних основних складових, принципах і методах реалізації. До таких належать: інституційні основи, базові набори геопросторових даних, бази метаданих та механізми обміну даними, стандарти на геопросторові дані, метадані та геоінформаційні сервіси, технологічні засоби інформаційно – комунікаційного середовища створення, оброблення та використання геопросторових даних [1].

Як нова сфера інформаційної діяльності, що швидко розвивається та широко застосовується, геоінформаційна сфера наприкінці ХХ століття досягла свого інформаційного і технічного бар'єра, коли технологічний рівень організації даних, їх накопичення, пошуку та доступу до них, оцінки їх якості та придатності для конкретного використання не відповідає постійно зростаючим обсягам геопросторових даних, що створюються різними суб'єктами геоінформаційної діяльності [2].

Основні ідеї сучасної інформаційної технології базуються на концепції, відповідно до якої дані повинні бути організовані в бази даних з метою адекватного відображення реального миру, що змінюється, і задоволення інформаційних потреб користувачів. Ці бази даних створюються й функціонують під управлінням спеціальних програмних комплексів, що мають назву системи управління базами даних (СУБД). А так як бази даних є основою ГІС, виникає необхідність у створенні цього навчального посібника.

Потреба в базах даних виникає на тому етапі досліджень, коли виділяються групи даних, які повинні зберігатись й об'єм інформації стає достатньо великим.

У першу чергу сюди можна віднести галузь "Гідромеліорацію", де бази даних є основою для оптимізації еколого – меліоративних заходів при управлінні водними і земельними ресурсами.

Вимоги до рівня засвоєння вмісту дисципліни

Магістр повинен:

знати:

- сучасні підходи до створення баз даних ГІС;
- сучасне програмне забезпечення для створення та роботи з базами даних;

вміти:

- створювати бази даних для побудови моделей в середовищі гідромеліоративної науки;
- маніпулювати даними та аналізувати просторові бази даних.

Структура дисципліни

Курс	Кількість студентів	Кількість груп	Назва дисциплін	Назва кафедри	I семестр								
					Кільк. кредитів	Навчальних годин						Форма підсум. контролю	Кільк. КП, КР, РГР (к)
						Всього	в т.ч. аудиторних	із них			Самостійні		
								Лекції	Лабор.	Практ			
1	2	3	4	5	6	7	8	9	10	11	12	13	14
6	20	1	Бази даних ГІС	20	2,0	72	36	12	16	8	36	Е	к

1 ВСТУП ДО БАЗИ ГЕОІНФОРМАЦІЙНИХ ДАНИХ

1.1 Основні поняття ГІС

Геоінформаційна система (ГІС) — інформаційна система, що призначена для обробки просторово-часових даних, основою інтеграції якої слугує географічна інформація. Це засіб або інструментарій для вирішення завдань територіального плану. ГІС являє собою набір підсистем збору, збереження, маніпулювання даними та їхнього аналізу, виведення та представлення просторово-організованої інформації [4,5,6,7].

Геоінформаційна технологія (ГІС-технологія) — комп'ютерна технологія вводу, зберігання, обробки і представлення просторово - координованої інформації; технологічна основа створення географічних й інформаційних систем на базі специфічних методів організації даних, аналізу й оцінювання їхньої просторової мінливості (геостатистики), інтеграції та представлення інформації до систем підтримки прийняття рішень [4, 5]. Визначення «географічна» у назві географічних інформаційних систем і геоінформаційних технологій є, по суті, синонімом просторової інформації.

Експертна система (ЕС) розглядається як система штучного інтелекту. Вона включає базу знань з набором правил, які дають змогу на основі фактів, отриманих користувачем, розпізнати ситуацію, здійснити діагностику та сформувати рішення або рекомендації [4, 6].

Система підтримки прийняття рішень (СППР) являє собою кінцевий продукт реалізації інформаційних технологій. Це — інтерактивна комп'ютерна система, що забезпечує діалог з користувачем й інтегрує моделі функціонування об'єкта для прийняття змістовного рішення. Як організований набір алгоритмів, оптимізаційних та імітаційних моделей, вона у сукупності з базами знань і базами даних забезпечує інтерпретацію та використання результатів вибору того чи іншого сценарію рішень [7]. При поєднанні СППР, побудованих на базі ЕС, з геоінформаційними системами формуються просторові системи підтримки прийняття рішень (ПСППР).

В основу формування будь-яких інформаційних баз покладені моделі організації даних для різних рівнів функціональних завдань.

Ядром будь-якої бази даних є модель даних. Модель даних являє собою безліч структур даних, обмежень цілісності й операцій маніпулювання даними. За допомогою моделі даних можуть бути представлені об'єкти предметної області й взаємозв'язку між ними.

Модель даних — сукупність структур даних і операцій їхньої обробки.

СУБД ґрунтується на використанні ієрархічної, мережної або реляційної моделі, на комбінації цих моделей або на деякій їхній підмножині [1].

Розглянемо три основних типи моделей даних: ієрархічну, мережну й реляційну.

Ієрархічна модель даних. Ієрархічна структура представляє сукупність елементів, зв'язаних між собою за певними правилами. Об'єкти, зв'язані ієрархічними відносинами, утворять орієнтований граф (перевернене дерево).

До основних понять ієрархічної структури відносяться: рівень, елемент (вузол), зв'язок. **Вузол** — це сукупність атрибутів даних, що описують деякий об'єкт. На схемі ієрархічного дерева вузли представляються вершинами графа. Кожний вузол на більш низькому рівні зв'язаний тільки з одним вузлом, що перебуває на більш високому рівні. Ієрархічне дерево має тільки одну вершину (корінь дерева), не підлеглу ніякій іншій вершині й яка знаходиться на самому верхньому (першому) рівні. Залежні (підлеглі) вузли перебувають на другому, третьому й т.д. рівнях. Кількість дерев у базі даних визначається числом кореневих записів.

До кожного запису бази даних існує тільки один (ієрархічний) шлях від кореневого запису.

Мережна модель даних. У мережній структурі при тих же основних поняттях (рівень, вузол, зв'язок) кожний елемент може бути пов'язаний з будь-яким іншим елементом.

Реляційна модель даних. Поняття реляційний (англ. relation — відношення) пов'язане з розробками відомого американського фахівця в області систем баз даних Е. Кодда [13].

Ці моделі характеризуються простотою структури даних, зручним для користувача табличним поданням і можливістю використання формального апарата алгебри відносин і реляційного вирахування для обробки даних.

Реляційна модель орієнтована на організацію даних у вигляді двовимірних таблиць. Кожна реляційна таблиця являє собою двовимірний масив і має наступні властивості:

- кожний елемент таблиці - один елемент даних;
- всі стовпці в таблиці однорідні, тобто всі елементи в стовпці мають однаковий тип (числовий, символьний і т.д.) і довжину;
- кожний стовпець має унікальне ім'я;
- однакові рядки в таблиці відсутні;
- порядок проходження рядків і стовпців може бути довільним.

Відносини представлені у вигляді **таблиць,** рядки яких відповідають кортежам або **записам,** а стовпці — атрибутам відносин, доменам, **полям.**

Поле, кожне значення якого однозначно визначає відповідний запис, називається **простим ключем** (ключовим полем). Якщо записи однозначно визначаються значеннями декількох полів, то така таблиця бази даних має **складний ключ**.

Щоб зв'язати дві реляційні таблиці, необхідно ключ першої таблиці увести до складу ключа другої таблиці (можливий збіг ключів); у противному випадку потрібно ввести в структуру першої таблиці **зовнішній ключ** — ключ другої таблиці.

Розрізняють концептуальний, внутрішній і зовнішній рівні подання даних баз даних, яким відповідають моделі аналогічного призначення,

Концептуальний рівень відповідає логічному аспекту подання даних предметної області в інтегрованому виді. **Концептуальна модель** складається з безлічі екземплярів різних типів даних, структурованих відповідно до вимог СУБД й логічної структури бази даних.

Внутрішній рівень відображає необхідну організацію даних у середовищі зберігання й відповідає фізичному аспекту подання даних. **Внутрішня модель** складається з окремих екземплярів записів, фізично збережених у зовнішніх носіях.

Зовнішній рівень підтримує приватні подання даних, необхідні конкретним користувачам. **Зовнішня модель** є підмножиною концептуальної моделі. Можливе перетинання зовнішніх моделей за даними. Приватна логічна структура даних для окремого додатка (завдання) або користувача відповідає зовнішній моделі або підсхемі БД. За допомогою зовнішніх моделей підтримується санкціонований доступ до даних БД додатків (обмежені склад і структура даних концептуальної моделі БД доступних у додатку, а також задані допустимі режими обробки цих даних: введення, редагування, видалення, пошук).

Поява нових або зміна інформаційних потреб існуючих додатків вимагають визначення для них коректних зовнішніх моделей, при цьому на рівні концептуальної й внутрішньої моделі даних змін не відбувається. Зміни в концептуальній моделі, викликані появою нових видів даних або зміною їх структур, можуть зачіпати не всі додатки, тобто забезпечується певна незалежність програм від даних. Зміни в концептуальній моделі повинні відбиватися й у внутрішній моделі, і при незмінній концептуальній моделі можлива самостійна модифікація внутрішньої моделі БД з метою поліпшення її характеристик (час доступу до даних, витрати пам'яті зовнішніх пристроїв та ін.). Таким чином, БД реалізує принцип відносної незалежності логічної й фізичної організації даних.

Проектування бази даних складається в побудові комплексу взаємозалежних моделей даних.

Найважливішим етапом проектування бази даних є розробка інфологічної (інформаційно-логічної) моделі предметної області, не орієнтованої на СУБД. В інфологічній моделі засобами структур даних в інтегрованому виді відбивають состав і структуру даних, а також інформаційні потреби додатків (завдань і запитів).

Інформаційно-логічна (інфологічна) модель предметної області відбиває предметну область у вигляді сукупності інформаційних об'єктів і їхніх структурних зв'язків.

Інфологічна модель предметної області будується першою. Попередня інфологічна модель будується ще на передпроектній стадії й, потім уточнюється на більш пізніх стадіях проектування баз даних. Потім на її основі будуються концептуальна (логічна), внутрішня (фізична) і зовнішня моделі.

1.2 Базові структури даних в ГІС

Дані - це ймовірно найбільш важливий компонент ГІС. Дані про просторове положення (географічні дані) і пов'язані з ними табличні дані можуть збиратися й підготовлятися самим користувачем, або здобуватися в постачальників на комерційній або іншій основі. У процесі управління просторовими даними ГІС інтегрує просторові дані з іншими типами й джерелами даних, а також може використовувати СУБД, що застосовуються багатьма організаціями для упорядочення й підтримки наявних у їхньому розпорядженні даних.

ГІС зберігає інформацію про реальний світ у вигляді набору тематичних шарів, які об'єднані на основі географічного положення. Цей простий, але дуже гнучкий підхід довів свою цінність при вирішенні різноманітних реальних завдань. Будь-яка географічна інформація містить відомості про просторове положення, будь те прив'язка до географічних або інших координат, або посилання на адресу, поштовий індекс, виборчий округ або округ перепису населення, ідентифікатор земельної або лісової ділянки, назва дороги й т.п. При використанні подібних посилань для автоматичного визначення місця розташування або місць розташування об'єкта (об'єктів) застосовується процедура, що має назву геокодування. З її допомогою можна швидко визначити й подивитися на карті, де перебуває об'єкт, що цікавить вас, або явище, такі як будинок, у якому проживає ваш знайомий або перебуває потрібна вам організація, де відбулося землетрус або повінь, за яким маршрутом простіше й швидше добратися до потрібного вам пункту.

ГІС може працювати із двома типами даних, що істотно відрізняються - векторними й растровими. У векторній моделі інформація про крапки, лінії й полігони кодується й зберігається у вигляді набору координат X,Y. Місце розташування крапки (крапкового об'єкта), наприклад свердловини, описується парою координат (X,Y). Лінійні об'єкти, такі як дороги, ріки або трубопроводи, зберігаються як набори координат X,Y. Полігональні об'єкти, типу річкових водозборів, земельних ділянок або областей обслуговування, зберігаються у вигляді замкнутого набору координат. Векторна модель особливо зручна для опису дискретних об'єктів і менше підходить для опису безупинно мінливих властивостей, таких як типи ґрунтів або доступність об'єктів. Растрова модель оптимальна для роботи з безперервними властивостями. Растрове зображення являє собою набір значень для окремих елементарних складових (осередків), воно подібно відсканованій карті або картинці. Обидві моделі мають свої переваги й недоліки. Сучасні ГІС можуть працювати як з векторними, так і з растровими моделями.

Геоінформаційні дані містять чотири інтегрованих компоненти:

- географічне положення (розміщення) просторових об'єктів представляється 2-х, 3-х або 4-х мірними координатами в географічно співвіднесеній системі координат (широта/довгота);

- атрибути - властивість, якісна або кількісна ознака, що характеризує просторовий об'єкт (але не пов'язана з його місцевказанням);
- просторові відносини визначають внутрішні взаємини між просторовими об'єктами (наприклад, напрямок об'єкта А у відношенні об'єкта В, відстань між об'єктами А и В, вкладеність об'єкта А в об'єкт В);
- тимчасові характеристики представляються у вигляді термінів одержання даних, вони визначають їхній життєвий цикл, зміну місця розташування або властивостей просторових об'єктів у часі.

Основні елементи бази просторових даних:

- елементи дійсності, змодельовані в базі даних ГІС мають дві тотожності: реальний об'єкт і змодельований об'єкт (об'єкт БД);
- реальний об'єкт - явище навколишнього світу, що представляє інтерес, що не може бути більше підрозділене на явища того ж самого типу;
- об'єкт БД - елемент, у тім виді, у якому він представлений у базі даних (об'єкт БД є "цифровим поданням цілого або частини реального об'єкта");
- метод цифрового подання явища змінюється виходячи з базового масштабу й ряду інших факторів.

Модель бази просторових даних:

- кожний тип реального об'єкта представляється певними просторовими об'єктами бази даних;
- просторові об'єкти можуть бути згруповані в шари, відомі як оверлеї, покриття або теми.
- один шар може представляти одиночний тип об'єкта або групу концептуально зв'язаних типів.

Подання просторових даних - спосіб цифрового опису просторових об'єктів. Найбільш універсальні й вживані з них:

- векторне подання (крапки, лінії, полігони):
  - (векторно-топологічне подання;
  - векторно-нетопологічне або модель "спагетті");
- растрове подання (осередку, сітки);
- регулярно-ніздрюватe подання;
- квадродерево (квадротомічне подання).

До менш розповсюджених або застосовуваних для подання просторових об'єктів певного типу відносяться також гіперграфова модель, модель типу TIN та її багатомірні розширення.

Існують способи й технології переходу від одних просторових даних до інших (растрово-векторне перетворення, векторно-растрове перетворення).

Подання просторових об'єктів реальної дійсності.

- Просторові дані складаються із цифрових подань реально існуючих дискретних просторових об'єктів.

- Властивості, показані на карті, наприклад, озера, будинки, контури, повинні розумітися як дискретні об'єкти.
- Вміст карти може бути зафіксований в базі даних, шляхом перетворення властивостей карти в просторові об'єкти.
- Багато властивостей, які показані на карті, насправді є віртуальні. Наприклад, контури або границі реально не існують, але будинки й озера - реальні об'єкти.

Вміст бази просторових даних включає:

- цифрові версії реально існуючих об'єктів (наприклад, будинків);
- цифрові версії штучно виділених властивостей карти (наприклад, контури);
- штучні об'єкти, створені спеціально для цілей побудови бази даних (наприклад, пікселі).

Різновид безперервних властивостей - деякі властивості просторових об'єктів існують повсюдно й змінюються безупинно над земною поверхнею (висота, температура, атмосферний тиск) і не мають реально представлених границь.

Безперервна мінливість може бути представлена в базі даних декількома способами:

- за допомогою величин вимірів у деяких характерних пунктах (крапках), (наприклад, метеостанції й пости);
- за допомогою описів трансектів (профілів - вертикальних розрізів ділянок земної поверхні);
- за допомогою поділу площі на контури, зони, приймаючи при цьому, що деяке значення властивості усередині контуру (зони) є величина постійна;
- за допомогою побудови ізоліній, наприклад горизонталей для відображення рельєфу.

Кожний з цих способів створює дискретні об'єкти, які можуть бути зафіксовані у вигляді крапок, ліній, площ.

Компоненти просторових даних:

- розташування: просторові дані взагалі часто називаються даними про розміщення;
- просторові відносини: взаємозв'язки між просторовими об'єктами описуються як просторові відносини між ними (наприклад, А містить В; суміжний з В, перебуває до півночі від D);
- атрибути: атрибути фіксують тематичні описи, визначаючи різні характеристики об'єктів;
- час: тимчасова мінливість фіксується різними способами:
    - - інтервалом часу, протягом якого існує об'єкт;
    - - швидкістю мінливості об'єктів;
    - - часом одержання значень властивостей [9].

1.3 Особливості експертних систем для обробки даних

Структура експертної системи визначається наступними модулями:

1) тимчасові бази даних, призначені для зберігання вихідних і проміжних даних поточного завдання;

2) бази знань, призначені для зберігання довгострокових відомостей /фактів/ і правил маніпулювання даними;

3) вирішатель /база програм/, що реалізує послідовність правил для рішення конкретного завдання на основі інформації, що зберігається в базах знань і базах даних;

4) компонент придбання знань, що автоматизує процес наповнення бази знань;

5) пояснювальний компонент, що формує пояснення про те, як система вирішувала поставлене завдання.

Експертні системи для обробки даних називають **статистичними експертними системами**. Такі системи за рахунок спільного користувальницького інтерфейсу повинні мати можливість допомогти починаючому користувачеві не тільки ввести результати спостережень, але й уточнити завдання обробки й, при необхідності, спланувати експеримент, що дозволяє вирішити поставлене завдання. У базі знань експертної системи повинна зберігатися досить велика й постійно поповнювана кількість відомостей та правил, для того, щоб забезпечити можливість рішення різноманітних завдань обробки даних. Пояснення про те як система вирішувала поставлене завдання повинні бути зрозумілі фахівцеві в предметній області й, у той самий час, містити досить інформації для аналізу вірогідності результатів обробки фахівцем з математичної статистики. Розробка експертних систем, призначених для обробки даних, натрапляє на величезні труднощі. «Інтелектуалізація» комп'ютерної обробки первинної інформації про навколишнє середовище ґрунтується, з одного боку, на ідеях і методах конкретної області знання, для якої створюється система обробки даних. З іншого боку, у комп'ютерній системі обробки використовуються різноманітні методи прикладної математики - математичної статистики, теорії рішення зворотних завдань і т.п. Відповідно, при створенні експертних систем обробки даних, з одного боку, доводиться враховувати методичні й метрологічні особливості методик виконання виміру, а з іншого боку - апріорні припущення й обмеження математичних алгоритмів обробки, що допускає участь досить великого колективу професіоналів - фахівців у предметній області, математиків, програмістів і фахівців з розробки експертних систем і високу вартість розробки. Тому при наявності величезного числа систем загального призначення - пакетів для статистичної обробки даних, електронних таблиць і т.п., існує невелике число експертних систем, здатних автоматично провести весь цикл аналізу даних [3].

Геоінформаційні системи забезпечують створення нових програмних модулів, які розширюють можливості ПІК та урізноманітнюють ступінь

деталізації інформації відповідно до завдань, що вирішуються, та рівнів їхньої ієрархії. Особливістю геоінформаційних систем є наявність у їхньому складі специфічних методів аналізу просторової мінливості даних, які у сукупності із засобами вводу, зберігання, маніпулювання та представлення просторово - координованої інформації (картографічної, атрибутивної) становлять основу ГІС-технологій. ГІС включає чотири основні підсистеми:

- збору даних, що вибирає, формує та здійснює попередню обробку інформації з різних джерел; ця підсистема також відповідає за перетворення різних типів просторової інформації (її збір та складання карт);
- збереження та вибірки даних, що забезпечує організацію просторової інформації з метою її компонування, оновлення та редагування відповідно до конкретних запитів користувача;
- маніпулювання даними, їхніх аналізу та оцінки, що вирішує алгоритми різних завдань на основі цієї інформації, згруповує та розділяє її, установлює параметри та обмеження, виконує функції моделювання;
- виведення, що відображає всю інформаційну базу даних, або частину її в табличній, графічній (діаграмній), картографічній формах залежно від функціональних завдань ГІС.

Інформацію до бази подають у вигляді тематично (або предметно) та системно організованих даних.

Організація тематичних відомостей передбачає формування наборів галузевих даних, що характеризують, насамперед, природне середовище як природно-ресурсну основу об'єктів і виробничо-технологічну структуру території з виділенням базових та функціональних модулів.

У базових модулях концентрується інформація, що характеризує параметри природного середовища та господарської діяльності. Основним базовим модулем природно-ресурсної основи є **географічна модель території,** яка описує основні компоненти геологічного середовища, природні чинники зовнішньої дії на нього, динаміку їхнього розвитку у просторі — часі.

Системна диференціація тематичних даних реалізується організацією спеціалізованих модулів **бази знань** (БЗ) і **бази даних** (БД), які становлять основу інформаційної бази ПІК об'єкту. Це, насамперед, блоки інформаційно-довідкової та нормативно-регламентуючої систем, локальної інформаційно-довідкової бази об'єкта досліджень, спеціалізованого тематичного картографічного фонду, банків даних оперативної, довгострокової та результуючої інформації.

За цільовим призначенням СППР ПІК об'єкту інформація в рамках бази знань і бази даних має низку певних особливостей і формується за суто адресними вимогами структурної організації даних в експертних та геоінформаційних системах.

Наповнення **бази знань** відбувається на основі узагальнення та систематизації досвіду досліджень з оцінювання природно-агромеліоративних об'єктів, організації контролю за їхнім станом, водогосподарських умов та ін.

База даних має забезпечити систему підтримки рішень базовою, довгостроковою, оперативною, а також результуючою інформацією для вибору сценаріїв та рекомендацій. Бази даних ГІС акумулюють інформацію у вигляді, відповідним чином закодованих шарів однорідних картографічних даних і просторово - прив'язаної до конкретної території або точки спостереження атрибутивної інформації.

Накопичення матеріалів у базах даних здійснюється покомпонентно відповідно до програм моніторингу, спеціальних обстежень та випробувань, що підпорядковані різним організаціям з наступною адаптацією інформації до завдань ПІК об'єкту.

База даних формується на рівні об'єкта досліджень (регіональний, локальний чи детальний) з урахуванням певних вимог до наповнення залежно від показників, що фіксуються, та методів їхнього одержання.

До складу БД входить локальна інформаційно-довідкова база об'єкта досліджень (регіональний та локальний рівень узагальнення інформації), спеціалізований тематичний картографічний фонд (банки даних картографічної інформації) та банк відповідних атрибутів до карт, банки даних точкової інформації — оперативної, довгострокової і результуючої.

Наповнення бази знань та бази даних відбувається у декілька етапів. На першому етапі виконується упорядкування наявної, одержаної з різних джерел, інформації щодо території, у часі, за конкретним змістом, тематикою тощо. Другий етап включає узгодження упорядкованої інформації та її оптимізацію. На третьому етапі інформація інтегрується — вирішуються завдання більш високих рівнів складності та комплексності, тобто завдання з вибору і прийняття рішень.

Інформаційні бази мають залишатися відкритими, здатними до трансформації, забезпечуючи легкість модифікації системи, як в еволюційному плані, так і для вирішення нових завдань, супроводжуватись гнучкими СУБД, системою SQL-запитань та розвинутим інтерфейсом.

СУБД являє собою сукупність програмних і технічних засобів, що забезпечують функціонування геоінформаційних та експертних систем, а саме — введення інформації, її накопичення й оновлення, засобів збереження, пошуку та перетворення інформації, видачі матеріалів користувачу.

Відношення між окремими підсистемами, модулями та блоками інформаційної бази встановлюють формуванням файлів зв'язку (система SQL-запитань). При цьому стійкі відношення між об'єктами у підсистемах фіксуються сукупністю ієрархічних класифікаційних графів природних, природно-господарських об'єктів, адміністративно-територіальних та природно-територіальних одиниць, а підсистеми предметних даних

вміщують файли, що описують у вигляді таблиць-відношень самі об'єкти та їхній стан.

Вимоги до інформаційної бази ПІК об'єкту можуть змінюватись, розширюватись або звужуватись залежно від конкретного призначення ГІС та ЕС, складу завдань, що вирішуються, та особливостей об'єкта досліджень [8].

1.4. Поняття бази даних

Мета будь-якої інформаційної системи — обробка даних про об'єкти реального миру. У широкому змісті слова база даних — це сукупність відомостей про конкретні об'єкти реального миру в якій-небудь предметній області. Під **предметною областю** прийнято розуміти частину реального миру, що підлягає вивченню для організації управління й в остаточному підсумку автоматизації, наприклад, підприємство, вуз і т. п.

Створюючи базу даних, користувач прагне впорядкувати інформацію з різними ознаками і швидко робити вибірку з довільним сполученням ознак. Зробити це можливо, тільки якщо дані структуровані.

Структурування — це введення угод про способи подання даних.

Неструктурованими називають дані, записані, наприклад, у текстовому файлі.

Поле, що зберігається, — це якнайменша одиниця даних, що зберігаються. Взагалі, база даних містить багато екземплярів кожного з декількох типів полів, що зберігаються. Наприклад, база даних, що містить інформацію про ландшафти, може містити тип полів з ім'ям, що зберігаються, "номер ландшафту", і для кожного виду ландшафту (степові, лісостепові і т.д.) передбачений один екземпляр цього поля, що зберігається. **Запис, що зберігається,** — це набір зв'язаних полів, що зберігаються. Тут знову розрізняють тип і екземпляр. В даному випадку екземпляр запису, що зберігається, складається з групи зв'язаних екземплярів полів, що зберігаються. Наприклад, екземпляр запису, що зберігається, в базі даних ландшафтів складається з екземплярів кожного з полів, що зберігаються: номер ландшафту, назва ландшафту, місцеположення ландшафту і вид меліорації даного ландшафту. База даних містить безліч екземплярів запису типу "ландшафт", що зберігається (один екземпляр для кожної певного ландшафту).

У файлі, що зберігається, може бути декілька типів записів, що зберігаються. У системах, відмінних від баз даних, звичайно логічний запис співпадає з відповідним записом, що зберігається. У базах даних це зовсім не обов'язково, оскільки АБД може знадобитися внести зміни в структуру зберігання даних (тобто в поля, що зберігаються, записи, файли), тоді як логічна структура залишиться незмінною.

Користувачами бази даних можуть бути різні прикладні програми, програмні комплекси, а також фахівці предметної області, що виступають у

ролі споживачів або джерел даних. Вони називаються **кінцевими користувачами**.

У сучасній технології баз даних передбачається, що створення бази даних, її підтримка й забезпечення доступу користувачів до неї здійснюються централізовано за допомогою спеціального програмного інструментарію — системи управління базами даних.

База даних (БД) — це пойменована сукупність структурованих даних, що відносяться до певної предметної області.

Система управління базами даних (СУБД) — це комплекс програмних і мовних засобів, необхідних для створення баз даних, підтримки їх в актуальному стані й організації пошуку в них необхідної інформації.

Централізований характер управління даними в базі даних допускає необхідність існування деякої особи (групи осіб), на яку покладають функції адміністрування даними, збереженими в базі.

За технологією обробки дані бази даних підрозділяються на централізовані й розподілені.

Централізована база даних зберігається в пам'яті однієї обчислювальної системи. Якщо ця обчислювальна система є компонентом мережі ЕОМ, можливий розподілений доступ до такої бази. Такий спосіб використання баз даних часто застосовують у локальних мережах ПК.

Розподілена база даних складається з декількох, можливо пересічних або навіть дублюючих одну одну частин, збережених у різних ЕОМ обчислювальній мережі. Робота з такою базою здійснюється за допомогою системи управління розподіленою базою даних (СУРБД).

За способом доступу до даних, бази даних розділяються на бази даних з локальним доступом і бази даних з вилученим (мережним) доступом.

Система баз даних (database system) — це, по суті, не що інше, як комп'ютеризована система зберігання записів, основна мета якої – зберігати інформацію й надавати її на вимогу. До інформації може відноситися все, що заслуговує на увагу окремого користувача або підприємства, що використовує систему. Користувачеві цієї системи надається можливість виконувати безліч різних операцій над такими файлами, наприклад:

- додавати нові порожні файли в базу даних;
- додавати нові дані в існуючі файли;
- вести пошук даних в існуючих файлах;
- змінювати дані в існуючих файлах;
- видаляти дані з існуючих файлів;
- видаляти існуючі файли з бази даних, тобто позбуватися від їхнього вмісту [41].

Однокористувальницька система (single-user system) — це система, у якій у той самий час до бази даних може одержати доступ не більше одного користувача; багатокористувальницька система (multi-user system) — це система, у якій до бази даних можуть одержати доступ відразу кілька користувачів.

У загальному випадку дані в базі даних (принаймні у великих системах) є інтегрованими й загальними. Ці два аспекти, інтеграція й дозвіл загального доступу, являють собою найбільш важливу перевагу використання систем баз даних на "великому" устаткуванні; і щонайменше один з них - інтеграція - є перевагою їхнього використання на "малому" устаткуванні.

Під поняттям інтегровані дані мається на увазі можливість представити базу даних як об'єднання декількох окремих файлів даних, які повністю або частково перекриваються.

Під поняттям загальні дані мається на увазі можливість використання окремих областей даних у базі даних декількома різними користувачами, тобто кожний з цих користувачів може мати доступ до однієї й тієї ж області даних (причому різні користувачі можуть використовувати ці дані для різних цілей). Як вже згадувалося, різні користувачі можуть мати доступ навіть до однієї й тієї ж області даних у той самий час (одночасний доступ) [10].

Вхідні дані — це інформація, передана системі (звичайно з термінала або робочої станції). Така інформація може стати причиною змін у постійних даних (вона може стати частиною постійних даних), але не є частиною бази даних, як такої.

Вихідні дані — це повідомлення й результати, видані системою (звичайно видаються на печатку або відображаються на екрані). І знову ж цю інформацію можна брати з постійних даних, але її не можна розглядати як частину бази даних.

Розходження між постійними й транзитними даними не можна назвати чітким - воно в деякій мері залежить від контексту (наприклад, від того, як використовуються дані). Однак допускаючи, що це розходження доступно принаймні на інтуїтивному рівні, можна дати більше точне визначення терміна "база даних":

Переваги системи баз даних у порівнянні з традиційним паперовим методом вмісту записів наступні.

• Компактність. Немає необхідності в багатотомних паперових картотеках.

• Швидкість. Комп'ютер може вести пошук і змінювати дані набагато швидше людини. Зокрема, на спеціальні питання, що виникають у процесі роботи (наприклад: "Яких труб в нас зараз більше – ВТ-12 або ВТ-15?"), можна одержати відповідь швидко, не затрачаючи часу на візуальний пошук.

• Низькі трудовитрати. Немає необхідності в стомлюючій ручній роботі над картотекою. Механічну роботу машини завжди виконують краще.

• Застосовність. Точна, свіжа інформація в будь-який момент під рукою.

Ці переваги здобувають ще більше значення в багатокористувальницькому середовищі, де база даних, імовірно, більше й складніше однокористувальницької. Крім того, багатокористувальницьке середовище має додаткову перевагу: система баз даних надає об'єкту

централізоване управління його даними (а таке управління є найціннішою властивістю бази даних). Уявіть собі протилежну ситуацію – об'єкт, що не використовує систему баз даних: для кожного окремого додатка створюються свої файли, найчастіше розташовані на окремих магнітних стрічках або дисках, у результаті чого дані виявляються розрізненими. Систематично управляти такими даними дуже складно [10, 41].

1.5 Переваги централізованого підходу в управлінні даними

Переваги централізованого підходу в управлінні даними наступні.

• Можливість скорочення надмірності.

У системах, що не використовують бази даних, кожний додаток має свої файли. Це часто призводить до надмірності збережених даних, а отже, до марнотратства пам'яті. Наприклад, як додаток, пов'язаний з викладачами, так і додаток, пов'язаний з обліком підвищення кваліфікації викладачів, можуть мати власний файл з відомчою інформацією про викладачів. Як відзначено вище, ці два файли можна об'єднати з усуненням надмірності (однакової інформації) за умови, якщо адміністратор даних знає про те, які дані потрібні для кожного додатка, тобто якщо на об'єкті здійснюється необхідне загальне управління.

У цьому випадку не мається на увазі, що надмірність даних може або повинна бути повністю усунута. Іноді вагомі практичні або технічні причини вимагають наявності декількох копій збережених даних. Однак така надмірність повинна строго контролюватися, тобто враховуватися в СУБД; також повинна бути передбачена можливість "множинного відновлення".

• Можливість усунення (певною мірою) суперечливості.

У дійсності цей наслідок попереднього пункту. Візьмемо приклад з життя — нехай викладач ЕЗ, що працює на кафедрі ГІС-технологій представлений двома різними записами в базі даних. Допустимо, що в СУБД не враховане це роздвоєння (тобто надмірність не контролюється). Тоді рано або пізно обов'язково виникне ситуація, при якій ці два записи перестануть погоджуватися, а саме: одна з них буде змінена, а друга - ні. У цьому випадку база даних стане суперечлива. Ясно, що суперечлива база даних може видавати користувачеві невірну, суперечливу інформацію.

Також очевидно, що якщо який-небудь факт представлений одним записом (тобто при відсутності надмірності), то протиріч виникнути не може. Протиріччя можна також уникнути, якщо не видаляти надмірність, а контролювати її (передбачивши це відповідним чином у СУБД), тоді, з погляду користувача, база даних ніколи не буде суперечливою. Це забезпечується тим, що якщо відновлення вноситься в один запис, то воно автоматично повинне поширюватися на всі інші. Цей процес називається множинним відновленням, де під відновленням розуміються будь-які операції вставки, видалення або відновлення [41].

• Можливість загального доступу до даних.

Загальний доступ до даних означає можливість доступу до них декількох існуючих додатків бази даних і розробки нових додатків для роботи з цими ж даними. Інакше кажучи, нові додатки можуть одержати доступ до тих же даних, і при цьому немає необхідності в створенні нових даних.

- Можливість дотримання стандартів.

Завдяки централізованому управлінню АБД (під управлінням адміністратора даних) може забезпечувати подання даних у певних стандартах. Стандарти можуть бути корпоративними, настановними, відомчими, промисловими, національними й інтернаціональними. Стандартизація подання даних найбільш важлива для обміну й перенесення даних між системами. Крім того, стандарти ідентифікації даних і документації важливі й для спільного використання даних, і для їхнього розуміння.

- Можливість введення обмежень для забезпечення безпеки.

Завдяки повному контролю над базою даних АБД може забезпечити доступ до бази даних через певні канали, а отже, може визначити правила безпеки, які будуть перевірятися при спробі доступу до уразливих даних (знову ж під керівництвом адміністратора даних). Для різних типів доступу (вибірки, вставки, видалення й т.д.) і різних частин бази даних можна встановити різні правила.

- Можливість забезпечення цілісності даних.

Завдання цілісності полягає в забезпеченні правильності й точності даних у базі даних. Протиріччя між двома записами, що представляють один "факт", являє приклад недоліку цілісності; звичайно, ця проблема може виникнути лише при наявності надмірності в збережених даних. Але навіть якщо надмірність відсутня, база даних може містити неправильну інформацію. Централізоване управління базою даних дозволяє уникнути подібних проблем - наскільки їх взагалі можливо уникнути. Для цього адміністратор даних визначає (а АБД реалізує) правила цілісності, які застосовуються при будь-якій спробі проробити яку-небудь операцію відновлення.

Відзначимо також, що інтеграція даних для багатокористувальницьких систем баз даних навіть більше важлива, чим для "приватних файлів", саме тому, що до такої бази даних є загальний доступ. При відсутності належного контролю один користувач може некоректно обновити дані, і від цього постраждають інші, ні в чому не винні користувачі. Варто також сказати, що в більшості продуктів баз даних підтримка контролю цілісності розвинена слабко, хоча в цей час у цьому напрямку спостерігаються деякі поліпшення.

• Можливість збалансувати суперечливі вимоги.

Знаючи загальні вимоги всього об'єкту (а не вимоги кожного окремого користувача), АБД (звичайно, під управлінням адміністратора даних) може структуризувати базу даних таким чином, щоб обслуговування в цілому для підприємства було найкращим.

Поняття бази даних тісно пов'язане з такими поняттями структурних елементів, як поле, запис, файл (таблиця).

Поле — елементарна одиниця логічної організації даних, що відповідає неподільній одиниці інформації — реквізиту. Для опису поля використовуються наступні характеристики:

ім'я, наприклад, Прізвище, Ім'я, По - батькові, дата народження;

тип, наприклад, символьний, числовий, календарний;

довжина, наприклад, 12 байт, причому буде визначатися максимально можливою кількістю символів;

точність для числових даних, наприклад два десяткових знаки для відображення дробової частини числа.

Запис — сукупність логічно зв'язаних полів. Екземпляр запису - окрема реалізація запису, що містить конкретні значення її полів.

Файл (таблиця) — сукупність екземплярів записів однієї структури. У структурі запису файлу вказуються поля, значення яких є ключами первинними (ПК), які ідентифікують екземпляр запису, і вторинними (ВК), які виконують роль пошукових або групованих ознак (за значенням вторинного ключа можна знайти кілька записів) [10].

Питання для самоперевірки

1. Предметна область і бази даних.
2. Роль баз даних в процесі формування інформаційних ресурсів, в тому числі для цілей управління територіями.
3. Класифікація інформаційних систем.
4. Використання ГІС в гідромеліорації.
5. Експертні системи для обробки даних.
6. Класифікації баз даних.
7. Централізована база даних.
8. Мета інформаційної системи.
9. Переваги централізованого підходу в управлінні даними.
10. Класифікація даних.
11. Основна різниця між базами знань та базами даних.
12. Основні підсистеми ГІС.

2. МОДЕЛІ ДАНИХ

2.1 Види моделей даних

На великих ЕОМ сімдесятих років програмісти прагнули з максимальною ефективністю використовувати машинний час і пам'ять. Тому широке поширення одержав ієрархічний підхід до організації баз даних. **Ієрархічні бази даних** організовуються у вигляді дерев, що припускає нерівноправність між даними – одні дані виявляються жорстко підлеглі іншим. Така організація даних нагадує деякі схеми побудови баз знань в експертних системах і забезпечує високоефективний пошук інформації. Але ієрархічний підхід до організації баз даних має й очевидні недоліки, наприклад, необхідність жорстко визначати зв'язок між даними, що істотно ускладнює організацію інформації. Щоб перебороти подібні недоліки була запропонована **мережна модель даних**, у якій крім вертикальних зв'язків між даними, передбачалися й горизонтальні. Прикладом реалізації такої моделі може служити система директорій (фолдерів), що дозволяє організувати інформацію на жорсткому диску персонального комп'ютера. Але й ця модель вимагає досить значних зусиль при організації інформації.

Дані в базі даних називають "постійними" (хоча насправді вони можуть недовго залишатися такими!). Під словом "постійні" маються на увазі дані, які відрізняються від інших, більше мінливих даних, таких як проміжні результати, вхідні й вихідні дані, оператори управління робочої черги, програмні блоки управління й взагалі всі транзитні дані.

Поєднуючи частки подання про вміст бази даних, отримані в результаті опитування користувачів, і свої подання про дані, які можуть знадобитися в майбутніх додатках, АБД спочатку створює узагальнений неформальний опис створюваної бази даних. Це опис, виконаний з використанням природної мови, математичних формул, таблиць, графіків і інших засобів, зрозумілих всім людям, що працюють над проектуванням бази даних, називають інфологічною моделлю даних.

Така людино-орієнтована модель повністю незалежна від фізичних параметрів середовища зберігання даних. Зрештою цим середовищем може бути пам'ять людини, а не ЕОМ. Тому інфологічна модель не повинна змінюватися доти, поки якісь зміни в реальному світі не зажадають зміни в ній деякого визначення, щоб ця модель продовжувала відбивати предметну область [37].

Інші моделі є комп'ютеро-орієнтованими. З їхньою допомогою СУБД дає можливість програмам і користувачам здійснювати доступ до збережених даних лише за їхними іменами, не піклуючись про фізичне розташування цих даних. Потрібні дані відшукуються СУБД на зовнішніх запам'ятовувальних пристроях за фізичною моделлю даних. Тому, що зазначений доступ здійснюється за допомогою конкретної СУБД, то моделі повинні бути описані мовою опису даних цієї СУБД. Такий опис,

створюваний АБД за інфологічною моделлю даних, називають даталогічною моделлю даних.

Трьохрівнева архітектура (інфологічний, даталогічний і фізичний рівні) дозволяє забезпечити незалежність збережених даних від їхніх програм, що використовують. АБД може при необхідності переписати збережені дані на інші носії інформації й (або) реорганізувати їхню фізичну структуру, змінивши лише фізичну модель даних. АБД може підключити до системи будь-яке число нових користувачів (нових додатків), доповнивши, якщо треба, даталогічну модель. Зазначені зміни фізичної й даталогіної моделей не будуть замічені існуючими користувачами системи (виявляться "прозорими" для них), так само як не будуть замічені й нові користувачі. Отже, незалежність даних забезпечує можливість розвитку системи баз даних без руйнування існуючих додатків.

Інфологічна модель відображає реальний мир у деякій зрозумілій людині концепції, повністю незалежної від параметрів середовища зберігання даних. Існує безліч підходів до побудови таких моделей: графові моделі, семантичні мережі, модель "сутність-зв'язок" і т.д. Найбільш популярною з них виявилася модель "сутність-зв'язок".

Інфологічна модель повинна бути відображена в комп'ютеро-орієнтовану даталогічну модель, "зрозумілу" СУБД. У процесі розвитку теорії й практичного використання баз даних, а також засобів обчислювальної техніки створювалися СУБД, що підтримують різні даталогічні моделі.

Спочатку стали використовувати ієрархічні даталогічні моделі. Простота організації, наявність заздалегідь заданих зв'язків між сутностями, подібність з фізичними моделями даних дозволяли домагатися прийнятної продуктивності ієрархічних СУБД на повільних ЕОМ з досить обмеженими обсягами пам'яті. Але, якщо дані не мали деревоподібної структури, то виникала маса труднощів при побудові ієрархічної моделі й бажанні домогтися потрібної продуктивності.

Мережні моделі також створювалися для малоресурсних ЕОМ. Це досить складні структури, що складаються з "наборів" - пойменованих дворівневих дерев. "Набори" з'єднуються за допомогою "записів - зв'язувань", створюючи ланцюжки й т.д. При розробці мережних моделей було вигадане безліч "маленьких хитрощів", що дозволяють збільшити продуктивність СУБД, але істотно ускладнили останні. Прикладний програміст повинен знати масу термінів, вивчити кілька внутрішніх мов СУБД, детально представляти логічну структуру бази даних для здійснення навігації серед різних екземплярів, наборів, записів і т.п. Один з розроблювачів операційної системи UNIX сказав "Мережна база - це самий вірний спосіб втратити дані" [37].

Складність практичного використання ієрархічних і мережних СУБД змушувала шукати інші способи подання даних. Наприкінці 60-х років з'явилися СУБД на основі інвертованих файлів, що відрізняються простотою організації й наявністю досить зручних мов маніпулювання

даними. Однак такі СУБД володіють рядом обмежень на кількість файлів для зберігання даних, кількість зв'язків між ними, довжину запису й кількість її полів.

Фізична організація даних впливає на експлуатаційні характеристики БД. Розроблювачі СУБД намагаються створити найбільш продуктивні фізичні моделі даних, пропонуючи користувачам той або інший інструментарій для налагодження моделі під конкретну БД. Розмаїтість способів корегування фізичних моделей сучасних промислових СУБД не дозволяє розглянути їх у цьому розділі.

2.2 Структури баз даних для управління даними

Організований набір взаємозалежних файлів даних називається базою даних. Складність роботи з множинними файлами в базі даних вимагає більш конкретного управління, реалізованого системою управління базою даних (СУБД). Хоча постійно створюються все нові реалізації структур баз даних, існує всього три основних типи: ієрархічна (деревоподібна), мережна і реляційна (таблична) структури баз даних.

Ієрархічна структура даних. У багатьох випадках існує взаємозв'язок між даними, що має назву відношенням "один до багатьох" (рис.2.1) [P.A. Burrough, 1983]. Це відношення має на увазі, що кожен елемент даних має прямий зв'язок з деяким числом так званих "нащадків", і, звичайно кожен такий нащадок, у свою чергу, може мати зв'язок зі своїми нащадками і т.д. Як випливає з назви, предки і нащадки прямо зв'язані між собою, що робить доступ до даних простим і ефективним. Така система добре ілюструється ієрархічною системою класифікації рослин та тварин, що має назву таксономія.

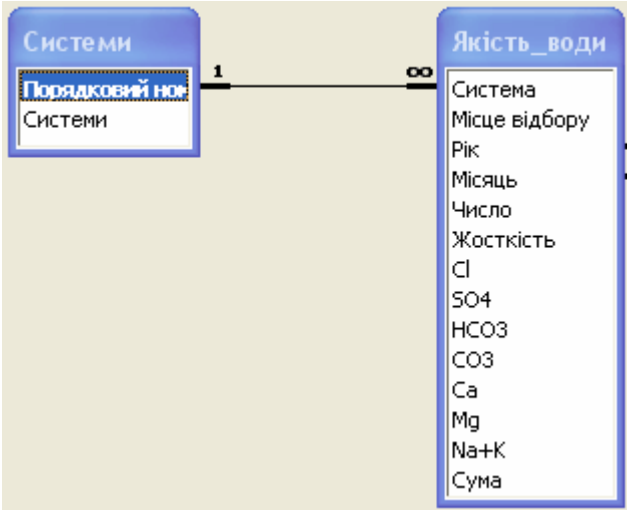


Рис. 2.1. Приклад взаємозв’язку між даними "один до багатьох"

Якщо інформація про ключову ознаку недостатня, то не можна просуватися по дереву. У дійсності, природа ієрархічної системи вимагає

явного визначення кожної відносини для того, щоб створити саму структуру і її правила розгалуження. Головною перевагою такої системи є те, що в ній дуже легко шукати, оскільки вона добре визначена і може відносно легко розширюватися додаванням нових галузей і формулюванням нових правил розгалуження. Для створення ієрархічної структури необхідне знання всіх можливих питань, що можуть задаватися, оскільки ці питання використовуються як основа для розробки правил розгалуження або ключів [37].

Мережні структури. Можливості швидкого пошуку, виконуваного в ієрархічній структурі даних, визначаються структурою самого дерева. Атрибутивні і геометричні дані можуть зберігатися в різних місцях, що зажадає установлення великого числа зв'язків між графічною й атрибутивною частинами БД. У такому випадку потенційне число розгалужень і зв'язаних з ними ключів ієрархічної структури може стати дуже великим. Мережні БД ГІС використовують відношення "багатьох до багатьох" (рис. 2.2), при якому один елемент може мати багато атрибутів, при цьому кожен атрибут зв'язаний явно з багатьма елементами. Для реалізації таких відносин разом з кожним елементом даних може бути зв'язане спеціальне перемінне, що має назву покажчик, яке направляє до всіх інших елементів даних, зв'язаних з цим. Замість того, щоб обмежуватися деревоподібною структурою зв'язків, кожен окремий елемент даних може бути прямо зв'язаний з будь-яким місцем бази даних, без уведення відносини "предок-нащадок".

Рис. 2.2. Взаємозв’язок між даними "багатьох до багатьох"

Мережні структури звичайно розглядаються як удосконалення ієрархічних структур, оскільки вони менш тверді і можуть представляти відношення "багато до багатьох". Тому вони допускають набагато велику гнучкість пошуку, ніж ієрархічні структури. Також на відміну від ієрархічних структур вони зменшують надмірність даних. Їхнім головним недоліком є те, що у великих БД ГІС кількість покажчиків може стати дуже великим, вимагаючи значної витрат пам'яті. Додатково, хоча зв'язки між елементами даних більш гнучкі, вони все-таки повинні бути явно визначені за допомогою покажчиків.

Реляційні бази даних. Недоліків великої кількості покажчиків можна уникнути використовуючи ще одну структуру баз даних - реляційну. В ній дані зберігаються як упорядковані записи або рядки значень атрибутів. Атрибути об'єктів групуються в окремих рядках у виді так званих відносин, оскільки вони зберігають свої положення в кожнім рядку і виразно зв'язані один з одним [Pr.G. Healey, 1991]. Кожний стовпчик містить значення одного атрибута для всього набору об'єктів.

Реляційні системи засновані на наборі математичних принципів, що мають назву реляційна алгебра або алгебра відносин [J.D. Ullman, 1982], що встановлює правила проектування та функціонування таких систем. Оскільки реляційна алгебра ґрунтується на теорії множин, кожна таблиця відносин функціонує як безліч, і перше правило говорить, що таблиця не може мати рядок, що цілком збігається з яким-небудь іншим рядком. Оскільки кожний з рядків унікальний, одна або трохи колонок можуть використовуватися для визначення критерію пошуку, визначеного імені з першого стовпчика. Такий критерій пошуку називається первинним ключем для пошуку значень в інших колонках бази даних (рис. 2.3) [11].

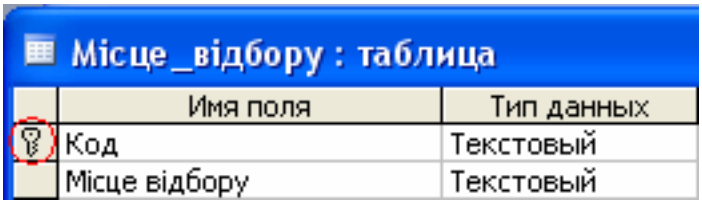


Рис. 2.3. Приклад ключового поля в базі даних

Реляційні системи коштовні тим, що дозволяють нам збирати дані в досить прості таблиці, при цьому задачі організації даних також прості. При необхідності можна стикувати рядки з однієї таблиці з відповідними рядками в іншій таблиці, використовуючи сполучний механізм, називаний реляційним з'єднанням. Будь-яка кількість таблиць може бути "зв'язана". З'єднання відбувається по рівності значень стовпчика первинного ключа однієї таблиці з іншим стовпчиком другої таблиці. Стовпчик другої таблиці, з яким зв'язаний первинний ключ, називається зовнішнім ключем.

Для визначення виду, який таблиці повинні мати, встановлений набір правил - нормальні форми [E.F. Codd, 1970]. Перша нормальна форма затверджує, що таблиця повинна складатися з рядків та стовпчиків і, оскільки стовпчики будуть використовуватися як ключі пошуку, у кожній з них на кожнім рядку повинне знаходитися тільки одне значення. Друга нормальна форма вимагає, щоб кожен стовпчик, що не є первинним ключем, цілком залежав від первинного ключа. Це спрощує таблиці і зменшує надмірність обмеження, що кожен рядок даних може бути знайдений тільки через його первинний ключ. Третя нормальна форма, зв'язана з другою, вимагає, щоб стовпчики, що не є первинним ключем, "залежали" від первинного ключа, у той час, як первинний ключ не залежить від якого-небудь не первинного ключа. Правила нормальних форм були підсумовані В. Кентом [W. Kent, 1983]. Ці правила досить корисні і повинні строго виконуватися.

2.3 Багатошарові моделі даних ГІС

У той час, як растрові і векторні структури даних дають засоби відображення окремих просторових феноменів на окремих картах, все-таки існує необхідність розробки більш складних підходів, що мають назву моделі даних, для включення в базу даних взаємин об'єктів, зв'язування об'єктів та їхніх атрибутів, забезпечення спільного аналізу декількох шарів карти (рис. 2.4).

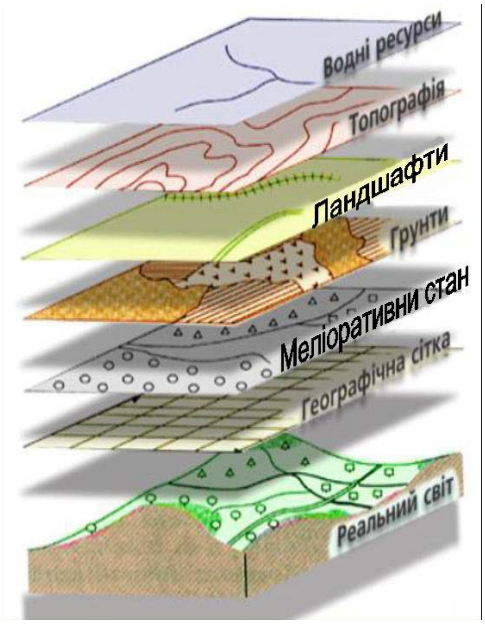


Рис. 2.4. Схематичний приклад накладання шарів карти

Растрові моделі. Кожен осередок у найпростішій такій структурі, зв'язаний з одним значенням атрибута. Для створення растрової тематичної карти збирають дані про визначену тему у формі двомірного масиву осередків, де кожен осередок представляє атрибут окремої теми. Такий двомірний масив називається покриттям. Використовують покриття для представлення різних типів тематичних даних (землекористування, рослинність, тип ґрунту, поверхнева геологія, гідрологія і т.д.) (рис 2.5).

Існує кілька способів збереження й адресації значень окремих осередків растра, їхніх атрибутів, назв покрить і легенд. У цій моделі кожен осередок містить всі атрибути начебто вертикального стовпчика значень, де кожне значення відноситься до окремої теми. Так, значення атрибута типу ґрунту в позиції X=10, У=10 буде знаходитися поруч зі значенням атрибута типу рослинності в тій же позиції X= 10, У= 10.

У той час, як растрові ГІС традиційно розроблялися для представлення одиночних атрибутів, збережених індивідуально для кожного осередку растра, деякі з них достигли стану використання прямих зв'язків з існуючими СУБД. Такі розширення растрової моделі даних дозволили також установити прямий зв'язок з ГІС, що використовують векторну структуру графічних даних. Оскільки такі інтегровані растрово-

векторні системи включають модулі, що перетворять інформацію з растрової форми у векторну і назад, користувач може використовувати переваги обох структур даних. Процес перетворення часто прозорий, так що користувачеві навіть не потрібно турбуватися про вихідну структуру даних.

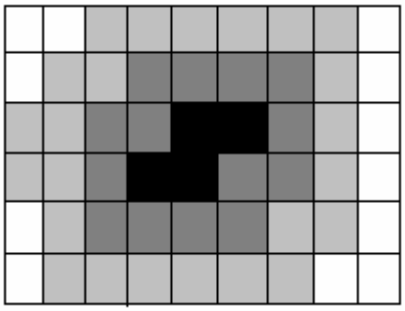


Рис. 2.5 Растрова модель даних

Ця можливість особливо важлива, оскільки вона підсилює взаємодію між програмним забезпеченням традиційної обробки цифрових зображень і геоінформаційними системами.

Векторні моделі даних. Векторні структури даних дають представлення географічного простору, більш інтуїтивно зрозумілим способом і очевидно більше нагадують добре відомі паперові карти. Існують кілька способів об'єднання векторних структур даних у векторну модель даних, що дозволяє досліджувати взаємозв'язки між показниками всередині одного покриття або між різними покриттями. Найпростішою векторною структурою даних є спагеті - модель [I. Dangermond, 1982], що по суті переводить "один в один" графічне зображення карти (рис. 2.6).

У цій моделі сусідні області повинні мати різні ланцюжки спагеті для загальних сторін. Тобто, не існує областей, для яких який-небудь ланцюжок спагеті був би загальним. Кожна сторона кожної області має свій унікальний набір ліній і пару координат. На відміну від спагеті - моделі, топологічні моделі [I. Dangermond, 1982], містять топологічну інформацію в явному виді. Для підтримки сучасних аналітичних методів потрібно внести в комп'ютер якнайбільше явної топологічної інформації. Топологічна модель даних поєднує рішення деяких з найбільш часто використовуваних у географічному аналізі функцій. Це забезпечується включенням у структуру даних інформації про суміжність для усунення необхідності визначення її при виконанні багатьох операцій. Топологічна інформація описується набором вузлів і дуг. Вузол (більш, ніж просто крапка, звичайно це перетинання двох або більше дуг, і його номер використовується для посилання на будь-яку дугу, якій він належить. Кожна дуга починається і закінчується або в крапці перетинання з іншою дугою, або у вузлі, що не належить іншим дугам. Дуги утворяться послідовностями відрізків, з'єднаних проміжними (формотворними) крапками. У цьому випадку кожна лінія має два набори чисел: пари координат проміжних крапок і номера

вузлів. Крім того, кожна дуга має свій ідентифікаційний номер, що використовується для вказівки того, які вузли представляє її початок і кінець.

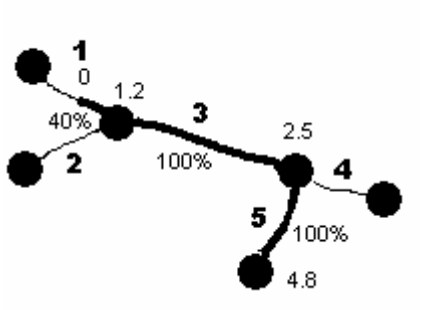


Рис 2.6 Векторна модель даних

Неефективність збереження і пошуку, властивої базової топологічної моделі усувається роздільним збереженням кожного типу об'єктів (крапки, лінії, області). Ці окремі об'єкти потім зв'язуються в ієрархічну структуру даних, де крапки, через покажчики, зв'язані з лініями, а лінії - з областями. Кожен набір відрізків, називається у даній моделі ланцюжком, починається і закінчується у визначених вузлах (перетинаннях двох ланцюжків) (рис. 2.7).

Існування гібридної моделі даних ГІС - підтвердження того, що хоча її структури даних ефективні стосовно графічних характеристик об'єктів, їм не дістає тієї ж ефективності в управлінні атрибутивними даними [P. Aronson, 1985; S.Morehouse, 1985]. І навпаки, СУБД загальновизнані як засіб управління атрибутивними типами даних, але погано пристосовані до роботи з графічними об'єктами. Виглядає цілком логічним, що програмне об'єднання цих двох технологій дозволить узяти краще з кожної. Для реалізації цього підходу координатні і топологічні дані, необхідні для графіки, зберігаються як окремий набір файлів. Таблиці атрибутів, що містять усі необхідні описові дані для кожного графічного об'єкта, зберігаються окремо або в інших файлах, або під управлінням СУБД загального призначення. Зв'язок між графікою й атрибутами здійснюється через ідентифікаційні коди графічних об'єктів, що маються в графічних файлах, і які також зберігаються в окремому стовпчику атрибутивної БД. Завдяки можливості зовнішнього збереження багатьох атрибутів для кожного об'єкта ростуть аналітичні можливості і можлива економія пам'яті.

Іншим підходом до збереження графічних і атрибутивних даних є інтегрована модель даних. У цьому випадку ГІС є процесором просторових запитів, надбудованим над стандартною СУБД, що використовується для збереження як атрибутивної, так і графічної інформації [Gaptill, 1987; Morehouse, 1989]. Інтегрована система зберігає координати об'єктів карти й атрибути в різних таблицях однієї БД, що зв'язуються механізмом,

подібним реляційному з'єднанню [R.G. Healey, 1991]. Крім того, атрибути можуть розміщатися в тих же таблицях, що і графіка.

Існують два способи збереження координатної інформації в реляційних таблицях. У першому записуються окремі пари координат, що представляють крапкові об'єкти, а також кінцеві та проміжні крапки ліній і границь областей, як індивідуальні атоми, або рядки, бази даних. Інтегрована модель може записувати в один стовпчик таблиці цілі ланцюжки координатної інформації. Таким чином, одна область може бути описана одним рядком таблиці, що містить в одній колонці ідентифікатор області, а в іншій - список ідентифікаторів ліній. Тоді лінії, ідентифікуємі за цим кодом в окремому стовпчику таблиці ліній описували би розташування області набором пари координат.

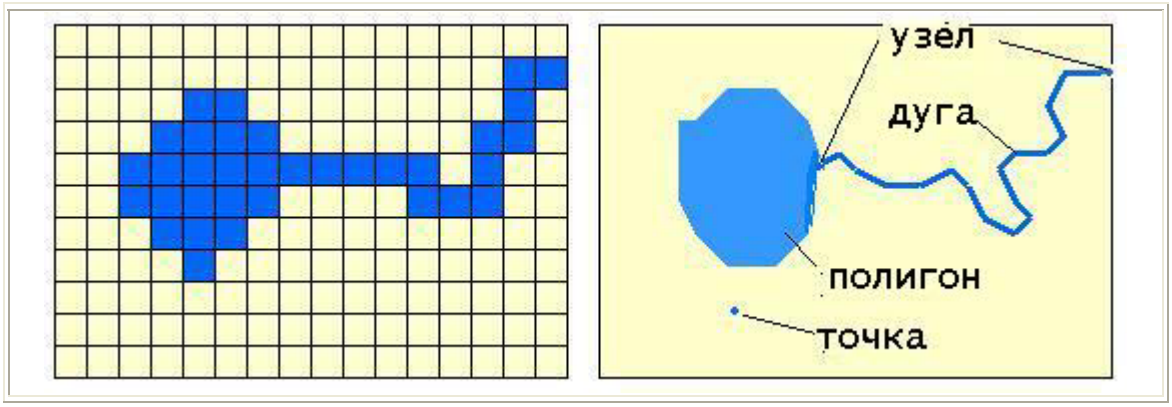


Рис.2.7 Співставлення растрової та векторної моделей просторових даних

Крім двох вже розглянутих моделей високого рівня на сцену виходить третя, що називається об'єктно-орієнтованою моделлю даних. Ця модель включає мову просторових запитів [R.G.Healey, 1991] і відбиває визнання того факту, що потрібен об'єктно-орієнтований доступ, і до БД ГІС, і до виконуємих з нею операціям. Ідеї, що лежать в основі цих систем практично ідентичні об'єктно-орієнтованому підходові в програмуванні [P.Aronson, 1987].

2.4 Введення, збереження та редагування БД ГІС

У растрових системах головними даними є значення атрибутів осередків растра, що зберігаються в комп'ютері звичайно на твердому диску. Положення кожного осередку растра визначається щодо положень інших осередків растра. З цієї причини редагування зв'язане головним чином з правильним відносним положенням кожного осередку растра.

Якщо растрова система забезпечує зв'язок з зовнішньою СУБД, питання стає трохи складнішим у тому, що кожному осередкові растра приєднані кілька різних кодів атрибутів.

У випадку векторів графіка й атрибути зберігаються, або як окремі таблиці всередині однієї БД, або як самостійні набори даних, зв'язані набором покажчиків. Поділ графіки й атрибутів вимагає уваги до процедур редагування, застосовуваним до графіки, атрибутам і базам даних. Багатовекторні ГІС дозволяють зберігати окремо частини БД, як великі секції для цілей архівування. Ця процедура, називається мозаїчним розміщенням, найчастіше використовується для зменшення обсягу даних, необхідних для одноразового аналізу в дуже великих БД. Хоча мозаїка не вимагає застосування такої формальної схеми, багато хто вважають її корисною для спрощення управління даними.

Найчастіше БД цілком редагується і вичищається перед мозаїчною розбивкою, архівацією і визначенням доступу для відновлення й аналізу. Але так буває не завжди, і тоді доведеться вибирати підходящі блоки для редагування. У деяких випадках може знадобитися виконання операції ув'язування по границях блоків для забезпечення стикування частин об'єктів, що перетинають границі блоків.

Загалом, сучасне програмне забезпечення ГІС, будь то растрове, векторне або на квадродереві, забезпечує механізм візуального відображення, що підвищує можливості візуалізації помилок. Конкретні методи будуть залежати від використовуваної моделі даних і складності системи. Оскільки більшість систем дають можливість інтерактивного редагування усередині підсистеми візуалізації, то звичайно мається також і можливість коректувати помилки безпосередньо при виявленні кожної з них.

Хоча деякі помилки можуть відбуватися в результаті недоліків обчислювальних алгоритмів, помилок кодування програм і помилок округлення, і це дійсно трапляється час від часу, усе-таки більшість помилок у БД обумовлені неправильним введенням.

Перший тип помилок відноситься головним чином до векторних систем і називається графічною помилкою. Такі помилки зустрічаються трьох видів: пропуск об'єкта, невірне положення об'єкта і невірний порядок об'єктів.

Другий тип помилок - це помилки атрибутів. Вони зустрічаються й у векторних і в растрових системах, з однаковою частотою. Найчастіше вони є помилками, а величезний обсяг роботи, що вимагається для великих БД, часто виявляється головним джерелом помилок. У векторних системах помилки атрибутів включають використання невірного коду для атрибута, помилки запису однакових за вимовою, але різних за написанням слів, що унеможливлює вибірку атрибута, якщо в запиті використаний коректний запис. У випадку растра введення найчастіше складається з атрибутів, тому результатом набору невірного коду або переміщення його в невірний осередок растра є карта, що показує ці невірно кодовані осередки в невірних місцях. Такі невірно розташовані атрибутивні дані утворять третій тип помилок, помилки узгодження графіки й атрибутів, що трапляються й у

векторних системах, коли вірно набрані коди атрибутів зв'язуються з невірними графічними об'єктами.

Методи, за допомогою яких це буде зроблено, залежать до деякої міри від наявного устаткування і від конкретної системи, що під рукою. Незалежно від того, що за система і як вводять у неї просторові дані, підсистема введення буде мати загальні з іншими характеристики.

По-перше, вона спроектована для переносу графічних і атрибутивних даних у комп'ютер. По-друге, вона повинна відповідати хоча б одному з двох фундаментальних методів представлення графічних об'єктів - растровому або векторному. По-третє, вона повинна мати зв'язок з системою збереження та редагування, щоб гарантувати збереження і можливість вибірки того, що введено, й що можна буде усувати помилки і вносити зміни в міру необхідності.

Правило номер один говорить: насамперед визначте, для чого створюється БД ГІС. Це щонайменше обмежить уведення даних темами, що швидше за все будуть використовуватися. Тематичні покриття, що вводяться, повинні бути прямо зв'язані з моделюванням і аналізом, що повинен виконуватися, і результатами, які треба одержати.

Необхідність визначення того, які покриття знадобляться в майбутньому, являє собою деяку проблему. Єдиним випадком, коли база даних може створюватися без чіткого розуміння передбачуваного результату (іноді називаного просторово-інформаційним продуктом, є проекти, головна мета яких - визначити можливі взаємозв'язки між даними тематичних покрить для формулювання початкової наукової гіпотези. Цей підхід не прийнятний для комерційних проектів.

Тому правило номер два, зв'язане з першим, вимагає як можна найбільш точного визначення цілей перед вибором тематичних покрить.

Правило третє: уникайте використання даних з екзотичних джерел, коли маються звичайні, особливо якщо останні забезпечують подібний рівень точності.

Правило четверте: використовуйте найкращі, найбільш точні дані, необхідні для вашої задачі.

Правило п'яте: вибирайте адекватний рівень точності даних.

Більшість тематичних карт містять також інформацію про дороги й інші антропогенні об'єкти, що можуть бути дуже корисними для введення в ГІС. Скрізь, де можливо, і де їхня якість прийнятна, належить уводити ці дані як окремі покриття з того ж листа карти.

Правило шосте, говорить, що кожне покриття повинне бути як можна більш спеціалізованим. Тобто покриття повинні бути як можна більш спеціалізовані за темами. Чим більш спеціалізоване покриття, тим легше виконувати пошук, якщо треба довідатися про щось, що відноситься до даних, які утримуються в одному покритті.

Питання для самоперевірки

1. Типи моделей даних.
2. Основні характеристики і можливості застосування ієрархічної, мережної й реляційної моделей даних.
3. Особливості багатошарових моделей ГІС.
4. Введення, збереження та редагування БД ГІС.
5. Структури баз даних для управління даними.
6. Трьохрівнева архітектура даних.
7. Взаємозв’язки між даними.
8. Растрова модель даних.
9. Векторна модель даних.
10. Спагеті – модель даних.
11. Поняття квадродерево.
12. Типи помилок в БД.

3. СУЧАСНІ ПІДХОДИ ДО СТВОРЕННЯ БАЗ ДАНИХ

3.1 Реляційні бази даних

У переважній більшості СУБД для персональних комп'ютерів інформація організовується у вигляді двомірних таблиць, і їх часто, хоча й не завжди коректно, називають реляційними базами даних.

Реляційні бази даних з'явилися із прагнення перебороти ці недоліки й забезпечити прості та гнучкі способи організації даних. Реляційну базу даних можна визначити як сукупність зв'язаних таблиць. У такій базі інформація організується у вигляді двовимірних таблиць, зі стовпчиками - полями й рядками - записами, у яких зберігаються як самі дані, так й інформація про зв'язки між ними.

Звичайно полів і записів незмірно більше. Кожне поле містить найменший елемент даних, якому можна зберігати в таблиці. Це може бути число, слово, якийсь текст або навіть зображення, музика, відеокліп і т.п. Запис (рядок таблиці) являє собою сукупність полів і містить одну копію кожного певного в базі даного поля, навіть у тому випадку, якщо поле не містить ніякої інформації. Безліч записів утворять таблицю. Безліч зв'язаних між собою таблиць утворять **реляційну базу даних** [41].

Майже всі продукти баз даних, створені з кінця 70-х років, засновані на підході, що називають реляційним; більше того, переважна більшість наукових досліджень в області баз даних протягом останніх 25 років проводилася (можливо, побічно) у цьому напрямку. Реляційний підхід являє собою основну тенденцію сьогоднішнього ринку, і реляційна модель - єдина найбільш істотна розробка в історії розвитку баз даних.

Отже, коротенько, реляційна система - це система, заснована на наступних принципах:

1) дані для користувача передаються у вигляді таблиць (і ніяк інакше);
2) користувачеві надаються оператори (наприклад, для вибірки даних), що генерують нові таблиці зі старих. Наприклад, у системі буде оператор одержання підмножини рядків у даній таблиці й оператор одержання підмножини стовпців - а підмножину рядків і підмножину стовпців, звичайно, можна розглядати як нові таблиці.

Реляційна база даних — це база даних, сприймана користувачем як набір нормалізованих відносин (тобто змінних відносин) різного ступеня.

Вираження "сприймана користувачем" є вирішальним: ідея реляційної моделі застосовується до зовнішнього й концептуального рівнів системи, а не до внутрішнього рівня. Можна сказати й інакше: реляційна модель представляє систему баз даних на рівні абстракції, трохи вилучену від подробиць лежачої в основі цієї системи машини; так само як, наприклад, мова Pascal представляє систему програмування на рівні абстракції, трохи вилучену від подробиць лежачої в основі цієї системи машини. У дійсності реляційну модель можна розглядати як мову програмування, спеціально орієнтовану на додатки баз даних.

Розходження між доменами й (іменованими) відносинами також можна трохи уточнити. Іменовані відносини називаються змінними, тому що їхні значення змінюються згодом. Домени не є змінними в цьому змісті.

В традиційних термінах відношення відповідає файлу (логічному, а не фізичному), кортеж — запису (екземпляру, а не типу), атрибут полю (типу, а не екземпляру). Однак ця відповідність є приблизною. Відношення варто розглядати не як "просто файл", а як файл, що підкоряється певним правилам, це відбивається в значному спрощенні структури об'єктів даних, з якою зіштовхується користувач, і, як наслідок, у спрощенні операторів, необхідних для роботи з цими об'єктами. Якщо говорити точніше, всі дані в реляційній системі представляються одним і тільки одним способом, а саме явними значеннями (цю властивість іноді називають "основним принципом реляційної моделі", а також "інформаційним принципом"). Зокрема, цими явними значеннями представляються логічні зв'язки у відношенні й між відносинами; не існує видимих для користувача покажчиків на файли або записи, не існує видимого для користувача порядку записів, не існує видимих для користувача груп повторення й т.д.

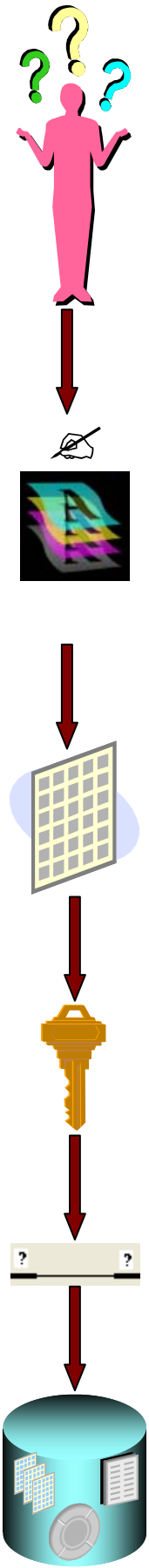
Файли .DBF стандарту dBASE являють собою відображення двовимірної таблиці зі стовпцями - полями й рядками - записами. При пошуку інформації в цих файлах часто доводиться використовувати відомості про положення даних у файлі (номер рядка таблиці, номер запису файлу .DBF) і щодо цього стандарт dBASE не задовольняє вимогам, пропонованим до реляційних баз даних. Поки бази даних на персональних комп'ютерах були відносно невеликі і їх можна було розмістити в одному файлі .DBF, ця обставина не грала істотної ролі, а звична простота таблиці залучала до цього способу організації інформації численних користувачів. Але при збільшенні розмірів баз даних зберігати їх в одній таблиці стає неможливим і виникає необхідність виконання інших вимог реляційної моделі.

Відносини типу один-до-багатьох використовуються, коли до одного запису першої таблиці необхідно поставити у відповідність кілька записів другої таблиці. Слід зазначити, що визначати відносини між таблицями "у ручну", заняття досить трудомістке. Але цей процес легко може бути автоматизований, або за допомогою макросів Microsoft Access, або за допомогою програми на Visual BASIC.

У дев'яності роки реляційна модель даних перетворилася в основний засіб організації інформації в базах даних не тільки на персональних комп'ютерах, але й на більших ЕОМ. У рамках цієї моделі була розроблена мова структурованих запитів SQL, який став основним засобом для одержання інформації з реляційних баз даних.

3.2 Етапи проектування бази даних

Етапи проектування бази даних мають наступний вигляд.



1. Визначте мету створення бази даних, основні її функції й інформацію, яку вона повинна містити. База даних повинна відповідати вимогам тих, хто буде безпосередньо з нею працювати. Для цього потрібно визначити теми, які повинна покривати база даних, звіти, які вона повинна видавати, проаналізувати форми, які в даний момент використовуються для запису даних, порівняти створювану базу даних з добре спроектованою, подібною їй базою.
2. Розробіть на папері структуру таблиць, які повинна містити база даних. При проектуванні таблиць, рекомендується керуватися наступними основними принципами:
   - інформація в таблиці не повинна дублюватися. Не повинне бути повторень і між таблицями. Коли певна інформація зберігається тільки в одній таблиці, то й змінювати її прийдеться тільки в одному місці. Це робить роботу більш ефективною, а також виключає можливість розбіжності інформації в різних таблицях;
   - кожна таблиця повинна містити інформацію тільки на одну тему. Відомості на кожну тему обробляються набагато легше, якщо утримуються вони в незалежних друг від друга таблицях.
3. Визначте необхідні в таблиці поля. Кожна таблиця містить інформацію про окрему тему, а кожне поле в таблиці містить окремі відомості по темі таблиці. При розробці полів для кожної таблиці необхідно пам'ятати:
   - кожне поле повинне бути пов'язане з темою таблиці;
   - не рекомендується включати в таблицю дані, які є результатом вираження;
   - у таблиці повинна бути присутня вся необхідна інформація.
4. Задайте ключове поле. Для того, щоб Microsoft Access міг зв'язати дані з різних таблиць, кожна таблиця повинна містити поле або набір полів, які будуть задавати індивідуальне значення кожного запису в таблиці. Таке поле або набір полів називають основним ключем.
5. Визначте зв'язки між таблицями. Після розподілу даних по таблицях і визначення ключових полів необхідно вибрати схему для зв'язку даних у різних таблицях. Для цього потрібно визначити зв'язки між таблицями.
6. Ще раз перегляньте структуру бази даних і виявите можливі недоліки. Бажано це зробити на даному етапі, поки таблиці не заповнені даними.

7. Додайте дані й створіть інші об'єкти бази даних. Якщо структури таблиць відповідають поставленим вимогам, то можна вводити всі дані. Потім можна створювати будь-які запити, форми, звіти, макроси й модулі.
8. Використовуйте засоби аналізу в Microsoft Access. В Microsoft Access існує два інструменти для вдосконалення структури баз даних. Майстер аналізу таблиць (рис.3.1) досліджує таблицю, якщо буде потреба пропонує

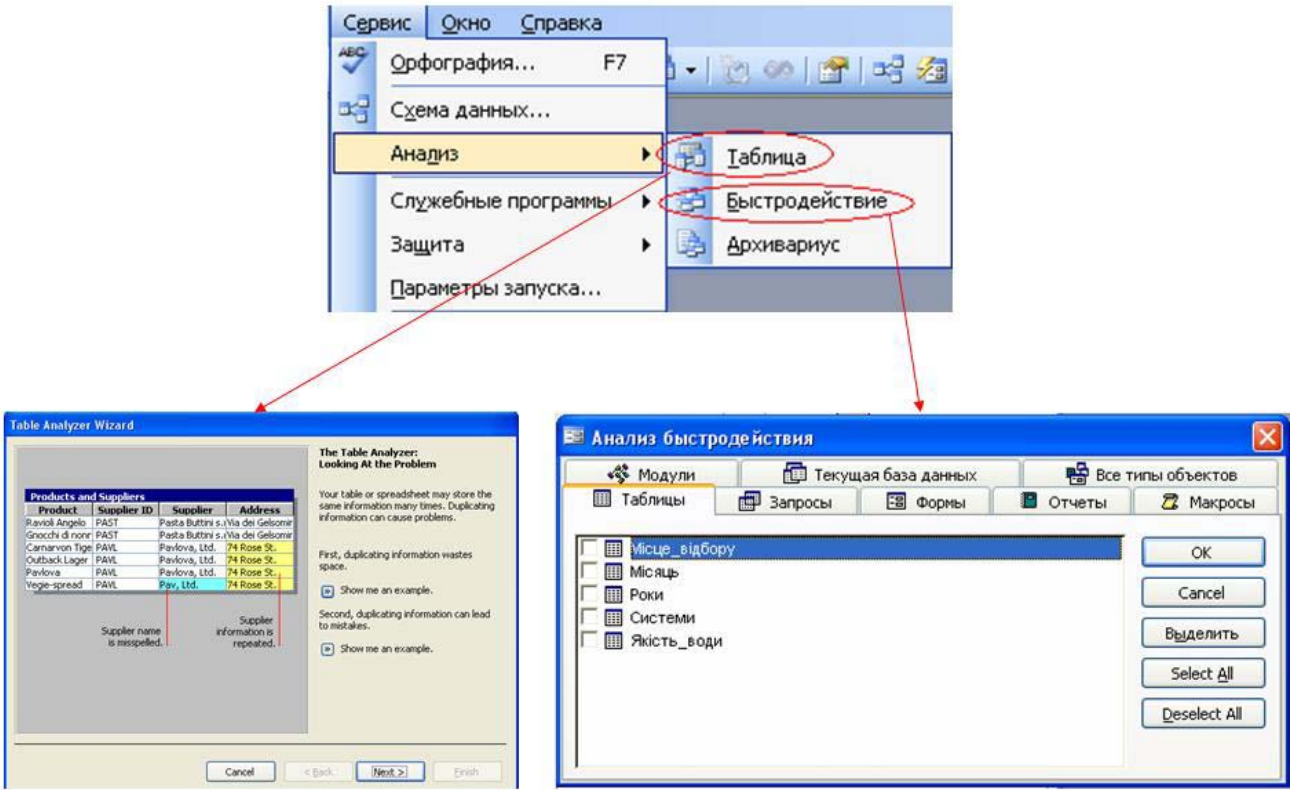


Рис.3.1. Майстер аналізу таблиць та аналізатор швидкодії

нову її структуру й зв'язки, а також переробляє її. Аналізатор швидкодії (рис. 3.1) досліджує всю базу даних, дає рекомендації з її поліпшення, а також здійснює їх.

3.3 Проектування БД: загальні положення

Крім основної ідеї просторово-інформаційних продуктів існують і інші сторони проектування ГІС для об'єкта. Коли маються кілька проектів, необхідно розглядати відповідні досліджувані області в окремості. Вони найчастіше ґрунтуються на формальному рішенні про те, чому область піддається вивченню. Так, для дослідження міського рівня, усе місто буде областю дослідження. Для дослідження, зв'язаного з природними явищами або об'єктами, як, наприклад, забруднення ріки, весь водозбірний басейн ріки може бути обраний як область вивчення, оскільки забруднення може приходити з будь-якої території, що дає стік в один із припливів ріки. Коротше кажучи, область вивчення може вибиратися на основі політичних і

адміністративних границь, фізичних границь, границь власності, або відбивати, головним чином, фінансові обмеження або обмеження за даними, що виявляються на ранніх стадіях процесу проектування.

Якщо для деякої частини передбачуваної області вивчення маються більш докладні дані, чим для іншої частини, вони не повинні диктувати розмір області вивчення. Цю обставину можна використовувати таким чином, що невелика частина з більш докладними даними може відігравати роль прототипу для детального аналізу всієї області вивчення, у розрахунку на те, що надалі і на всю область можна буде одержати дані такої ж подробиці. Крім того, вони можуть бути придатні для поліпшення знання про всю область вивчення.

Коли планується використовувати який-небудь вид інтерполяції, границя області вивчення повинна бути розширена в достатньому ступені, щоб забезпечити коректні результати інтерполяції на її краях. І, нарешті, чим більше область вивчення, тим більше грошей і часу буде потрібно на створення бази даних. Часто саме питання вартості є головним обмежником у встановленні границь області вивчення. Крім цього, розширення області вивчення підвищує імовірність недоліку даних для всіх покрить, необхідних для проведення аналізу.

Масштаб, дозвіл і рівень детальності. Існує зв'язок між розміром досліджуваної області і масштабом карт, що вводяться. Чим дрібніше масштаб карти, тим вище ступінь генералізації, тим менш точне представлення об'єктів реального світу. На вибір масштабу впливає важливість відповідного покриття для моделей аналізу. Хоча не існує загальних указівок для визначення підходящого масштабу, більшість професіоналів використовують підхід "найкращих доступних даних", вважаючи, що краще більше подробиць, чим менше.

Растрові дані мають особливість: зі зменшенням розміру осередків растра швидко зростає обсяг даних. І хоча сучасні комп'ютери мають досить ємні пристрої збереження, великий обсяг даних значно сповільнює виконання операцій над покриттями, що використовують пошук по периферії. В одних випадках розмір осередків растра може диктуватися найменшим об'єктом, що представляється, в інші - вимогами моделі [DeMers, 1992], у третіх - питаннями сумісності з іншими цифровими даними.

Класифікація. При проектуванні ГІС треба розглядати не тільки наявні джерела даних, але і систему класифікації, що задовольняє вимогам моделювання. Рішення про останнє краще приймати після розгляду видів даних, що вводяться. Використання більш докладної класифікації часто переважно за двома причинами; вона дає користувачеві більший обсяг відомостей, і при порівнянні з даними іншого покриття меншої подробиці завжди можна агрегірувати деякі класи, у той час як зворотний процес, або утруднений, або зовсім неможливий.

Але класифікація - більше чим просто вибір належного рівня детальності. Потрібно врахувати, що робиться порівняння між покриттями,

і, таким чином, між класифікаціями. Крім того, часто необхідно мати можливість проводити погоджену класифікацію в межах одного покриття даних з різних джерел. Допустимо, потрібно покриття з класами ґрунтів регіонального охоплення, створене на основі різних досліджень ґрунтів більш детального окружного рівня. Якщо ці дослідження проводилися на значному тимчасовому видаленні друг від друга, то буде необхідно переробити них, щоб домогтися однорідності класифікації Бэрроу [P.A.Burrough, 1986].

Система координат і проекція. Вибір проекції визначається площею області вивчення і доступних даних. При цьому варто враховувати, що перетворення проекцій вносять додаткову погрішність у дані. На вибір проекції впливає також те, які характеристики земної поверхні повинні зберігатися (звичайно ті, що найбільш важливі для аналізу). Загалом, і стосовно системи координат, і стосовно проекції, просторова і тимчасова сумісність дуже важливі для коректності процесу прийняття рішень.

Введення даних в ГІС. У випадку ГІС одна тільки побудова бази даних часто займає три чверті часу. У комерційних додатках це означає, що три чверті вартості системи також піде на цю операцію і редагування. Є багато способів уведення даних. Одні виглядають примітивними, начебто переміщення прозорої сітки на карту. Інші - більш сучасні, тому що використовують пристрої цифрового введення — дигітайзери і сканери.

Дигітайзер (**digitizer**), цифруватель - пристрій для вводу в комп'ютер координат крапок із прикріпленого до планшету (tablet) листа за допомогою рухомого курсору з клавіатурою [36].

Сканер (**scanner**) - пристрій для сканування, т.є. автоматизованого оцифрування растрових зображень: а) периферійний пристрій комп'ютера; б) сенсор дистанційного зондування (наприклад, MSS, Thematic Mapper) [36].

Питання для самоперевірки

1. Підходи до проектування реляційних баз даних.
2. Методи і моделі, за допомогою яких виконується опис и структуруються різноманітні предметові середовища.
3. Класична методологія проектування баз даних.
4. Етапи проектування баз даних.
5. Реляційні бази даних, їх характеристика.
6. Принципи реляційної системи.
7. Мета створення бази даних.
8. Інструменти вдосконалення структури баз даних в
9. Зв’язок між розміром досліджуваної області та масштабом карт.

4. КОНЦЕПТУАЛЬНА МОДЕЛЬ БАЗИ ДАНИХ

4.1 Концептуальна модель організації даних

Концептуальна модель даних необхідна винятково для проектування і виступає як засіб відображення уявлень людини про системи та процеси предметної області реального світу, сприяє їхньому розумінню. Вона описує об'єкти предметної області та їхні взаємозв'язки на основі сформованих загальносистемних принципів і вимог до **організації** інформаційних баз без зазначення засобів їхнього **фізичного** зберігання [21].

Проектування інформаційних баз передбачає три основних рівні організації даних — концептуальний, логічний і фізичний.

Першим етапом розробки будь-якої інформаційної системи має стати побудова **концептуальної моделі організації даних,** як засобу точного відображення уявлень людини про системи та процеси предметної області реального світу. При формуванні концептуальної моделі основна увага спрямовується на структуризацію даних і виявлення взаємозв'язків між ними без розгляду особливостей реалізації та ефективності обробки.

Предметною областю дорадчих систем, побудованих на базі ГІС, є територія з усією притаманною їй специфікою природних умов, ресурсним потенціалом, поширеними в її межах видами господарської діяльності.

Для організації даних важливі не тільки взаємозв'язки, що здійснюються у просторових системах реального світу, але й заснована на них інформаційна взаємодія між структурно-функціональними одиницями різних рангів у самій інформаційній базі. Тому використання ГІС як засобу вирішення завдань у територіальному плані потребує, з одного боку, формалізації геоінформації, з іншого, — її структуризації на основі визначення об'єктів інформаційної діяльності.

Головним об'єктом інформаційної діяльності і відповідно структуризації інформації є природно — агромеліоративна геосистема, що поєднує в собі елементи природно-ресурсної основи, виробничо-технологічної структури території. Концептуальна модель організації даних описує об'єкти предметної області та їхні взаємозв'язки саме в рамках ПАМС. Вона побудована за принципами функціональної диференціації інформації з виділенням базових та спеціалізованих (функціональних) модулів.

Вона включає чотири основні підсистеми, що описують елементи природно-ресурсної основи (ПС-1), виробничо-технологічної структури (ПС-2) території, територіальної прив'язки об'єктів (ПС-3) і нормативно-регламентуючої бази прийняття рішень (ПС-4) [8].

4.2 Структура і технологія наповнення

Структура і склад інформаційної бази на різних рангах інтеграції даних з визначенням методів фіксації показників та форм представлення інформації обгрунтовані на основі сформованих систем вимог і концептуальних моделей організації даних у різних функціональних модулях ПІК об'єкту, які дають можливість створення адекватної інформаційної моделі території досліджень.

При цьому системну організацію тематичної інформації здійснюють, як було зазначено вище, з урахуванням адресних вимог бази знань і бази даних.

База знань формується як сукупність знань для обгрунтування та оптимізації прийняття рішень. Концептуальна модель організації даних у її межах передбачає диференціацію інформації з видокремленням нормативно-регламентуючої бази (НРБ) і системи загальнодовідкових даних, що забезпечує вибір параметрів на стадії ідентифікації об'єктів.

Нормативно-регламентуюча база. Одним із перших етапів створення бази знань є розробка нормативно-регламентуючої бази з системою кодифікації та відповідними експертними системами, що забезпечують вибір оптимальних умов вирощування тієї чи іншої сільськогосподарської культури, визначення складу і параметрів технологічних операцій рослинництва.

У її основу покладено структуризацію даних у середині кожного з тематичних та системних модулів ПС-1, ПС-2, ПС-3 з використанням класифікаційних схем як засобів описування вміщеної у них інформації.

Основою формування інформаційної бази при вирішенні завдань просторового оцінювання має бути територіально-галузева інформаційна база та природно-ресурсний паспорт об'єкта, який вміщує елементи інформаційно-довідкової і геоінформаційної баз даних [8].

Вихідною інформацією для побудови територіально-галузевої бази щодо природно-ресурсної основи регіону слугують матеріали:

- комплексних гідрогеологічних та інженерно-геологічних зйомок для цілей меліорації (М 1:50000-1:200000), виконаних організаціями Державного комітету природних ресурсів України;
- вишукувальних підрозділів і організацій Держводгоспу, Мінагрополітики, Держкомзему, з вивчення кліматичних, гідрогеологічних, ґрунтових, грунтово-меліоративних, інженерно-геологічних та екологічних умов зрошуваних і прилеглих до них земель (М 1:10000-1:200000);
- гідрогеолого-меліоративної і водогосподарської служб Держводгоспу України з вивчення меліоративного й еколого-меліоративного стану земель, технічного стану зрошувальних систем і водогосподарських об'єктів, стану діючих агроіригаційних навантажень і режимів експлуатації (М 1:10000-1:100000); оперативна

та довгострокова інформація мережі еколого-меліоративного моніторингового контролю й спеціального випробування;

• спеціальних (у тому числі експериментальних) досліджень впливу зрошення на природні комплекси, зокрема на стан і еволюцію грунтів, що виконані різними науковими установами країни;

• великомасштабних суцільних ґрунтових і агрохімічних обстежень, спеціального випробування ґрунтів (вміст поживних речовин і забруднювальних елементів, вологозапаси, мінливість продуктивності грунтів);

• космічних досліджень, аерофотозйомок, геофізичного зондування тощо;

• топооснови, плани землекористування тощо (М 1:2000-1:200000).

Створення регіональних баз знань, локальних інформаційно-довідкових і відповідно геоінформаційних баз даних здійснюється в автономному режимі для різних рівнів функціональних завдань на основі єдиних концептуальних моделей організації і структуризації інформації в рамках ПІК [8]

Концептуальне подання — це подання всієї інформації бази даних у трохи більш абстрактній формі у порівнянні з фізичним способом зберігання даних. Однак концептуальне подання істотно відрізняється від способу подання даних окремому користувачеві. Загалом кажучи, концептуальне подання - це подання даних такими, які "вони є насправді", а не такими, якими змушений їх бачити користувач у рамках, наприклад, певної мови або використовуваного апаратного забезпечення.

Концептуальне подання складається з безлічі екземплярів кожного типу концептуального запису. Концептуальний запис зовсім не обов'язково повинен збігатися із зовнішнім записом, з одного боку, і зі збереженим записом - з іншої.

Концептуальне подання визначається за допомогою концептуальної схеми, що включає визначення кожного типу концептуальних записів.

Концептуальна схема використовує іншу мову визначення даних — концептуальну. Щоб домогтися незалежності даних, не можна включати у визначення концептуальної мови будь-який розгляд структури зберігання або методу доступу. Визначення концептуальної мови повинні ставитися тільки до змісту інформації. Це означає, що в концептуальній схемі не повинно бути ніякого згадування про подання збереженого файлу, послідовності збережених записів, індексування, хеш-адресації, покажчиках або інших подробицях зберігання чи доступу. Якщо концептуальна схема дійсно забезпечує незалежність даних у цьому змісті, то зовнішні схеми, певні на основі концептуальної, свідомо будуть забезпечувати незалежність даних.

Концептуальне подання — це подання всього вмісту бази даних, а концептуальна схема — це визначення такого подання. Однак сьогоднішня система реально не підтримує такого концептуального рівня, що б на небагато наблизився до цього ступеня розвиненості; у більшості

існуючих систем "концептуальна схема" у дійсності являє собою трохи більше, ніж просте об'єднання всіх окремих зовнішніх схем з додатковими засобами безпеки й правилами забезпечення цілісності [10].

4.3 Відображення

Відображення концептуальний-внутрішній визначає відповідність між концептуальним поданням і збереженою базою даних, тобто як концептуальні записи й поля представлені на внутрішньому рівні. При зміні структури збереженої бази даних, тобто при внесенні змін у визначення структури зберігання, змінюється й відображення концептуальний-внутрішній таким чином, щоб концептуальна схема залишилася незмінною. Інакше кажучи, щоб зберегти незалежність даних, результати таких змін не повинні торкнутися концептуального рівня.

Відображення зовнішній-концептуальний визначає відповідність між деяким зовнішнім поданням і концептуальним поданням. Загалом, розходження, які можуть існувати між цими двома рівнями, подібні до розходжень між концептуальним поданням і збереженою базою даних. Між іншим, більшість систем дозволяє виражати визначення одного зовнішнього подання через інше (тобто за допомогою відображення зовнішній-зовнішній) не вимагаючи обов'язково явно визначати відображення на концептуальний рівень. Ця можливість корисна, якщо кілька подань схожі між собою. Зокрема, ця можливість є в багатьох реляційних системах.

Первинний ключ - це унікальний ідентифікатор для таблиці, тобто стовпець або така комбінація стовпців, що в будь-який момент часу не існує двох рядків, що містять однакове значення в цьому стовпці або комбінації стовпців.

І нарешті, **домен** - це загальна сукупність значень, з якої беруться справжні значення для певних атрибутів певного відношення [10]. Домени насамперед мають концептуальну природу. Вони можуть бути або не бути явно збережені в базі даних як реальні набори значень; фактично в більшості випадків вони не зберігаються. Але вони повинні бути, принаймні, визначені в рамках визначень бази даних і тоді кожне визначення атрибута повинне включати посилання на відповідний домен, у такий спосіб системі буде відомо, які атрибути можна порівнювати, а які - ні.

Щоб конкретизувати ідеї реляційної моделі використовується гіпотетично реляційна мова для ілюстрації цих ідей. Ясно, що в першу чергу потрібний спосіб створення нового домена: CREATE DOMAIN domain data-type ; тут domain— це ім'я нового домена, а data-type відповідає типу даних [10].

4.4 Інфологічна модель даних "Сутність-Зв'язок"

4.4.1 Основні поняття

Мета інфологічного моделювання - забезпечення найбільш природних для людини способів збору й подання тієї інформації, яку передбачається зберігати в створюваній базі даних. Тому інфологічну модель даних намагаються будувати за аналогією із природною мовою (остання не може бути використана у чистому виді через складність комп'ютерної обробки текстів і неоднозначності будь-якої природної мови). Основними конструктивними елементами інфологічних моделей є сутності, зв'язки між ними та їхні властивості (атрибути).

Сутність – будь-який помітний об'єкт (об'єкт, що можна відрізнити від іншого), інформацію про який необхідно зберігати в базі даних. Сутностями можуть бути люди, місця, літаки, рейси, смак, колір і т.д. Необхідно розрізняти такі поняття, як тип сутності й екземпляр сутності. Поняття тип сутності ставиться до набору однорідних особистостей, предметів, подій або ідей, що виступають як ціле. Екземпляр сутності ставиться до конкретної речі в наборі. Наприклад, типом сутності може бути МІСТО, а екземпляром - Херсон, Київ і т.д.

Атрибут – пойменована характеристика сутності. Його найменування повинне бути унікальним для конкретного типу сутності, але може бути однаковим для різного типу сутностей. Атрибути використовуються для визначення того, яка інформація повинна бути зібрана про сутність. Абсолютне розходження між типами сутностей і атрибутами відсутне. Атрибут є таким тільки у зв'язку з типом сутності. В іншому контексті атрибут може виступати як самостійна сутність.

Ключ – мінімальний набір атрибутів, за значеннями яких можна однозначно знайти необхідний екземпляр сутності. Мінімальність означає, що виключення з набору будь-якого атрибута не дозволяє ідентифікувати сутність за тими що залишилися.

Зв'язок – асоціювання двох або більше сутностей. Якби призначенням бази даних було тільки зберігання окремих, не зв'язаних між собою даних, то її структура могла б бути дуже простою. Однак одне з основних вимог до організації бази даних - це забезпечення можливості відшукання одних сутностей за значеннями інших, для чого необхідно встановити між ними певні зв'язки. А тому, що в реальних базах даних нерідко утримуються сотні або навіть тисяча сутностей, те теоретично між ними може бути встановлені більше мільйона зв'язків. Наявність такої безлічі зв'язків і визначає складність інфологічних моделей.

Існують три основні класи сутностей: стрижневий, асоціативний й характеристичний, а також підклас асоціативних сутностей – позначення.

Стрижнева сутність (стрижень) – це незалежна сутність.

Асоціативна сутність (асоціація) – це зв'язок виду "багато-до-багатьох" ("- до-багатьох" і т.д.) між двома або більше сутностями або екземплярами сутності . Асоціації розглядаються як повноправні сутності:

- вони можуть брати участь в інших асоціаціях і позначеннях точно так само, як стрижневі сутності;

- можуть мати властивості, тобто мати не тільки набір ключових атрибутів, необхідних для вказівки зв'язків, але й будь-яке число інших атрибутів, що характеризують зв'язок.

Характеристична сутність (характеристика) – це зв'язок виду " багато -до-однієї" або "одна-до-однієї" між двома сутностями. Єдина мета характеристики в рамках розглянутої предметної області складається в описі або уточненні деякої іншої сутності. Необхідність у них виникає у зв'язку з тим, що сутності реального миру мають іноді багатозначні властивості [22].

4.4.2 Первинні й зовнішні ключі

Нагадаємо, що ключ або можливий ключ – це мінімальний набір атрибутів, за значеннями яких можна однозначно знайти необхідний екземпляр сутності. Мінімальність означає, що виключення з набору будь-якого атрибута не дозволяє ідентифікувати сутність за тими, що залишилися. Кожна сутність володіє хоча б одним можливим ключем. Один з них приймається за первинний ключ. При виборі первинного ключа варто віддавати перевагу нескладним ключам або ключам, складеним з мінімального числа атрибутів. Недоцільно також використовувати ключі з довгими текстовими значеннями. Також необхідно забезпечити унікальність первинного ключа.

Зовнішні ключі:

- якщо сутність У зв'язує сутності А та В, то вона повинна включати зовнішні ключі, що відповідають первинним ключам сутностей А та В.
- якщо сутність У позначає сутність А, то вона повинна включати зовнішній ключ, що відповідає первинному ключу сутності А [23].

У будь-якій таблиці, як правило, хоча б одне поле є **ключовим**. Завдяки ключовим полям будь-який запит до бази даних може бути оформлений у вигляді операцій з таблицями та їхніми полями (атрибутами), але не з їхніми рядками. Крім того, ключові поля використовуються для підтримки цілісності реляційної бази даних. Таблиця може мати тільки один унікальний ідентифікатор - **первинний ключ (**primary key**)** і трохи вторинних (зовнішніх) ключів (foreign key), що посилаються на первинні ключі інших таблиць. При цьому зовнішні ключові поля таблиці, повинні мати одне із значень, певне для первинних ключів.

Назви полів у допоміжній таблиці можуть і не збігатися з назвами первинних ключів, але тип даних і розмір цих полів повинні збігатися обов'язково. Забезпечення цілісності даних досягається при встановленні

двох перемикачів у меню СУБД Access – «Каскадне відновлення зв'язаних полів» і «Каскадне видалення зв'язаних полів» [11].

4.5 Нормалізація відносин

Ті самі дані можуть групуватися в таблиці (відносини) різними способами, тобто можлива організація різних наборів відносин взаємозалежних інформаційних об'єктів. Угруповання атрибутів у відносинах повинне бути раціональним, тобто мінімізуючим дублювання даних і процедури, що спрощує, їхню обробку й відновлення.

Певний набір відносин має кращі властивості при включенні, модифікації, видаленні даних, чим всі інші можливі набори відносин, якщо він відповідає вимогам нормалізації відносин.

Нормалізація відносин — формальний апарат обмежень на формування відносин (таблиць), що дозволяє усунути дублювання, забезпечує несуперечність збережених у базі даних, зменшує працевитрати на ведення (уведення, коректування) бази даних.

Виділені три **нормальні форми відносин** і запропонований механізм, що дозволяє будь-яке відношення перетворити до третьої (самої зробленої) нормальній формі.

Перша нормальна форма. Відношення називається нормалізованим або наведеним до першої нормальної форми, якщо всі його атрибути прості (неподільні). Перетворення відносини до першої нормальної форми може привести до збільшення кількості реквізитів (полів) відносини й зміні ключа.

Друга нормальна форма. Щоб розглянути питання приведення відносин до другої нормальної форми, необхідно дати пояснення до таких понять, як функціональна залежність і повна функціональна залежність. Описові реквізити інформаційного об'єкта логічно пов'язані із загальним для них ключем, цей зв'язок носить характер функціональної залежності реквізитів.

Функціональна залежність реквізитів — залежність, при якій екземпляру інформаційного об'єкта (певному значенню ключового реквізиту) відповідає тільки одне значення описового реквізиту.

Таке визначення функціональної залежності дозволяє при аналізі всіх взаємозв'язків реквізитів предметної області виділити самостійні інформаційні об'єкти. У випадку складного ключа вводиться поняття **функціонально повної** залежності.

Функціонально повна залежність не ключових атрибутів полягає в тім, що кожний не ключовий атрибут функціонально залежить від ключа, але не перебуває у функціональній залежності ні від якої частини складного ключа.

Відношення буде перебувати в другій нормальній формі, якщо воно перебуває в першій нормальній формі, і кожний не ключовий атрибут функціонально повно залежить від складного ключа.

Третя нормальна форма. Поняття третьої нормальної форми ґрунтується на понятті нетранзитивної залежності.

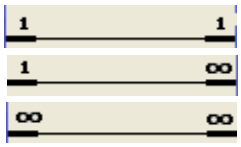
Транзитивна залежність спостерігається в тому випадку, якщо один з двох описових реквізитів залежить від ключа, а інший описовий реквізит залежить від першого описового реквізиту.

Відношення буде перебувати в третій нормальній формі, якщо воно перебуває в другій нормальній формі, і кожний неключовий атрибут нетранзитивно залежить від первинного ключа.

Для усунення транзитивної залежності описових реквізитів необхідно провести "розщеплення" вихідного інформаційного об'єкта. У результаті розщеплення частина реквізитів віддаляється з вихідного інформаційного об'єкта й включається до складу інших (можливо, знову створених) інформаційних об'єктів.

Типи зв'язків. Всі інформаційні об'єкти предметної області зв'язані між собою. Розрізняються зв'язки декількох типів, для яких уведені наступні позначення:

- один до одного (1:1);
- один до багатьох (1 : М);
- багато до багатьох (М : М).



Зв'язок **один до одного** (1:1) допускає, що в кожний момент часу одному екземпляру інформаційного об'єкта А відповідає не більше одного екземпляра інформаційного об'єкта В й навпаки.

При зв'язку **один до багатьох** (1:М) одного екземпляра інформаційного об'єкта А відповідає 0, 1 або більше екземплярів об'єкта В, але кожний екземпляр об'єкта У зв'язаний не більш ніж з 1 екземпляром об'єкта А.

Зв'язок **багато до багатьох** (М:М) допускає, що в кожний момент часу одному екземпляру інформаційного об'єкта А відповідає 0, 1 або більше екземплярів об'єкта В й навпаки [25].

Питання для самоперевірки

1. Два підходи до моделювання предметної області.
2. Технологія аналізу предметної області.
3. Поняття сутностей, їх атрибутів, зв’язків між ними.
4. Поняття первинних та зовнішніх ключів.
5. Поняття нормальні форми.
6. Основні рівні організації даних.
7. Головний об’єкт інформаційної діяльності в Гідромеліорації.
8. Основні підсистеми природно – ресурсної основи.
9. Нормативно – регламентуюча база.
10. Основні матеріали територіально – галузевої бази.
11. Концептуальне подання.
12. Типи сутностей.

5 ЛОГІЧНА ТА ФІЗИЧНА МОДЕЛІ БАЗ ДАНИХ

5.1 Основні поняття

Логічна модель даних відображує концептуальну модель у вигляді структур, що використовуються системою управління базами, тобто це формалізоване подання концептуальної моделі.

Реляційна модель — один із видів логічної моделі, в якій об'єкти взаємозв'язку представляють за допомогою таблиць. При цьому взаємозв'язки також розглядаються як об'єкти. Кожна таблиця відображує один об'єкт і складається з рядків та стовпців [21].

Фізична модель даних визначає порядок розміщення даних на зовнішніх носіях ЕОМ, формат їхнього зберігання та конкретні способи доступу до них; це цифрове подання даних, що зберігаються у фізичних пристроях комп'ютерів [21].

Моделі даних ґрунтуються на двох основних видах диференціації або групування інформації: тематичному і системному.

Тематична організація інформації (тематичні дані) — групування даних за характеристиками основних компонентів природно-агромеліоративних геосистем або предметної області, що здійснюються за принципом змістовної організації інформації [5]. Предметна область — це частина реального світу, в межах якої визначається набір даних і методів, маніпулювання ними для вирішення конкретних завдань або виконання досліджень.

Експертні системи забезпечують регламентацію правил виконання того чи іншого завдання, у тому числі з оптимізації та нормування, їхньої формалізації і потребують розробки комплексу програм, що дають змогу приймати рішення за допомогою комп'ютера. Для їхнього функціонування необхідна наявність трьох основних компонентів — фактів, правил та структур управління.

Фактичні знання повідомляються експертній системі експертом у процесі діалогу і відображують погляди людини щодо інтерпретації даних на момент роботи. Процедурні знання тісно пов'язані з фактичним накопиченим досвідом, на основі якого відпрацьовувались правила, що регламентують поведінку системи. Знання, на основі яких здійснюють управління, дають змогу підбирати найкращу стратегію у роботі системи. Аналітична частина ЕС у вигляді системи SQL - запитань являє собою алгоритми так званих продукцій, побудованих на виразах «якщо — то», логічних описів у відповідних модулях [8].

Системи управління базами даних призначені для зберігання й управління всіма типами даних, включаючи географічні (просторові) дані. СУБД оптимізовані для подібних завдань, тому в багатьох ГІС вбудована підтримка СУБД. Ці системи не мають подібних з ГІС інструментів для аналізу й візуалізації.

До апаратного забезпечення системи відносяться:

• накопичувачі для зберігання інформації (звичайні диски з переміщуваними головками) разом з приєднаними пристроями введення-виводу, контролерами пристроїв, каналами введення-виводу й т. п.;

• процесор (або процесори) разом з основною пам'яттю, що використовується для підтримки роботи програмного забезпечення системи.

Між властиво фізичною базою даних (тобто даними, які в дійсності збережені) і користувачами системи розташовується рівень програмного забезпечення— диспетчер бази даних (database manager) або, що більш звично, система управління базами даних, СУБД (database management system (DBMS)). Всі запити користувачів на доступ до бази даних обробляються СУБД; можливості додавання файлів (або таблиць), вибірки й відновлення даних у цих файлах або таблицях також забезпечує СУБД.

Основна функція, виконувана СУБД, — це надання користувачеві бази даних можливості працювати з нею, не вникаючи в деталі на рівні апаратного забезпечення (користувач більше відсторонений від цих деталей, чим прикладний програміст, що використовує середовище програмування). Іншими словами, СУБД дозволяє користувачеві розглядати базу даних, як об'єкт більш високого рівня в порівнянні з апаратним забезпеченням, а також підтримує користувальницькі операції, що виражаються в термінах високого рівня (наприклад, операції, які можна виконувати за допомогою мови SQL). СУБД - найбільш важливий, але не єдиний програмний компонент системи. Серед інших - утиліти, засоби розробки додатків, засоби проектування, генератори звітів та ін. [10].

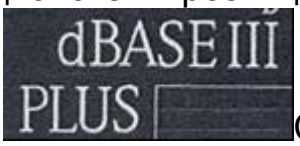
5.2. Бази даних і системи управління базами даних

Для маніпулювання інформацією (введенням, пошуком і т.п.) використовуються спеціальні пакети програм, що мають назву **системи управління базами даних** (СУБД). Цей вид програмного забезпечення в останні роки дуже швидко вдосконалюється. З однієї сторони СУБД все ширше використовуються для маніпулювання новими типами інформації (мультімедіа, геоінформаційні системи і т.п.) З іншого боку - створені нові технології (архітектура " клієнт-сервер", розподілені бази даних, гіпертекст і т.п.), які дозволяють забезпечити доступ до інформації широкому колу користувачів у рамках мережі Internet, відкриваючи тим самим принципово нові можливості для вивчення навколишнього середовища.

В останні роки, завдяки розвитку технологій мультимедіа, за допомогою комп'ютерів стало можливим обробляти практично будь-які типи інформації про навколишнє середовище - замальовки, звуки, відео, і термін «інформація» став часто використовуватися як синонім терміна «дані». Системи управління базами даних (СУБД) з'явилися раніше, ніж персональні комп'ютери. Специфіка великих ЕОМ, на яких відбувалося їхнє становлення в сімдесяті роки, багато в чому визначило особливості тих СУБД - вони дозволяли кваліфікованому програмістові робити дуже багато,

починаючи від обробки транзакцій і закінчуючи перенесенням даних і програм в інші операційні системи й на інші ЕОМ (платформи). Але ці СУБД (Oracle, INGRES і т.п.) не стали стандартом для персональних комп'ютерів, тому що пред'являли занадто високі вимоги до характеристик використовуваної обчислювальної техніки й до кваліфікації користувачів.

У вісімдесяті роки було розроблено велике число СУБД спеціально для персональних комп'ютерів. У нашій країні найбільше поширення одержали такі СУБД, як FoxBASE, dBASE III plus, R:Base, Paradox, а наприкінці вісімдесятих років придбав популярність пакет Clipper. Слід зазначити, що FoxBASE, dBASE III plus і Clipper використовували ті самі принципи організації інформації, й були сумісні на рівні файлів баз даних, тому іноді всі ці системи розглядали як модифікації dBASE III plus.



Система програмування **dBASE III plus** була розроблена фірмою Ashton-Tate на основі своїх більш ранніх СУБД - dBASE II і dBASE III. В dBASE III plus основна увага була приділена вдосконаленню користувальницького інтерфейсу (режим ASSIST), що істотно спростило процедуру створення й модифікації баз даних, сортування й індексацію записів. Створення й використання досить складних структур баз даних було можливо безпосередньо з режиму ASSIST без складання прикладних програм мовою dBASE, що робило цю СУБД доступною для широкого кола користувачів. Це забезпечило величезну популярність dBASE III plus, і наприкінці вісімдесятих років ця СУБД була фактичним стандартом для реляційних баз даних, незважаючи на деякі недоліки, властиві цій системі.

Однак до початку 90-х років ситуація змінилася. Найбільш популярними СУБД для ПК сталі FoxBASE (FoxPro) і Paradox. СУБД FoxPro була розроблена фірмою Fox Software Inc. у першу чергу для створення додатків для користувачів. Ця СУБД мала дуже потужні програмні засоби й дозволяла легко писати прикладні програми, хоча, бути може, і не мала такого дружнього інтерфейсу, як dBASE III plus. СУБД Paradox була розроблена фірмою Borland International і також була більше орієнтована на створення додатків на основі вбудованої повнофункціональної мови програмування PAL. У цієї СУБД використовувався новий метод організації інформації, заснований на метафорі таблиці, що дозволяло з однієї сторони легко реалізувати систему запитів за зразком (Query by Example), а з іншого боку - забезпечити дуже високу швидкодію при пошуку інформації.

Одним з недоліків СУБД FoxPro, dBASE III plus, Paradox була неможливість створення з їхньою допомогою файлів .EXE, що автономно працюють під управлінням DOS. Саме тому широке поширення (у всякому разі в нашій країні) придбав пакет Clipper фірми Nantucket, що з самого

початку призначався для компіляції прикладних програм. Clipper працював з файлами .DBF, забезпечуючи досить високу швидкодію. У той час це була відкрита система, що дозволяла розширювати можливості мови за рахунок додатків, написаних на інших мовах програмування - Assembler'е й С.

В 1991 році, коли фірма Ashton-Tate була придбана фірмою Borland International, фірма Fox Software Inc. - MicroSoft, а фірма Nantucket - Computer Associates, формально почався новий етап у розвитку СУБД для ПК, хоча основні ідеї цього етапу пророблялися значно раніше. Можна виділити три основні особливості нового етапу: розподілені бази даних, графічний користувальницький інтерфейс і архітектура " клієнт-сервер". В 1993 році перші СУБД цього нового покоління - Microsoft Access, dBase IV і т.п. з'явилися на ринку й одержали широке поширення серед користувачів персональних комп'ютерів.

У цей час фактичним стандартом систем управління базами даних для персональних комп'ютерів є СУБД **Microsoft Access**. Пакет Microsoft Access for Windows 95 є потужним засобом управління базами даних, що підтримує реляційну модель даних і дозволяє створювати складні додатки на особливому діалекті Visual BASIC (VBA). Microsoft Access можна застосовувати для пошуку й обробки всіляких даних, а також для підготовки звітних документів. Користувальницький інтерфейс досить простий і надає користувачеві зручні можливості для маніпулювання базами даних, так що освоєння пакета не викликає складностей.

У зв'язку з бурхливим розвитком мережі Internet, яка є гігантською розполільною базою даних, зріс інтерес до таких СУБД як **Oracle**. У цей час ця система управління базами даних установлена на багатьох серверах Мережі.

Система управління базами даних Oracle фірми Oracle є одним з лідерів ринку багатоплатформених СУБД. Вона може працювати на більш ніж двохстах типах ЕОМ, включаючи ПК типу IBM PC і Apple Macintosh. У програмне забезпечення цієї СУБД входить одна з найбільш повних реалізацій мови структурованих запитів SQL, а також генератори меню, звітів та інших екранних форм. Крім того, програмне забезпечення дозволяє на підставі інформації, що зберігається в СУБД, будувати більше 50 типів графіків і діаграм. Oracle містить дуже надійну систему захисту даних, їхньої цілісності й несуперечності [29].

5.3 Характеристика ACCESS

Access — це, насамперед, система управління базами даних (СУБД). Як і інші продукти цієї категорії, вона призначена для зберігання й пошуку даних, подання інформації в зручному виді й автоматизації часто

повторюваних операцій (облік, планування й т.п.). За допомогою Access можна розробляти прості й зручні форми введення даних, а також здійснювати обробку даних і видачу складних звітів.

Access - потужний додаток Windows; уперше продуктивність СУБД органічно сполучається з тими зручностями, які є в розпорядженні користувачів Microsoft Windows. Оскільки обоє ці продукти- дітища компанії Microsoft, вони прекрасно взаємодіють між собою. Система Access (рис. 5.1)працює під управлінням Windows 2000 або Windows NT, так що при роботі з нею користувачеві доступні всі переваги Windows. Можна вирізати, копіювати й вставляти дані з будь-якого додатка Windows в Access і навпаки; можна створити проект форми в Access і вставити його в конструктор форм.

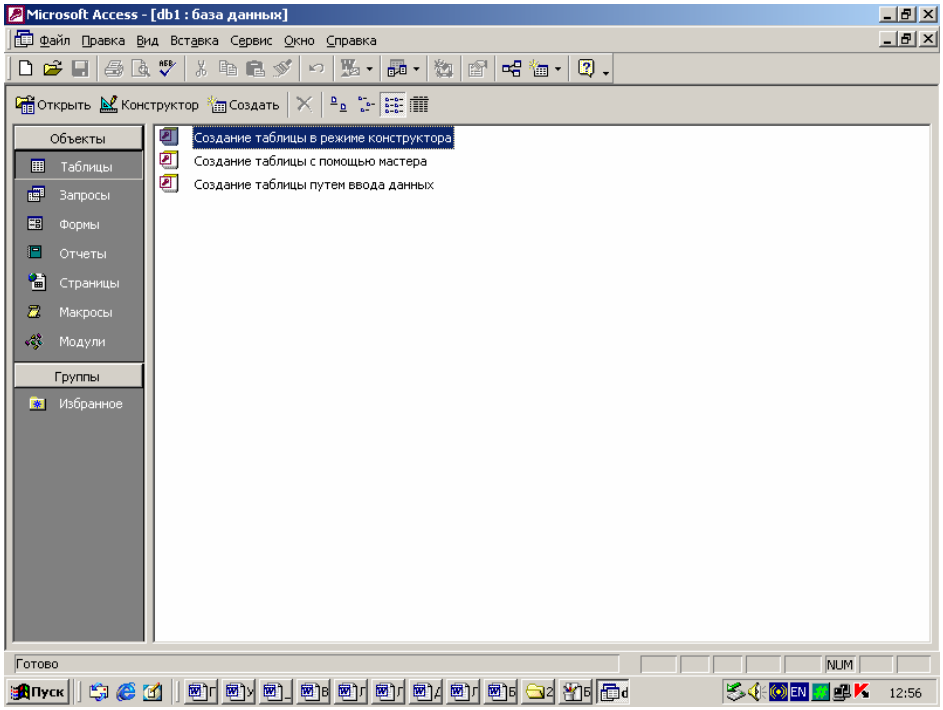


Рис. 5.1. Вікно СУБД Access

За допомогою об'єктів OLE (рис. 5.2) (Object Linking and Embedding - зв'язування й впровадження об'єктів) в Windows 2000 і компонентах Microsoft Office 2000 (Excel, Word, PowerPoint і Outlook) можна перетворити Access у справжнє операційне середовище баз даних. За допомогою нових розширень для Internet можна створювати форми, які будуть прямо взаємодіяти з даними з World Wide Web, і транслювати їх у подання мовою HTML, що забезпечує роботу з такими продуктами, як Internet Explorer і Netscape Navigator.

При всьому цьому Access — не просто СУБД. Як реляційна СУБД Access забезпечує доступ до всіх типів даних і дозволяє використовувати одночасно кілька таблиць бази даних. При цьому можна істотно спростити структуру даних, полегшуючи тим самим виконання поставлених завдань.

Таблицю Access можна зв'язати з даними, що зберігаються на великих ЕОМ або на сервері. З іншого боку, можна використовувати таблиці, створені в середовищі Paradox або dBASE. Отримані результати можна швидко й легко зв'язати й об'єднати з даними з електронних таблиць Excel. Працюючи в середовищі Microsoft Office 2000, користувач одержує у своє розпорядження повністю сумісні між собою Access і Word, Excel і PowerPoint.

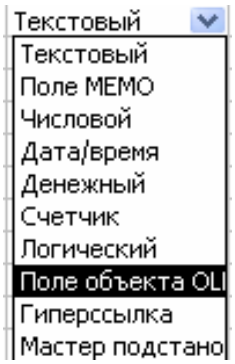


Рис. 5.2. Поле об’єкту OLE

Система Access - це набір інструментів кінцевого користувача для управління базами даних. У її состав входять конструктори таблиць, форм, запитів і звітів. Цю систему можна розглядати і як середовище розробки додатків. Використовуючи макроси або модулі для автоматизації рішення завдань, можна створювати орієнтовані на користувача додатки такіж потужні, як і додатки, написані безпосередньо на мовах програмування. При цьому вони будуть включати кнопки, меню й діалогові вікна. Програмуючи мовою VBA, можна створювати такі потужні програми, як сама система Access.

Потужність і доступність Access роблять цю систему кращою СУБД із представлених сьогодні на ринку.

Access це:

Справжня реляційна модель баз даних. В Access повною мірою реалізоване управління реляційними базами даних. Система підтримує первинні та зовнішні ключі й забезпечує цілісність даних на рівні ядра (що запобігає несумісним операціям відновлення або видалення даних). Крім того, таблиці в Access забезпечені засобами перевірки допустимості даних, що запобігають некоректному введенню поза залежністю від того, як він здійснюється, а кожне поле таблиці має свій формат і стандартні описи, що істотно полегшує введення даних. Access підтримує всі необхідні типи полів, у тому числі текстовий, числовий, лічильник, грошовий, дата/час, MEMO, логічний, гіперпосилання й поля об'єктів OLE. Якщо в процесі спеціальної обробки в полях не виявляється ніяких значень, система забезпечує повну підтримку порожніх значень.

Реляційна обробка даних в Access за рахунок гнучкої архітектури системи здатна задовольнити будь-які потреби. При цьому Access може

використовуватися як автономна СУБД у режимі файл-сервера або клієнтського компонента таких продуктів, як SQL Server. Крім того, Access підтримує протокол ODBC (Open Database Connectivity), що дозволяє підключатися до баз даних безлічі різних форматів, таких як SQL Server, Oracle, Sybase і навіть DB/2 для великих ЕОМ фірми IBM. Система Access підтримує обробку транзакцій з гарантією їхньої цілісності. Крім того, передбачений захист на рівні користувача, що дозволяє контролювати доступ до даних окремих користувачів і цілих груп.

Прості у використанні майстри й конструктори. Майстер (Wizard) може перетворити години роботи в лічені хвилини. Майстри задають навідні запитання щодо вмісту, стилю й формату створюваного об'єкта; потім вони автоматично будують потрібний об'єкт. У складі Access біля ста майстрів, що допомагають конструювати бази даних, додатки, таблиці (рис.5.3), форми, звіти, діаграми, поштові наклейки, елементи управління й властивості. Допускається навіть настроювання майстрів для рішення різних завдань.

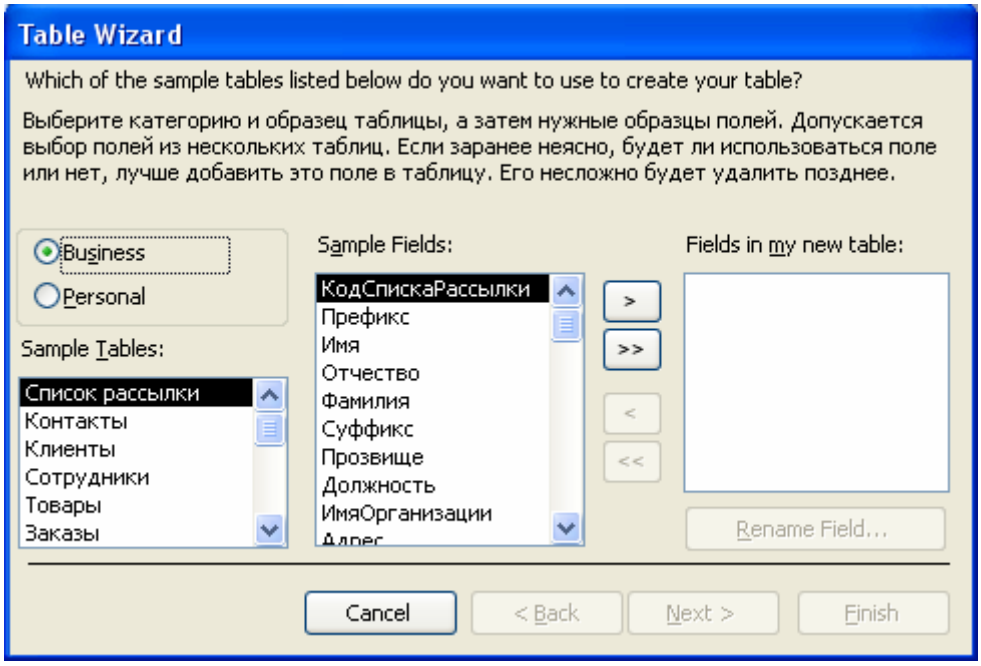
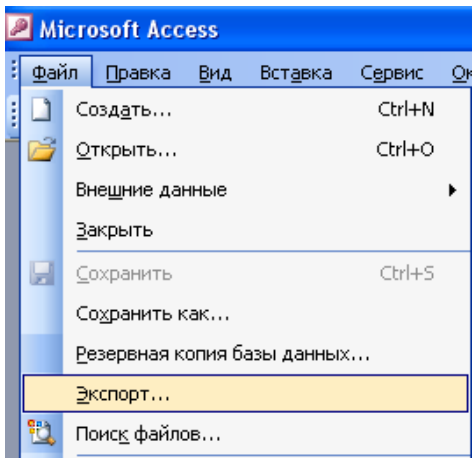
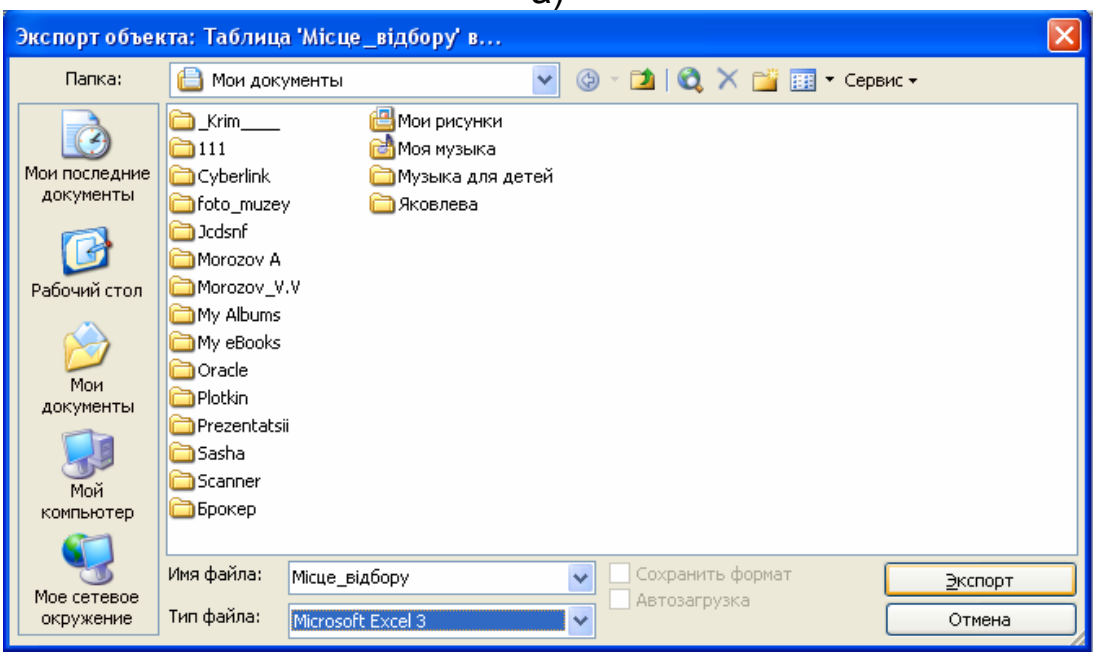


Рис. 5.3. Приклад діалогового вікна Майстра таблиць

Імпортування, експортування й зв'язування зовнішніх файлів. Access дозволяє імпортувати й експортувати (рис. 5.4) файли багатьох відомих форматів, включаючи dBASE, FoxPro, Excel, SQL Server, Oracle, Btrieve, багато текстових форматів ASCII (у тому числі з фіксованою довжиною рядка або заданим обмежником), а також дані у форматі HTML. У результаті імпортування створюється таблиця Access; у результаті експортування таблиці Access створюється файл у заданому форматі.



а)



б)

Рис. 5.4. Приклад експорту даних із Microsoft Access в Microsoft Excel

Зв'язування (раніше йменувалося приєднанням) означає, що можна використовувати зовнішні дані без створення таблиці Access. Можна встановлювати подібний зв'язок з даними dBASE, FoxPro, Excel, ASCII і SQL. Дуже потужна можливість - зв'язування таблиць Access з їхніми зовнішніми таблицями з наступним спільним використанням; це відноситься до таблиць Access, dBASE, FoxPro і SQL Server.

Форми й звіти WYSIWYG. Вікна конструкторів форм (рис. 5.5) і звітів (рис. 5.6) мають однаковий інтерфейс і надають користувачеві багато можливостей. Форма або звіт конструюється за принципом WYSIWYG (What You See Is What You Get - що бачиш, то й одержиш). Додаючи черговий елемент управління, користувач бачить, як при цьому змінюється створювана форма.

У форми й звіти можна включати написи, поля текстових даних,

перемикачі, прапорці, лінії й прямокутники, а також оформляти їх, виділяючи елементи кольором і тінню. Більше того, можна включати цілі малюнки, діаграми, підформи й підзвіти. При цьому всі параметри подання даних залишаються повністю підконтрольними користувачеві. Форми можуть займати багато сторінок, а у звітах може бути передбачене багато рівнів групування даних і підведення підсумків.

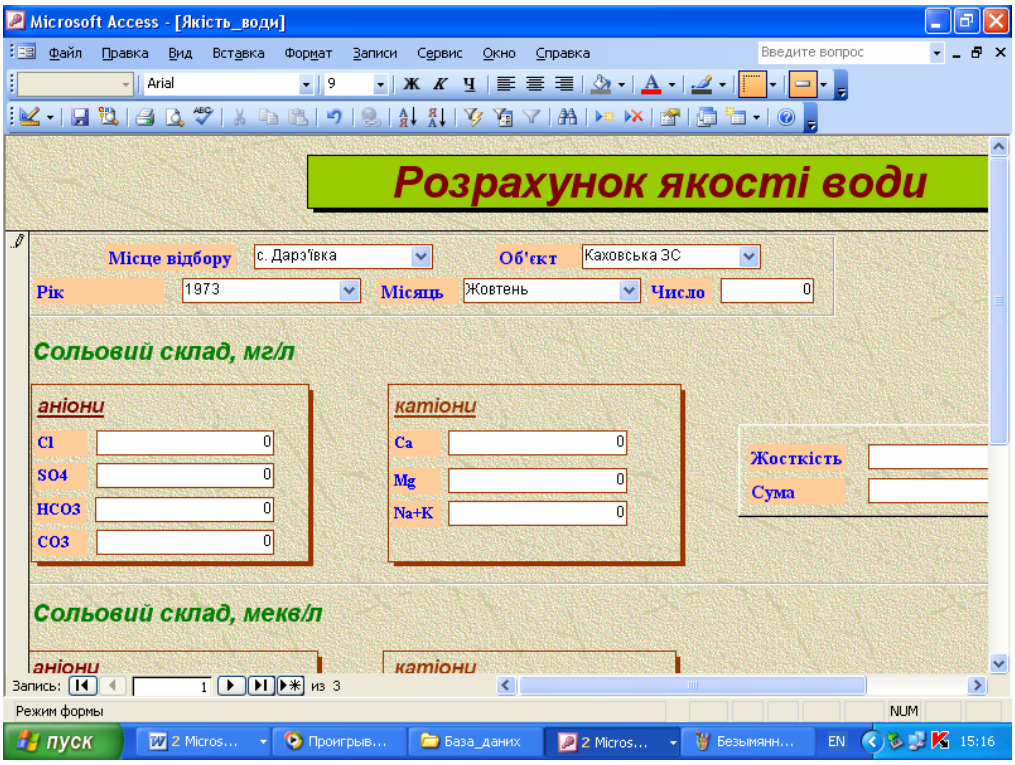


Рис. 5.5. Приклад форми

У форми й звіти можна включати написи, поля текстових даних, перемикачі, прапорці, лінії й прямокутники, а також оформляти їх, виділяючи елементи кольором і тінню. Більше того, можна включати цілі малюнки, діаграми, підформи й підзвіти. При цьому всі параметри подання даних залишаються повністю підконтрольними користувачеві. Форми можуть займати багато сторінок, а у звітах може бути передбачене багато рівнів групування даних і підведення підсумків.

Форми й звіти можна переглядати в режимі попереднього перегляду, забезпечуючи погляд "з висоти пташиного польоту" шляхом зміни масштабу. У режимі конструювання звіт можна переглядати з фіктивними даними, щоб не чекати обробки великого реального файлу.

Конструктор звітів - дуже потужний засіб, що допускає використання до десяти рівнів угруповання й сортування. Завдяки йому існує можливість створення звітів, що демонструють процентні й підсумкові показники, одержати які можна лише за два проходи. Допускається створення багатьох типів звітів, які включають поштові наклейки й списки розсилання пошти.

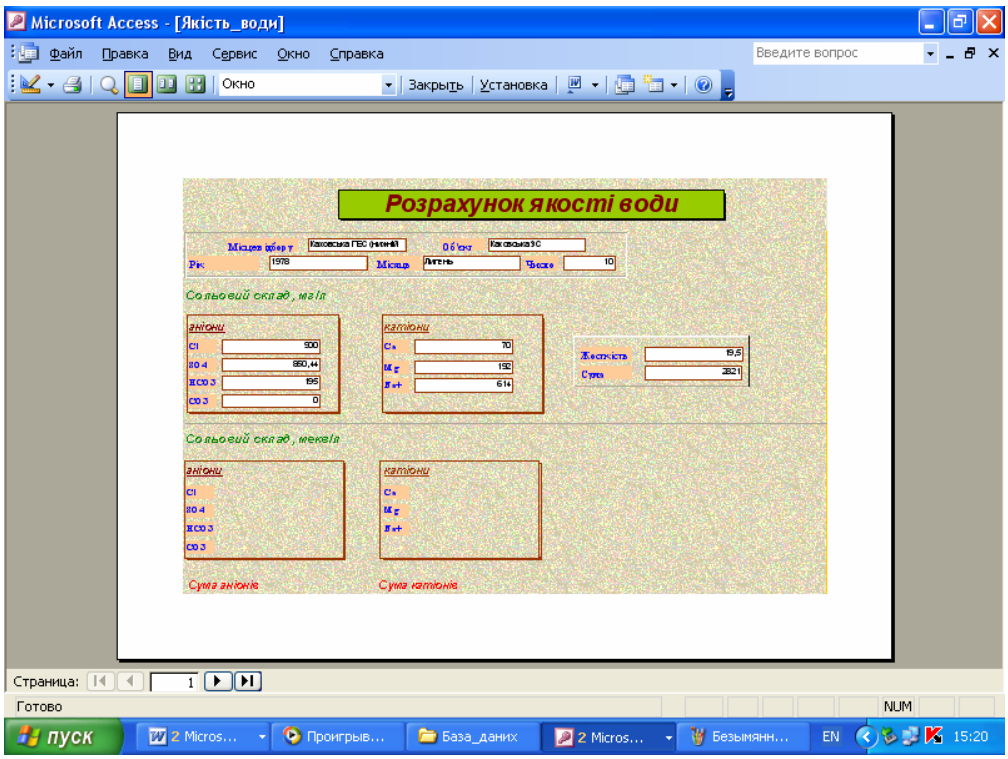


Рис. 5.6. Приклад звіту

Багатотабличні запити й відносини. Одна з самих потужних можливостей Access одночасно є й найбільш важливою. Відносини дозволяють зв'язати таблиці графічно (рис. 5.7). Можна навіть зв'язувати таблиці, що представляють файли різних типів (наприклад, таблицю Access і таблицю dBASE). Після подібного зв'язування таблиці виступають вже як одне ціле, і тепер можна будувати запити стосовно до будь-яких даних у них. Можна вибирати конкретні поля, визначати порядок сортування, вводити критерії відбору потрібних записів. Можна відображати результати виконання запиту у вигляді таблиці, форми або

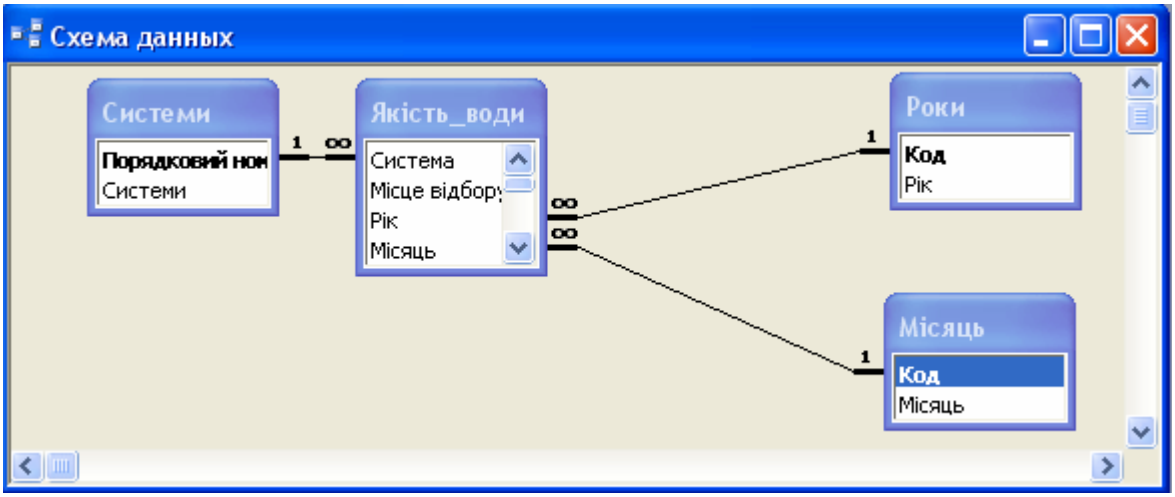


Рис. 5.7. Схематичний вигляд відносин між таблицями

звіту. Від користувача не потрібно попередньої установки зв'язків: замість цього досить увійти в конструктор запитів (наприклад, коли потрібно побудувати певний звіт).

Запити (рис. 5.8) застосовують і в інших випадках. Можна створювати запити, які забезпечують обчислення підсумків, відображення згрупованих і побудова нових таблиць. Запит можна використовувати навіть для відновлення даних у таблицях, видалення записів і додавання однієї таблиці до іншої.

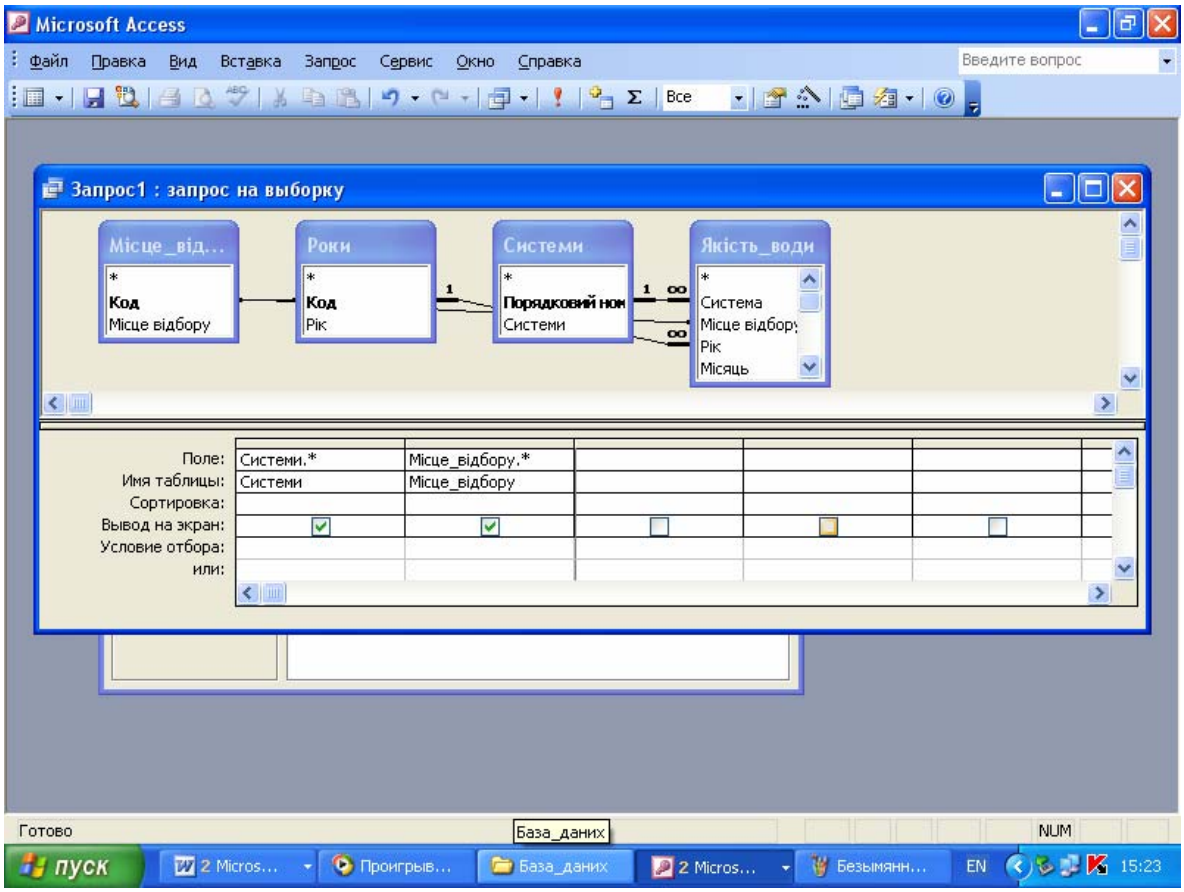


Рис. 5.8. Приклад створення запиту на вибірку

Графіки й діаграми. В Access використовується той же самий графічний додаток (рис. 5.9), що й в Microsoft Word, Excel, PowerPoint і Project. Він дозволяє створювати сотні типів графіків і діаграм, набудовуючи їх, виходячи з конкретних потреб. Можна створювати гістограми, лінійчаті, кругові, поверхневі й інші діаграми, причому як двох-, так і тривимірні. Їх можна довільно супроводжувати текстом, оформляти різними кольорами й візерунками. Значення можуть відображатися в стовпцях або секторах кругових діаграм. Можна розвертати зображення діаграм так, щоб вони відтворювалися під будь-яким зручним кутом зору. Все це забезпечує програма Access Graph.

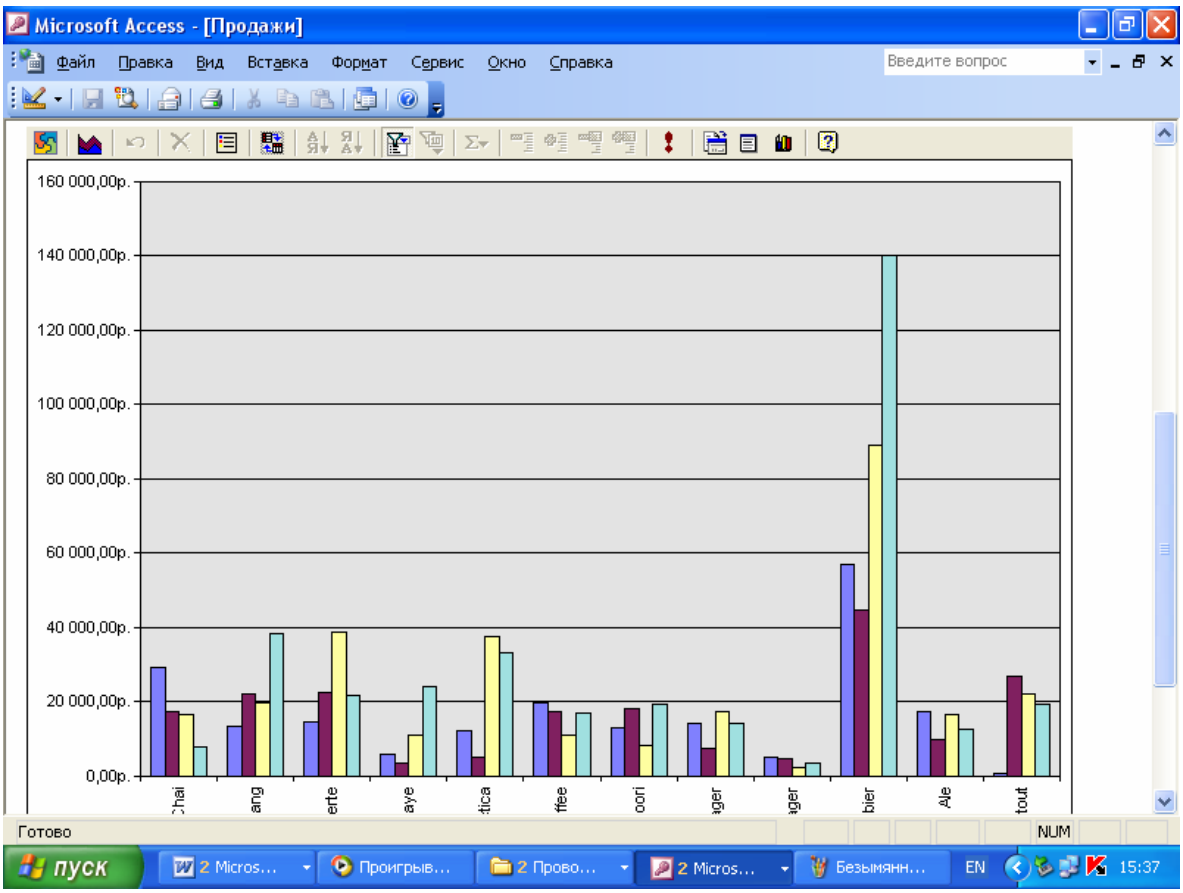


Рис. 5.9. Приклад діаграми в Access

Можливості DDE і OLE. За допомогою DDE (Dynamic Data Exchange - динамічний обмін даними) і OLE (Object Linking and Embedding - зв'язування й впровадження об'єктів) у форми й звіти Access можна додавати всілякі нові об'єкти. Такими об'єктами можуть бути звук, малюнки, діаграми й навіть відеокліпи. Можна впроваджувати об'єкти OLE (наприклад, растрові зображення) або документи текстових процесорів (Word або WordPerfect) або встановлювати зв'язки з електронними таблицями Excel. Зв'язуючи ці об'єкти зі своєю базою даних, користувач може створювати динамічні форми й звіти, а також використовувати ту саму інформацію в різних додатках Windows.

Доступ до Internet. В Access тепер передбачені всі можливості, що забезпечують зв'язок додатку з Internet/intranet. Одним клацанням кнопкою миші можна зберегти таблиці, запити, форми й звіти у форматі HTML. Відповідний майстер дозволяє навіть новачкові перенести коди HTML з об'єкта на Web-Сторінку, роблячи їх доступними для використання всім, хто подорожує по Internet. Гіперпосилання дозволяють одержувати доступ до даних, які розміщені на Web-Сторінці, прямо з форм Access (рис. 5.10).

Дехто вважає, що розміщення даних на Web-Сторінках повинне здійснюватися Web-Адміністраторами. Access з повною визначеністю доводить, що ця операція може бути з успіхом виконана будь-яким користувачем. А допоможе йому в цьому майстер розміщення на Web-

Сторінці, що забезпечує перетворення обраних об'єктів бази даних у формат HTML і перенесення їх вже в такому виді на Web-Сторінку. За допомогою цього майстра можна створити статичні або динамічні сторінки, перенести їх на Web-Сервер, створити свою початкову сторінку й навіть використовувати шаблони для одержання стандартного зовнішнього вигляду всіх HTML-Сторінок.

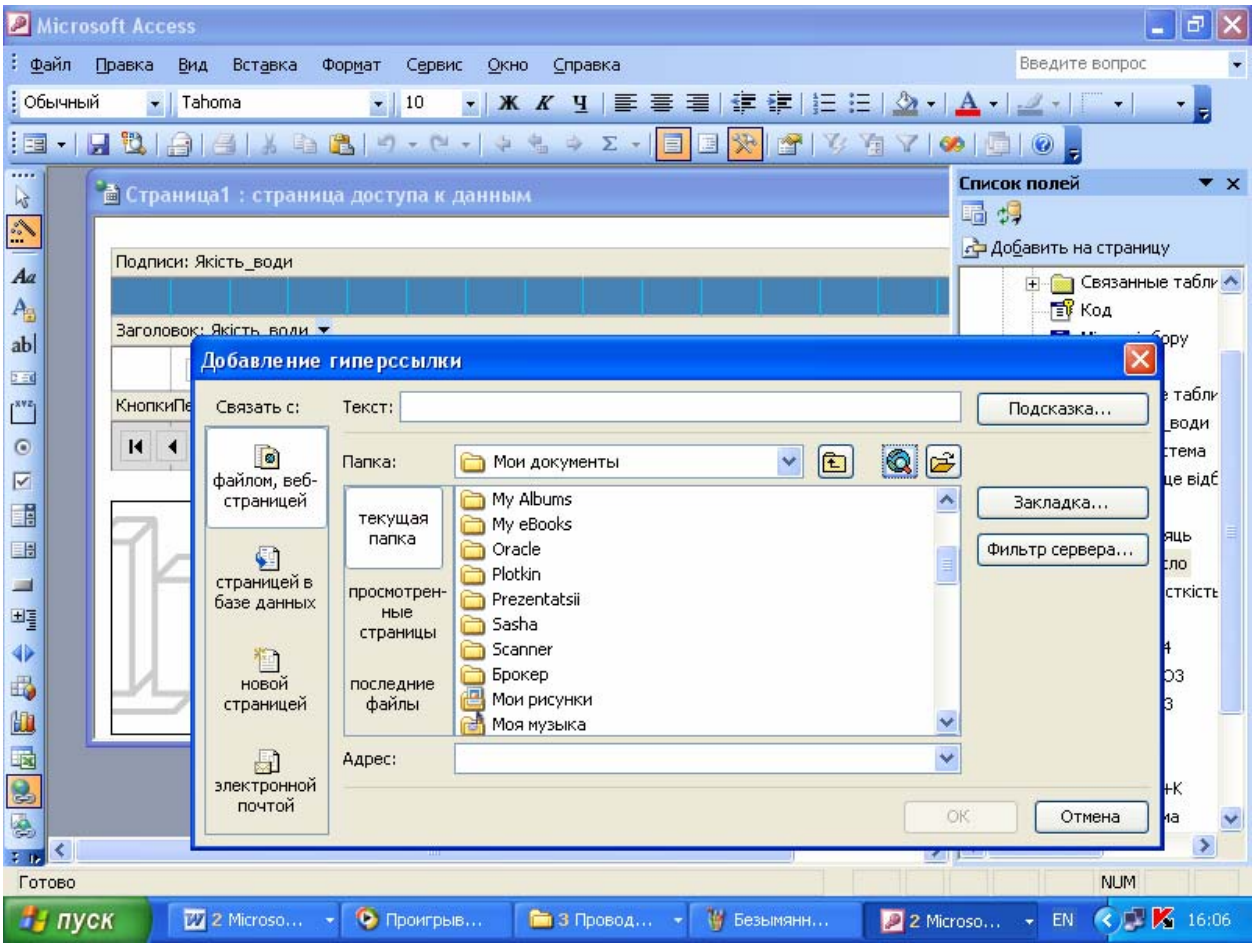
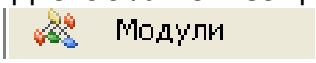


Рис. 5.10. Створення гіперпосилання з Access

Вбудовані функції. Access містить понад сто функцій (невеликих вбудованих програм, які в результаті виконання повертають значення), що виконують безліч різноманітних завдань. Є функції для маніпулювання базами даних, рядками, числами у форматі дати й часу, математичні, ділові й фінансові. Їх можна використовувати для створення виражень, що обчислюються, у формах, звітах і запитах.

Макроси: програмування без програмування. Для непрограмістів (або досвідчених користувачів, які просто не бажають програмувати) в Access передбачені макроси. Вони дозволяють автоматизувати виконання деяких завдань. Біля п'ятдесятьох макросів дають можливість маніпулювати даними, створювати меню й діалогові вікна, відкривати форми й звіти, словом, автоматизувати виконання

практично будь-якого завдання. За допомогою макросів можна вирішити порядка 90% всіх завдань обробки даних.



Модулі: Visual Basic for Applications програмування баз даних. Access - це серйозне середовище розробки додатків з повнофункціональною мовою програмування. Мова VBA (раніше відомий як Access Basic) реалізує об'єктно-орієнтований підхід до програмування й дозволяє програмістові робити практично все, що тільки можна собі представити. Це потужна мова структурного програмування. Вона є повністю розширюваною і підтримує процедури API у будь-яких динамічних бібліотеках (DLL) операційних систем Windows 95 і Windows NT.

Повнофункціональне середовище розробки підтримує безліч потужних сучасних можливостей: багатовіконий режим для редагування й налагодження, автоматичну перевірку синтаксису, контрольні крапки, покрокове виконання й навіть синтаксична довідка, що відображає на екрані варіанти команд, що вводяться [23, 28].

5.4 Адміністратор бази даних

Адміністратор даних (АД) — це людина, відповідальна за стратегію й політику прийняття рішень, пов'язаних з даними об'єкта, а адміністратор бази даних (АБД) — це людина, що забезпечує необхідну технічну підтримку виконання цих рішень. Таким чином, АБД відповідає за загальне управління системою на технічному рівні.

Функції АБД.

• Визначення концептуальної схеми.

Адміністратор даних визначає, які саме дані необхідно зберігати в базі даних, тобто визначає об'єкти, у яких зацікавлене підприємство, а також інформацію, яку необхідно записувати про ці об'єкти. Цей процес називають логічним (або концептуальним) проектуванням бази даних. Після того як вміст бази даних визначений адміністратором даних на абстрактному рівні, АБД створює відповідну концептуальну схему за допомогою концептуальної мови визначення даних. Об'єктна (відкомпільована) форма цієї схеми буде використовуватися СУБД при відповідях на запити, пов'язані з доступом. Вихідна (не відкомпільована) форма буде відігравати роль довідкового документа для користувачів системи.

• Визначення внутрішньої схеми.

Адміністратор бази даних повинен також вирішувати, як дані повинні бути представлені в збереженій базі даних. Цей процес називають фізичним проектуванням бази даних. Після завершення фізичного проектування АБД за допомогою внутрішньої мови визначення даних повинен створити відповідну структуру зберігання (тобто описати внутрішню схему). Крім цього, він повинен визначити відповідне

відображення між внутрішньою й концептуальною схемами. На практиці концептуальна й внутрішня мови визначення даних можуть містити в собі засоби визначення цього відображення, але ці дві функції повинні бути чітко розділені. Як і концептуальна схема, внутрішня схема й відповідне відображення будуть існувати у вихідній та об'єктній формах.

• Взаємодія з користувачами.

В обов'язки АБД також входить взаємодія з користувачами, забезпечення наявності необхідних їм даних і написання (або надання допомоги користувачеві в написанні) необхідних зовнішніх схем за допомогою прикладної зовнішньої мови визначення даних. Крім цього, необхідно також визначити відображення між кожною зовнішньою й концептуальною схемами. На практиці зовнішня мова визначення даних може включати засоби визначення цього відображення, але знову ж схема й відображення повинні бути чітко розділені. Зовнішня схема й відповідне відображення будуть існувати у вихідній та об'єктній формах.

Інші аспекти взаємодії з користувачами включають консультації щодо розробки додатків, забезпечення технічного утворення, допомога у виявленні й рішенні виникаючих проблем та інше пов'язане з системою професійне обслуговування.

• Визначення правил безпеці й цілісності.

Правила безпеки й цілісності розглядаються як частина концептуальної схеми. Концептуальна мова визначення даних повинна містити в собі засоби визначення цих правил.

• Визначення процедур резервного копіювання й відновлення.

Після того як об'єкт довірив свої дані системі баз даних, він став критично залежним від успішного функціонування системи. У випадку ушкодження якої-небудь частини бази даних внаслідок помилки людини, відмови устаткування або збоїв допоміжної операційної системи дуже важливо мати можливість відновити дані з мінімальною затримкою й з найменшим впливом на іншу частину системи. В ідеалі дані, які не були ушкоджені, зовсім не повинні зачіпатися. Адміністратор бази даних повинен визначити й реалізувати підходящу схему відновлення, що використовує, наприклад, періодичне вивантаження (або "дампинг") бази даних на пристрій резервного копіювання й процедури, при необхідності загружаючи базу даних з останнього дампа.

• Управління продуктивністю й реагування на вимоги, що змінюються.

АБД відповідає за таку організацію системи, при якій можна одержати найбільшу для всього об'єкту продуктивність, а також за переналагодження системи відповідно до вимог, що змінюються.. Наприклад, може виникнути необхідність у поетапній реорганізації збереженої бази даних, щоб бути впевненим, що рівень продуктивності залишається прийнятним. Як уже згадувалося, будь-які зміни на рівні фізичної пам'яті системи (внутрішньому рівні) повинні супроводжуватися відповідними змінами визначень відображення з концептуального рівня, тому концептуальна схема може залишатися незмінною [10].

Питання для самоперевірки

1.Три основні категорії засобів моделювання даних.
2. Системи управління базами даних.
3. Логічна модель баз даних.
4. Характеристики Access.
5. Поняття адміністратор бази даних.
6. Основна функція СУБД.
7. Основні версії СУБД.
8. Історія становлення СУБД.
9. Майстри Access.
10. Конструктори Access.
11. Формати файлів.
12. Форми та звіти Access.
13. Поняття макроси.
14. Різниця між АД та АБД.
15. Основні функції АБД.

6. ІНФОРМАЦІЙНА МОДЕЛЬ СУБД

6.1 Попереднє планування, підготовка даних, послідовність створення інформаційної моделі

При проектуванні системи обробки даних найбільш важлива організація даних. Допомогти зрозуміти організацію даних покликана інформаційна модель.

Процес створення інформаційної моделі починається з визначення концептуальних вимог ряду користувачів. Концептуальні вимоги можуть визначатися й для деяких завдань (додатків), які найближчим часом реалізовувати не планується. Це може трохи підвищити трудомісткість роботи, однак допоможе найбільш повно врахувати всі нюанси функціональності, необхідної для розроблювальної системи, і знизить імовірність переробки надалі. Вимоги окремих користувачів повинні бути представлені в єдиному «узагальненому поданні». Останнє називають концептуальною моделлю.

Об'єкт - це абстракція безлічі предметів реального миру, що володіють однаковими характеристиками й законами поводження. Об'єкт являє собою типовий невизначений екземпляр такої безлічі.

Об'єкти поєднуються в класи за загальними характеристиками.

Клас - це безліч предметів реального миру, зв'язаних спільністю структури й поводженням.

Концептуальна модель представляє об'єкти та їхні взаємозв'язки без указування способів їхнього фізичного зберігання. Таким чином, концептуальна модель є, по суті, моделлю предметної області. При проектуванні концептуальної моделі всі зусилля розроблювача повинні бути спрямовані в основному на структуризацію даних і виявлення взаємозв'язків між ними без розгляду особливостей реалізації й питань ефективності обробки. Проектування концептуальної моделі засноване на аналізі розв'язуваних на цьому підприємстві завдань щодо обробки даних. Концептуальна модель включає описи об'єктів та їхніх взаємозв'язків, що представляють інтерес у розглянутій предметній області й даних, що виявляються в результаті аналізу. Маються на увазі дані, використовувані як у вже розроблених прикладних програмах, так і в тих, які тільки будуть реалізовані.

Нормалізація інформаційної моделі виконується в кілька етапів.

Дані, представлені у вигляді двовимірної таблиці, є першою нормальною формою реляційної моделі даних. Перший етап нормалізації полягає в створенні двовимірної таблиці, що містить всі необхідні властивості інформаційної моделі, і у виділенні ключових властивостей. Очевидно, що отримана досить значна таблиця буде містити дуже різнорідну інформацію. У цьому випадку будуть спостерігатися аномалії включення, відновлення й видалення даних, тому що при виконанні цих дій доведеться приділити увагу даним (уводити або піклуватися про те, щоб

вони не були стерті), які не мають до поточних дій ніякого відношення. Наприклад, може спостерігатися така парадоксальна ситуація.

Відношення задане в **другій нормальній формі**, якщо воно є відношенням у першій нормальній формі й кожна властивість, що не є первинною властивістю щодо цього, повністю залежить від будь-якого можливого ключа цього відношення.

Якщо всі можливі ключі відносини містять по одній властивості, то це відношення задане в другій нормальній формі, тому що в цьому випадку всі властивості, що не є первинними, повністю залежать від можливих ключів. Якщо ключі складаються більш ніж з однієї властивості, відношення, задане в першій нормальній формі, може не бути відношенням у другій нормальній формі. Приведення відносин до другої нормальної форми полягає в забезпеченні повної функціональної залежності всіх властивостей від ключа за рахунок розбивки таблиці на кроки, у яких всі наявні властивості будуть мати повну функціональну залежність від ключа цієї таблиці. У процесі приведення моделі до другої нормальної форми в основному виключаються аномалії дублювання даних.

Відношення задане в **третій нормальній формі**, якщо воно задано в другій нормальній формі й кожна властивість цього відношення, що не є первинним, не транзитивно залежить від кожного можливого ключа цього відношення.

Транзитивна залежність виявляє дублювання даних в одному відношенні. Якщо А, У и С - три властивості одного відношення й С залежить від В, а У від А, то говорять, що С транзитивно залежить від А. Перетворення в третю нормальну форму відбувається за рахунок поділу вихідного відношення на два [31].

Концептуальна модель переноситься потім у модель даних, сумісну з обраною СУБД. Можливо, що відбиті в концептуальній моделі взаємозв'язки між об'єктами виявляться згодом нереалізованими засобами обраної СУБД. Це зажадає зміни концептуальної моделі. Версія концептуальної моделі, що може бути забезпечена конкретною СУБД, називається **логічною моделлю.**

Логічна модель відбиває логічні зв'язки між елементами даних поза залежністю від їхнього змісту й середовища зберігання. Логічна модель даних може бути реляційною, ієрархічною або мережною. Користувачам виділяються підмножини цієї логічної моделі, що мають назву зовнішні моделі, що відбивають їхні подання про предметну область. Зовнішня модель відповідає поданням, які користувачі одержують на основі логічної моделі, у той час як концептуальні вимоги відбивають подання, які користувачі спочатку бажали мати і які лягли в основу розробки концептуальної моделі. Логічна модель відображається у фізичну пам'ять, таку, як диск, стрічка або який-небудь інший носій інформації.

Ієрархічна модель даних будується за принципом ієрархії типів об'єктів, тобто один тип об'єкта є головним, а інші, що перебувають на

нижчих рівнях ієрархії, — підлеглими. Між головним і підлеглим об'єктами встановлюється взаємозв'язок «один до багатьох». У той же час для кожного екземпляра головного об'єкта може бути кілька екземплярів підлеглих типів об'єктів. Взаємозв'язки між об'єктами нагадують взаємозв'язки в генеалогічному дереві за єдиним виключенням: для кожного породженого (підлеглого) типу об'єкта може бути тільки один вихідний (головний) тип об'єкта. Отже, отриману концептуальну модель, будемо вважати логіко-ієрархічною моделлю даних.

Фізична модель, що визначає розміщення даних, методи доступу й техніку індексування, називається внутрішньою моделлю системи.

Зовнішні моделі ніяк не пов'язані з типом фізичної пам'яті, у якій будуть зберігатися дані, і з методами доступу до цих даних. Це положення відбиває **перший рівень незалежності даних**. З іншого боку, якщо концептуальна модель здатна враховувати розширення вимог до системи в майбутньому, те внесені в неї зміни не повинні робити впливу на існуючі зовнішні моделі. Це — **другий рівень незалежності даних**. Побудова логічної моделі обумовлена вимогами використовуваної СУБД. Тому при заміні СУБД вона також може змінитися.

З погляду прикладного програмування незалежність даних визначається не технікою програмування, а його дисципліною, тобто для того щоб при будь-якій зміні системи уникнути перекомпіляції додатка, рекомендується не визначати константи (постійні значення даних) у програмі. Краще рішення складається в передачі програмі значень як параметрів.

Всі актуальні вимоги предметної області й адекватні їм «сховані» вимоги на стадії проектування повинні знайти своє відбиття в концептуальній моделі. Звичайно, не можна передбачити всі можливі варіанти використання й зміни бази даних. Але в більшості предметних областей такі основні дані, як об'єкти і їхні взаємозв'язки, відносно стабільні. Міняються тільки інформаційні вимоги, тобто способи використання даних для одержання інформації.

Ступінь незалежності даних визначається старанністю проектування бази даних. Всебічний аналіз об'єктів предметної області та їхніх взаємозв'язків мінімізує вплив зміни вимог до даних в одній програмі на інші програми. У цьому й складається всеосяжна незалежність даних.

Основне розходження між зазначеними вище трьома типами моделей даних (концептуальною, логічною й фізичною) складається в способах подання взаємозв'язків між об'єктами. При проектуванні БД потрібно розрізняти взаємозв'язки між об'єктами, між властивостями одного об'єкта й між властивостями різних об'єктів.

У процесі проектування об'єкти перетворяться у відносини, властивості в поля таблиць, методи - у процедури, форми й т.д. Вірно проведений об'єктно-орієнтований аналіз дозволяє значно полегшити роботу [31].

6.2 Електронні таблиці (на прикладі Excel)

Електронні таблиці, також як файли баз даних формату .DBF, виникли як метафора таблиці. Проста ідея про те, що було б добре, якби в деяких клітинах таблиці відразу ж відображався результат обчислень, що залежить від даних в інших клітинах таблиці, привела до появи в 1979 році програми VisiCalc. У лютому 1981 року з'явилася програма SuperCalc, а на початку 1983 року - пакет Lotus 1-2-3. В останній із цих програм було передбачене відображення результатів розрахунків у графічній формі й це багато в чому визначило успіх Lotus 1-2-3 на ринку в середині 80-х років. Наприкінці 80-х років ситуація на ринку електронних таблиць змінилася - завзята конкурентна боротьба йшла між програмами 1-2-3 версії 2.01 і 3.0 фірми Lotus Development Corporation, Excel 2.1 фірми Microsoft, Quattro 1.0 фірми Borland International і SuperCalc 5 фірми Computer Associates. До середини 1990 років найбільшу популярність придбав пакет Excel, чому чимало сприяли ринкові успіхи системи Microsoft Windows.

Електронні таблиці Excel дозволяють вирішувати дуже багато завдань обробки даних, сформованих у таблиці. В Excel передбачене автоматичне введення даних з SQL-Серверів і різних СУБД. Обробка даних передбачає, крім арифметичних операцій, можливість маніпулювання комплексними числами й матрицями, обчислення параметрів регресійних залежностей, перетворень Фур'є й інших статистичних характеристик. Передбачені дуже різноманітні методи для графічного висновку результатів, у тому числі, починаючи з Excel 7.0, відображення одержуваних результатів на числові карти у форматі ГІС

	A	B	C	D	E	F	G	H	I	J
1	Вміст гіпотетичних солей у зрощувальній воді (Північно-Кримський канал)									
2	Рік	$Ca(HCO_3)_2$	$Mg(HCO_3)_2$	$NaHCO_3$	$CaSO_4$	Na_2SO_4	$MgSO_4$	NaCl	$MgCl_2$	
3	1987	40	21,2	-	11,4	8,6	2,7	17,1	-	
4	1988	42,4	21,2	-	12,1	9,1	-	15,2	-	
5	1989	35	25	2,5	12,5	12,5	-	12,5	-	
6	1990	33,3	23,8	-	9,5	14,3	4,8	14,3	-	
7	1991	38,9	25	-	11,1	5,6	2,7	16,7	-	
8	1992	37,8	18,9	10,9	16,2	-	-	16,2	-	
9	1993	40	17,1	-	8,6	8,6	8,6	17,1	-	
10	1994	36,8	18,4	-	5,3	-	21,1	15,8	2,6	
11	1996	26,9	13,5	-	15,4	-	28,8	9,6	5,8	
12										

Рис. 6.1 Приклад БД в Excel

MapInfo. Складні завдання можуть вирішуватися в межах декількох зв'язаних таблиць, що мають назву в Excel'і - робоча книга (рис. 6.1).

У пакеті Excel передбачені різні засоби для збереження логічної цілісності інформації - автоматична заміна номерів аркушів і осередків при внесенні змін у книгу, автоматичне копіювання формул зі зрушенням номерів осередків у межах стовпців з подібними обчисленнями й багато чого іншого [25].

Питання для самоперевірки

1. Принципи логічного і фізичного проектування БД.
2. Корпоративні та «настільні» СУБД.
3. Переваги й недоліки корпоративних та «настільних» СУБД.
4. Послідовність створення інформаційної моделі.
5. Електроні таблиці.
6. Нормалізація інформаційної моделі.
7. Рівні незалежності даних.
8. Транзитивна залежність.
9. Історія створення електронних таблиць.

7. ЗАПИТИ ДО БАЗ ДАНИХ

Звернення користувача до системи за інформацією називається запитом.

Запити мови обробки даних бувають "плановані" і "неплановані".

1. Плановий запит - це запит, необхідність якого передбачена заздалегідь. Адміністратор бази даних, можливо, повинен настроїти фізичний проект бази даних таким чином, щоб гарантувати достатню швидкодію для таких запитів.

2. Непланований запит - це, навпаки, спеціальний запит, необхідність якого не була передбачена заздалегідь. Фізичний проект бази даних може підходити, а може й не підходити для розглянутого спеціального запиту. Загалом, одержання найбільшої можливої продуктивності для непланованих запитів являє собою одну із проблем СУБД. Плановані запити характерні для "операційних" додатків, а неплановані - для додатків "підтримки рішень". Більше того, плановані запити звичайно здійснюються з написаних заздалегідь додатків, а неплановані запити за визначенням виробляються інтерактивно.

7.1 Типи та принципи побудови запитів до баз даних

Типи запитів до бази даних - запити на вибірку, підсумковий, перекреслений запити й т.п., їх призначення й принципи побудови. У найпростішому випадку фіксується невелике число можливих запитів (рис. 7.1), на які заздалегідь готові відповіді.

Трохи більш складний випадок - запит, у якому зафіксований набір ознак. Значення деяких ознак вказуються користувачем (за ними треба шукати), значення інших ознак повинна вказати система. Тут природна форма запиту - бланк (на папері або на екрані дисплею), графи якого - стандартні, але число варіантів заповнення може бути великим.

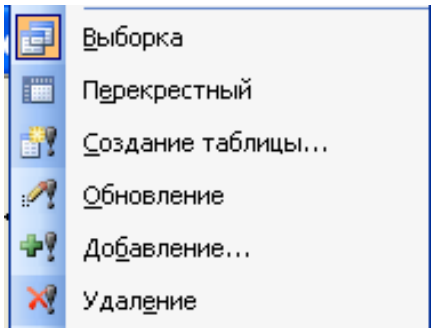


Рис. 7.1. Види запитів в Access

Користувачів можна розділити на три більші групи.

• Перша — прикладні програмісти, які відповідають за написання прикладних програм, що використовують базу даних. Для цих цілей

застосовуються мови COBOL, PL/1 або більш сучасні, такі як Pascal. Прикладні програми виконують над даними всі стандартні операції: вибірку існуючої інформації, вставку нової інформації, видалення або відновлення існуючої інформації. Всі ці функції виконуються через відповідний запит до СУБД. Ці програми можуть бути простими програмами пакетної обробки або оперативних додатків, функція яких — підтримка роботи кінцевого користувача, що має безпосередній оперативний доступ до бази даних через робочу станцію або термінал. Більшість сучасних додатків відноситься до оперативного.

• Друга — кінцеві користувачі, які працюють з системами баз даних безпосередньо через робочу станцію або термінал. Кінцевий користувач може одержати доступ до бази даних, використовуючи один з оперативних додатків, або ж скористатися інтегрованим інтерфейсом програмного забезпечення самої системи баз даних. Такий інтерфейс також підтримується оперативним додатком, але цей додаток не створюється користувачем, він є вбудованим у систему баз даних. У більшості систем є, принаймні, один такий вбудований додаток, а саме: процесор мови запитів, що дозволяє користувачеві вказувати команди або вираження високого рівня (такі як SELECT або INSERT) для даної СУБД. Мова SQL - типовий приклад мови запитів для бази даних.

Загальноприйнятий термін "мова запитів" не зовсім точно відображає розглянуте поняття, оскільки слово "запит" має на увазі лише вибірку, у той час як за допомогою мови SQL виконуються також операції відновлення, вставки й видалення (а можливо, і багато інших).

Крім мови запитів, у більшості систем також надаються додаткові вбудовані інтерфейси, у яких користувач у явному виді не використовує команд, таких як SELECT. Для роботи з базою даною користувач, наприклад, вибирає необхідні команди меню або заповнює поля у формах. Такі інтерфейси, засновані на меню й формах, полегшують роботу з базами даних тим, хто не має досвіду роботи з інформаційними технологіями (IT) Командний інтерфейс, тобто мова запитів, навпроти, вимагає деякого досвіду роботи з IT (можливо, не дуже великого). Однак командний інтерфейс звичайно більш гнучкий, ніж заснований на меню й формах; крім того, у мовах запитів звичайно є певні функції, які не підтримуються інтерфейсами, заснованими на меню й формах.

• Третя група — адміністратори бази даних, або АБД [31].

7.2 Можливості запитів та інструментальні засоби розробки прикладних програм

СУБД, орієнтовані на розроблювачів, мають розвинені засоби для створення додатків. До елементів інструментарію розробки додатків можна віднести:

• потужні мови програмування;

• засоби реалізації меню, екранних форм введення-виводу даних і генерації звітів;

• засоби генерації додатків (прикладних програм);

• генерацію здійсненних файлів.

Функціональні можливості моделей даних доступні користувачеві СУБД завдяки її мовним засобам.

Реалізація мовних засобів інтерфейсів може бути здійснена різними способами. Для висококваліфікованих користувачів (розроблювачів складних прикладних систем) мовні засоби найчастіше представляються в їх явній синтаксичній формі. В інших випадках функції мов можуть бути доступні непрямим образом, коли вони реалізуються у формі різного роду меню, діалогових сценаріїв або заповнюваних користувачем таблиць. За такими вхідними даними інтерфейсні засоби формують адекватні синтаксичні конструкції мови інтерфейсу й передають їх на виконання або включають у генерований програмний код додатка. Інтерфейси з неявним використанням мови широко використовуються в СУБД для персональних ЕОМ. Прикладом такої мови є мова QBE (Query-By-Example).

Мовні засоби використовуються для виконання двох основних функцій:

- опису подання бази даних;
- виконання операцій маніпулювання даними.

Перша із цих функцій забезпечується **мовою опису (визначення) даних** (МОД). Опис бази даних засобами МОД називається **схемою бази даних**. Вона включає опис структури бази даних і обмежень цілісності, що накладаються на неї, у рамках тих правил, які регламентовані моделлю даних використовуваної СУБД. МОД деяких СУБД забезпечують також можливості завдання обмежень доступу до даних або повноважень користувачів [32].

7.3 Мова маніпулювання даними

Мова маніпулювання даними (ММД) дозволяє запитувати, передбачені в системі операції над даними, з бази даних. Є численні приклади мов СУБД, що поєднують можливості опису даних і маніпулювання даними в єдиних синтаксичних рамках. Популярною мовою такого роду є реляційна мова SQL.

ММД не завжди синтаксично оформляється у вигляді самостійної мови. Вона може бути складовою частиною єдиної мови даних, що сполучить можливості визначення даних і маніпулювання даними. СУБД dBASE IV і FoxPro підтримують мову програмування xBASE, що дотепер є важливим стандартом для баз даних.

FoxPro 2.6 надає xBASE-програмам віконні, подійно-керовані якості. При складанні прикладної програми FoxPro використовує диспетчер проекту, керуючий різними файлами вихідного тексту й даних. Ця складова відслідковує індивідуальні елементи: програми, набори екранних форм,

звіти й файли баз даних і дозволяє компілювати прикладну програму в здійсненний файл.

Мова програмування Access Basic містить функції забезпечення зв'язку за протоколом OLE 2.0, що дозволяють управляти об'єктами з інших прикладних програм, сумісних з OLE 2.0. Крім того, ця мова дозволяє створювати об'єкти баз даних (запити, таблиці), змінювати структуру бази даних і створювати індекси безпосередньо із прикладної програми.

Всі розглянуті програмні засоби мають автоматизовані засоби створення екранних форм, запитів, звітів, меню, наклейок, стандартних листів. Для створення зазначених візуальних і структурних об'єктів ряд СУБД використовує спеціальні інструментальні засоби, що мають назву "майстри" або " чарівники" [9, 10, 23].

7.4. Мова структурованих запитів

Мова структурованих запитів SQL (Structured Query Language) була розроблена корпорацією IBM у сімдесятих роках, але загальне поширення одержала істотно пізніше, коли після появи комп'ютерних мереж, що зв'язують комп'ютери різних типів, потрібні були стандартні мови для обміну інформацією. Завдяки своїй незалежності від специфіки комп'ютера, а також підтримці лідерами в області технології реляційних баз даних, SQL стала, і в найближчому доступному для огляду майбутньому, залишиться такою, стандартною мовою. Синтаксис SQL схожий на синтаксис англійської мови й дозволяє конструювати досить складні запити. SQL є **непроцедурною мовою,** у ній відсутні багатостандартні для процедурних мов конструкції – функції, цикли, умовні оператори. SQL складається з інструкцій, які передаються СУБД, забезпечуючи виконання певних дій. Ці інструкції в загальному виді називаються пропозиціями, але частіше використовується термін «команда SQL». Інтерпретатори команд SQL вбудовуються в багато процедурних мов програмування, такі як Visual BASIC, C/C++. У цьому випадку команда звичайно формується у вигляді строкової змінної.

У багатьох пакетах команди SQL формуються автоматично зі спеціальних меню або форм. У цих випадках команди SQL використовуються в схованому від користувача виді. Прикладом такого використання є мова запитів за зразком QBE, що, наприклад, досить широко застосовувалася в СУБД Paradox. Іншим прикладом можуть служити електронні таблиці Excel. У цих таблицях за допомогою додатка MS-Query можна формувати різні запити за зразком до баз даних форматів Access, dBASE, Parodox, FoxPro і інших, доступ до яких здійснюється через інтерфейс ODBS (Open Data Base Connectivity).

Мова запитів SQL (Structured Query Language) реалізована у цілому ряді популярних СУБД для різних типів ЕОМ або як базова, або як альтернативна. У силу свого широкого використання вона є міжнародним

стандартом мови запитів. Мова SQL надає розвинені можливості як кінцевим користувачам, так і фахівцям в області обробки даних [5].

Сумісність з SQL-Системами відіграє більшу роль, коли передбачається проведення роботи з корпоративними даними. СУБД, добре підготовлені до роботи як засоби первинної обробки інформації для SQL-Систем, можуть відкрити двері в системи з архітектурою клієнт-сервер.

СУБД мають доступ до даних SQL у наступних випадках:

- бази даних сумісні з ODBC (Open Database Connectivity - відкрите з'єднання баз даних);
- реалізовано природну підтримку SQL-Баз даних;
- можлива реалізація SQL-Запитів локальних даних.

Багато СУБД можуть "прозоро" підключатися до вхідних SQL-Підсистем за допомогою ODBC або драйверів, що є їхньою частиною, тому існує можливість створення прикладних програм для них.

Access 2.0 і Paradox for Windows працюють з джерелами SQL-Даних, сумісних із системою ODBC. FoxPro (for DOS і for Windows) поставляються з додатковими бібліотеками, які забезпечують доступ до SQL-Баз даних, здатними працювати разом з системою ODBC, але ця можливість менш інтегрована, чим засоби первинного введення інформації в Access і Paradox for Windows.

Можна прямо управляти базами даних Access за допомогою мови SQL і передавати наскрізні SQL-Запити сумісним зі специфікацією ODBC SQL-Базам даних, таким, як MS SQL Server і Oracle, так що Access здатна служити засобом розробки масштабованих систем клієнт-сервер.

Тепер перейдемо до операторів SQL мови обробки даних (data manipulation language— DML). Основні оператори DML— це SELECT, INSERT, OPDATE і DELETE. Спочатку розглядаються операції вибірки (SELECT), а далі — операції відновлення (INSERT, UPDATE, DELETE).

Як уже згадувалося, мова обробки даних DML включає три операції відновлення: INSERT (вставка), UPDATE (зміна) і DELETE (видалення) [10].

Табличні вирази. Для вичерпного освітлення табличних виразів необхідно достатньо багато місця. Проте, принаймні для посилань, на рис. 7.2 приводиться достатньо повний опис BNF-граматики для таких виразів.

Виразом вибірки вважають табличний вираз, що не використовує операторів **UNION, INTERSECT** або **EXCEPT**. Як показано на рис. 7.2, вираз вибірки містить декілька компонентів: інструкції **SELECT, FROM, WHERE, GROUP BY** і **HAVING**.

Розглянемо послідовно кожний з цих компонентів.

Інструкція **SELECT.** Інструкція **SELECT** записується у вигляді

SELECT [ALL ¦ DISTINCT] select-item-commalist.

1. Список елементів вибірки select-item-commalist не повинен бути порожнім.

2. Якщо ні ключове слово ALL, ні DISTINCT не вказані, то мається на увазі ALL.

```
table-expression
        : : = join-table-expression
          I nonjoin-table-expression
join-table-expression
        : : = table-reference [NATURAL] JOIN
            table-reference [ON conditional-expression
I USING (column-commalist) ]
          I table-reference CROSS JOIN table-reference
          I ( join-table-expression )
table-reference
        : : = table [ [ AS ] range-variable
                  [ ( column-commalist ) ] ]
          I ( table-expression ) [ AS ] range-variable
                  [ ( column-commalist ) ]
                      I join-table-expression
nonjoin-table-expression
        : : = nonjoin-table-term
          I table-expression UNION [ ALL ]
                  [ CORRESPONDING [ BY ( column-commalist ) ] ]
                      table-term
          I table-expression EXCEPT [ ALL ]
                  [ CORRESPONDING [ BY ( column-commalist ) ] ]
                      table-term
nonjoin- table-term
        : : = nonjoin- table-primary
          I table-term INTERSECT [ ALL ]
                  [ CORRESPONDING [ BY ( column-commalist ) ] ]
                      table-primary
table-term
        : : = nonjoin-table-term
          I join-table-expression
table-primary
        ; : = nonjoin-table-primary
          I join-table-eapression
nonjoin-table-primary
        : : = TABLE table
          I table-constructor
          I select-expression
          I ( nonjoin-table-expression )
table-constructor
        : : = VALUES row-constructor-commalist
row-constructor
        : : = scalar-expression
          I ( scalar-expression-commalist )
          I ( table-expression )
select-expression
        : : = SELECT [ ALL | DISTINCT ] select-item-commalist
FROM table-reference-commalist
                    [ WHERE conditional-expression ]
                        [ GROUP BY column-commalist ]
                            [ HAVING conditional-expression ]
select-item
        : : = scalar-expression [ [ AS ] column ]
          I [ range-variable . ]
```

Рис. 7.1. BNF-грамматика для табличних виразів SQL

Інструкція FROM. Інструкція FROM записується у вигляді
FROM table-reference-commalist

Список посилань на таблиці (tabie-reference-commalist) не повинен бути порожнім. Хай обчислення вказаних табличних посилань дає таблиці A, B, .., C. Тогда результат обчислення інструкції FROM буде таблицею, еквівалентною декартовому перемноженню таблиць A, B,..., C.

Інструкція WHERE. Інструкція WHERE записується у вигляді
WHERE conditional-expression

Хай T — результат обчислення попередньої інструкції FROM. Тоді результатом інструкції WHERE буде таблиця, що похідна від T і виключає всі рядки, для яких обчислення умовного виразу (conditional-expression) не дає істину. Якщо інструкція WHERE опущена, результатом буде просте T.

Інструкція GROUP BY. Інструкція GROUP BY записується у вигляді
GROUP BY column-commalist

Список стовпців (column-commalist) не повинен бути порожнім. Хай T — результат обчислення попередніх інструкцій FROM і WHERE (якщо вони використовувалися). Кожен стовпець, вказаний в списку інструкції GROUP BY, повинен бути представлений уточненим ім'ям стовпця таблиці T. А результатом цієї інструкції буде згрупована таблиця, тобто набір груп рядків, похідних від таблиці T за допомогою концептуального перегруповування таблиці T в мінімальну кількість таких груп, що в межах однієї групи всі рядки мають однакове значення для комбінації стовпців, вказаних в інструкції GROUP BY. Тому результат — це не зовсім таблиця Проте інструкція GROUP BY ніколи не застосовується без відповідної інструкції SELECT, яка перетворить цей проміжний результат у вірну таблицю.

Вираз вибірки включає інструкцію GROUP BY, тобто обмеження на форму, яку може приймати інструкція SELECT.

Інструкція HAVING. Інструкція HAVING записується у вигляді
HAVING conditional-expression

Хай G (згрупована таблиця) — результат виконання попередньої інструкції FROM, інструкції WHERE (якщо вона присутня) і інструкції GROUP BY (якщо вона є). Якщо інструкції GROUP BY немає, то як G береться результат виконання попередньої інструкції FROM або інструкції WHERE і розглядається як згрупована таблиця, що складається тільки з однієї групи; тобто, в цьому випадку є неявна концептуальна інструкція GROUP BY, яка указує, що групованих стовпців немає зовсім. Результат інструкції HAVING — це згрупована таблиця, що похідна від G і виключає всі групи, для яких умовний вираз не є істиною.

Умовні вирази. Як і табличні вирази, умовні вирази зустрічаються в численних контекстах мови SQL і, зокрема, використовуються в інструкції WHERE з метою включення або виключення рядків для подальшої обробки.

Розглянемо умови MATCH і "все-або-який-небудь".

Умова MATCH. Умова MATCH має наступний вигляд:

row-constructor MATCH UNIQUE (table-expression)

Хай r1 — це рядок, одержаний в результаті обчислення row-constructor, і хай T — таблиця, одержана в результаті обчислення table-expression. Тоді обчислення умови MATCH даватиме істину тоді і тільки тоді, коли таблиця T міститиме рівно один рядок r2, таку, що порівняння r1 = r2 даватиме істину.

Умова "все-або-який-небудь".

Умова "все-або-який-небудь" має таку загальну форму:

row-constructor

comparison-operator qualifier(table-expression).

Скалярні вирази. Скалярні вирази в SQL, за суттю, прості. Табличні вирази, укладені в дужки, можуть тлумачити як скалярні значення, якщо тільки вони в результаті обчислення зводяться до таблиці з одним рядком і одним стовпцем. Як указувалося раніше, ця можливість, яка була введена в SQL/92, представляє головне удосконалення нової версії SQL в порівнянні з первинним варіантом.

Список основних операторів в алфавітному порядку: арифметичні оператори (+, —,*,/); BIT_LENGTH; CASE; CAST; CHARACTER_LENGTH; CURRENT_USER; LOWER; OCTET_LENGTH; POSITION; SESSION_USER; SUBSTRING; SYSTEM_USER; TRIM; UPPER; USER.

Оператор CASE. Оператор CASE повертає одне значення з вказаного набору значень залежно від певних умов.

Оператор CAST. Оператор CAST перетворить певне скалярне значення до певного скалярного типу даних (можливо, до домена, визначеного користувачем). Не всі пари типів даних взаємно конвертовані [10].

7.5 Архітектура " клієнт-сервер"

Загальне поширення комп'ютерних мереж породило ще одну, крім необхідності розробки стандартизованої мови запитів, проблему. Ця проблема виникає, коли кілька користувачів з різних комп'ютерів починають змінювати ту саму базу даних. Доти поки база даних відкрита «тільки для читання» особливих труднощів не виникають, але, як тільки декільком користувачам дозволяється модифікувати базу, виникають важко розв'язні конфлікти. Ці проблеми переборюються в рамках моделі бази даних типу «клієнт-сервер» (рис. 7.3). При реалізації цієї моделі система управління базами даних розділяється на дві частини - «клієнт» і «сервер». Програма «клієнт» розміщається на користувальницькій машині й дозволяє формувати запити (як правило, мовою SQL), які по мережі передаються на спеціалізовану машину («сервер»), де працює програма «сервер». У такий спосіб термін «сервер» іноді ставиться до комп'ютера, а іноді до програмного забезпечення.

Рис. 7.3. Схема «Клієнт-сервер»

Програма «сервер» обробляє запит, формує з бази даних необхідну вибірку записів і відсилає її програмі «клієнтові». Якщо користувач припускає змінювати інформацію в запитаній вибірці, доступ будь-якого іншого користувача для модифікації обраних записів блокується ("монопольне захоплення"). Якщо користувач запитує інформацію «тільки для читання», то доступ до обраних записів не обмежується ("колективне захоплення").

Основний механізм, що дозволяє уникнути конфліктів між користувачами, полягає в розбивці процесу обробки інформації на елементарні події – групи команд SQL, які можуть виконуватися (або не виконуватися) тільки всі разом. Такі групи команд називаються **трансакціями**. Трансакція починається щораз, коли на вхід «сервера» починають надходити команди SQL, якщо ніяка інша трансакція не є активною. Трансакція закінчується або командою внести зміни в базу даних, або відмовою від внесення змін («відкіт»). Якщо в процесі виконання команд виникає яка-небудь помилка, автоматично виконується «відкіт» і база даних залишається у вихідному стані [27, 32].

Питання для самоперевірки

1. Типи запитів до бази даних – запит на вибірку, підсумковий, перекреслений запити.
2. Призначення й принципи побудови запитів
3. Використання мови SQL для опису БД і маніпулювання даними в БД.
4. Поняття архітектура “клієнт – сервер”.
5. Поняття архітектура “файл – сервер”.
6. Групи користувачів.
7. Елементи інструментарію розробки прикладних програм.
8. Мова маніпулювання даними.
9. Характеристика SQL.
10. Основні оператори SQL – мови.
11. Інструкції табличних виразів SQL – мови.
12. Умовні вирази.
13. Скалярні вирази.
14. Поняття трансакції.

8. БЕЗПЕКА БАЗ ДАНИХ

8.1 Поняття захисту інформації

Захист інформації — комплекс заходів, що направлені на забезпечення найважливіших аспектів інформаційної безпеки (цілісності, доступності і, якщо потрібно, конфіденційності інформації та ресурсів, які використовуються для введення, зберігання, обробки і передачі даних) [25].

Система називається безпечною, якщо вона, використовуючи відповідні апаратні та програмні засоби, управляє доступом до інформації так, що тільки належним чином авторизовані особи або ж діючі від їх імені процеси одержують право читати, писати, створювати і видаляти інформацію.

Система вважається надійною, якщо вона з використанням достатніх апаратних та програмних засобів забезпечує одночасну обробку інформації різного ступеня секретності групою користувачів без порушення прав доступу [39].

Основними критеріями оцінки надійності є: політика безпеки і гарантованість.

Політика безпеки, будучи активним компонентом захисту (включає аналіз можливих загроз і вибір відповідних заходів протидії), відображає той набір законів, правил та норм поведінки, яким користується конкретна організація при обробці, захисті й розповсюдженні інформації. Вибір конкретних механізмів забезпечення безпеки системи виробляється відповідно до сформульованої політики безпеки.

Гарантованість, будучи пасивним елементом захисту, відображає міру довіри, яка може бути надана архітектурі та реалізації системи (показує, наскільки коректно вибрані механізми, що забезпечують безпеку системи). При оцінці ступеня гарантованості, за яким систему можна вважати надійною, центральне місце займає достовірна (надійна) обчислювальна база. Достовірна обчислювальна база (ДОБ) є повною сукупністю захисних механізмів комп'ютерної системи, яка використовується для втілення в життя відповідної політики безпеки.

Надійність ДОБ залежить виключно від її реалізації і коректності введених даних (наприклад, даних про благонадійність користувачів, що визначаються адміністрацією). Межа ДОБ утворює периметр безпеки. Компоненти ДОБ, що знаходяться усередині цієї межі, повинні бути надійними. Від компонентів, що знаходяться поза периметром безпеки, взагалі кажучи, не вимагається надійності. Оскільки зараз широко застосовуються розподілені системи обробки даних, то під «периметром безпеки» розуміється межа володінь певної організації, в підпорядкуванні якої знаходиться ця система [9].

Контроль допустимості виконання суб'єктами певних операцій над об'єктами, тобто функції моніторингу, виконується достовірною

обчислювальною базою. При кожному зверненні користувача до програм або даних монітор перевіряє допустимість даного звернення (узгодженість дії конкретного користувача із списком дозволених для нього дій). Реалізація монітора звернення називається ядром безпеки, на базі якої будуються всі захисні механізми системи. Ядро безпеки повинне гарантувати власну незмінність [9].

Існує два традиційні способи захисту бази даних: установка пароля, що вимагається при відкритті бази даних, і захист на рівні користувачів, яка дозволяє обмежити, до якої частини бази даних користувач матиме доступ або яку її частину він зможе змінювати. Простим способом захисту баз даних є установка пароля для відкриття бази даних. Після того, як пароль встановлений, при кожному відкритті бази даних з'являтиметься діалогове вікно, в яке вимагається ввести пароль.

Найгнучкіший і поширеніший спосіб захисту бази даних називається захистом на рівні користувачів. Цей спосіб захисту подібний способам, що використовувані в більшості мережних систем [39].

8.2 Захист ПК від несанкціонованого доступу (НСД)

Основні механізми захисту ПК від НСД можуть бути представлені наступним переліком:

1) фізичний захист ПК і носіїв інформації;
2) розпізнання (аутентифікація) користувачів і використовуваних компонентів обробки інформації;
3) розмежування доступу до елементів інформації, що захищається;
4) криптографічне закриття захищаємої інформації, що зберігається на носіях (архівація даних);
5) криптографічне закриття інформації, що захищається, в процесі безпосередньої її обробки;
6) реєстрація всіх звернень до інформації, що захищається [39].

8.3 Захист інформації в базах даних

У сучасних СУБД підтримується один з двох найзагальніших підходів до питання забезпечення безпеки даних: виборчий підхід і обов'язковий підхід. У обох підходах одиницею даних або «об'єктом даних», для яких повинна бути створена система безпеки, може бути як вся база даних цілком, так і будь-який об'єкт усередині бази даних [37].

Ці два підходи відрізняються наступними властивостями.

У разі виборчого управління деякий користувач має різні права (привілеї або повноваження) при роботі з даними об'єктами. Різні користувачі мають різні права доступу до одного і того ж об'єкту. Виборчі права характеризуються значною гнучкістю.

У разі обов'язкового управління, навпаки, кожному об'єкту даних привласнюється деякий класифікаційний рівень, а кожен користувач має

деякий рівень допуску. При такому підході доступом до певного об'єкту даних мають тільки користувачі з відповідним рівнем допуску.

Для реалізації обов'язкового принципу передбачені наступні методи. У базу даних вводиться новий тип об'єктів БД — це користувачі. Кожному користувачу в БД привласнюється унікальний ідентифікатор. Для додаткового захисту кожен користувач окрім унікального ідентифікатора забезпечується унікальним паролем, причому якщо ідентифікатори користувачів в системі доступні системному адміністратору, то паролі користувачів зберігаються найчастіше в спеціальному кодованому вигляді і відомі тільки самим користувачам [37].

Привілеї або повноваження користувачів або груп — це набір дій (операцій), які вони можуть виконувати над об'єктами БД.

У останніх версіях ряду комерційних СУБД з'явилося поняття «ролі». Роль — це пойменований набір повноважень. Існує ряд стандартних ролей, які визначені у момент установки серверу баз даних. І є можливість створювати нові ролі, групуючи в них довільні повноваження. Введення ролей дозволяє спростити управління привілеями користувачів, структурувати цей процес [25].

Користувачу може бути призначена одна або декілька ролей.
Об'єктами БД, які підлягають захисту, є всі об'єкти, що зберігаються в БД: таблиці, уявлення, процедури, що зберігаються, і тригери. Для кожного типу об'єктів є свої дії, тому для кожного типу об'єктів можуть бути визначені різні права доступу.

На самому елементарному рівні концепції забезпечення безпеки баз даних виключно прості. Необхідно підтримувати два фундаментальні принципи: перевірку повноважень і перевірку достовірності (аутентифікацію) [39].

Перевірка повноважень заснована на тому, що кожному користувачу або процесу інформаційної системи відповідає набір дій, які він може виконувати по відношенню до певних об'єктів. Перевірка достовірності означає достовірне підтвердження того, що користувач або процес, що намагається виконати санкціоновану дію, дійсно той, за кого він себе видає.

Система призначення повноважень має в деякому роді ієрархічний характер. Найвищі права і повноваження маіє системний адміністратор або адміністратор серверу БД. Традиційно тільки цей тип користувачів може створювати інших користувачів і наділяти їх певними повноваженнями [24].

Далі схема надання повноважень будується за наступним принципом. Кожен об'єкт в БД має власника — користувача, який створив даний об'єкт. Власник об'єкту має всі права-повноваження на даний об'єкт, зокрема він має право надавати іншим користувачам повноваження щодо роботи з даним об'єктом або забирати у користувачів раніше надані повноваження [25].

8.4 Технічний захист конфіденційної інформації

Технічний захист конфіденційної інформації здійснюється криптографічними методами, направленими на запобігання просочуванню інформації, що захищається по технічних каналах; на захист інформації від несанкціонованого доступу до неї і спеціальних дій на інформацію з метою її знищення, спотворення або блокування.

Захист конфіденційної інформації здійснюється методом прозорого шифрування за допомогою технології роботи з віртуальним логічним диском. Захист конфіденційної інформації забезпечується шифруванням даних "на льоту" за допомогою стійких алгоритмів шифрування. При читанні даних з диска відбувається їх розшифрування, при записі на диск - шифрування. Дані, що знаходяться на диску, завжди зашифровані [25, 39].

Апаратні засоби захисту інформації - це спеціальні засоби, що безпосередньо входять до складу технічного забезпечення і виконують функції захисту як самостійно, так і в комплексі з іншими засобами, наприклад з програмними. Під програмно-апаратними засобами захисту, в даному випадку, розуміються засоби, засновані на використовуванні так званих "апаратних (електронних) ключів".

Існуючі апаратні засоби захисту даних, що розмежовують доступ в систему на основі різної ідентифікуючої інформації (сітківка ока, відбитки пальців, електронні ключі і т. п.), можуть дати апаратний збій, пов'язаний з невірною обробкою системної дати при ідентифікації користувача.

До апаратних засобів захисту даних відносяться різні електронні, електронно-механічні, електронно-оптичні пристрої. До теперішнього часу розроблене значне число апаратних засобів захисту даних різного призначення, проте найбільше поширення набувають наступні:

- спеціальні регістри для зберігання реквізитів захисту: паролів, ідентифікуючих кодів, грифів або рівнів секретності;
- генератори кодів, призначені для автоматичної генерації ідентифікуючого коду пристрою;
- пристрої вимірювання індивідуальних характеристик людини (голосу, відбитків) з метою його ідентифікації;
- спеціальні біти секретності, значення яких визначає рівень секретності інформації, що зберігається в захисному пристрої, якому належать дані біти;
- схеми переривання передачі інформації в лінії зв'язку з метою періодичної перевірки адреси видачі даних [40].

Апаратні ключі захисту складаються з власне ключа, що підключається до LPT або COM-порту комп'ютера і програмного забезпечення (драйверів для різних операційних систем і модуля, вбудовуваного в програму, що захищається).

Апаратні ключі захисту можна намагатися класифікувати за декількома ознаками. Якщо розглядати можливі типи апаратних ключів захисту підключення, то бувають, наприклад, ключі на порт принтера (LPT), послідовний порт (COM), USB-порт і ключі, що підключаються до

спеціальної плати, що вставляється всередину комп'ютера. Апаратні ключі захисту мають дуже низьке енергоспоживання, внаслідок чого їх можна підключати до інформаційних портів без додаткового живлення. Існують також апаратні ключі захисту на основі PIC-контролерів і ASIC-чипів [24].

Захист даних шифруванням - одне з можливих рішень проблеми їх безпеки. Шифруванням (encryption) називають процес перетворення відкритих даних (plaintext) в зашифровані (шифр-текст, ciphertext) або зашифрованих даних у відкриті за певними правилами із застосуванням ключів.

Процес захисту даних шифруванням відбувається в два методи шифрування даних: необоротне (one-way) (хешування) і оборотне (two-way). Технічно хешування - це не шифрування, але обидві технології при захисті даних можуть застосовуватися схожим чином, тобто перетворювати дані з відкритого тексту в шифрований. Більшість засобів захисту даних шифруванням реалізована у вигляді спеціалізованих апаратних пристроїв [27].

8.5 Багаторівнева модель безпеки баз даних

Сполучення засобів перевірки повноважень і перевірки дійсності - могутнє знаряддя боротьби за безпеку інформаційних систем і баз даних. Якщо всі користувачі, що працюють в інтерактивному режимі або запускають пакетні додатки, досить надійні і мають доступ до максимально закритої інформації, збереженої в системі, то згаданий підхід до забезпечення безпеки може бути цілком достатнім [40].

Однак він виявляється незадовільним, якщо в установі необхідно організувати дійсно багаторівневе середовище захисту інформації.

Багаторівневий захист означає, що:

- в обчислювальній системі зберігається інформація, що відноситься до різних класів таємності;
- частина користувачів не мають доступу до максимально секретного класу інформації.

Багаторівневий захист баз даних будується на основі моделі Белл-ЛаПадула (Bell-LaPadula), що призначений для управління суб'єктами, тобто активними процесами, що запитують доступ до інформації, і об'єктами, тобто файлами, представленнями, записами, полями або іншими сутностями даної інформаційної моделі. Об'єкти піддаються класифікації, а суб'єкти зараховуються до одного з рівнів благонадійності (clearance). Класи та рівні благонадійності називаються класами доступу або рівнями. Клас доступу характеризується двома компонентами [39].

Перший компонент визначає ієрархічне положення класу. Другий компонент являє собою безліч елементів з неієрархічного набору категорій, що можуть відноситися до будь-якого рівня ієрархії.

У приватній компанії можливі такі рівні ієрархії:

- секретно;
- для обмеженого поширення;
- конфіденційно;
- для службового користування;
- несекретно.

Другий компонент у такій компанії міг би належати до набору категорій:
- недоступно для службовців за контрактом;
- фінансова інформація компанії;
- інформація про тарифи [40].

Коли деяка область пам'яті, наприклад утримуючий об'єкт бази даних, прямо або побічно використовується як засіб розкриття класифікованих матеріалів, відкривається непрямий канал пам'яті [9].

Непрямі канали часу звичайно виключаються за рахунок відповідних рішень у сфері комунікацій, що дозволяє замаскувати дійсні обсяги переданої інформації. Т. е. непрямі канали часу - це, насамперед, питання комунікацій. Але в міру того, як росте ступінь розподіленності баз даних і зростає значення мір безпеки, у надійному середовищі баз даних ці питання повинні розглядатися поряд із проблемами непрямих каналів пам'яті [40].

8.6 Сучасні досягнення в забезпеченні безпеки БД

Компанія **Microsoft** випустила СУБД **SQL Server 2005** з кодовою назвою Yukon, куди ввійшли засоби шифрування, що затрудняють доступ до інформації для хакерів і інших не наділених відповідними правами користувачів У новій версії SQL Server шифруються і дані, що зберігаються в самій БД, що робить її набагато стійкішою до атак. Крім того, до неї увійшла нова концепція уніфікованої системи зберігання даних, яка полегшить пошук і витягання інформації [33].

Oracle випустила опцію **Oracle Label Security** для своїх продуктів **Oracle 8i і Oracle 9i.** Ця опція дозволяє позначати дані на самому нижньому рівні БД, щоб управляти доступом користувачів до певної інформації. Вона дозволяє управляти доступом до бази даних на рівні записів.

Інструмент Oracle Label Security є особливо корисним сервіс - провайдерам, що надають ту або іншу послугу декільком користувачам. Вони можуть надавати доступ до даних одному, і лише одному клієнту.

IBM випустила нову версію DB2 під Windows 2000 і Unix з функціями шифрування і дешифрування. В даний час тільки версія DB2 для мейнфреймів має підтримку апаратного шифрування [34].

Корпорація Fujitsu Siemens Computers випустила нову версію програмного рішення NetWorker Backup Suite. Версія 7.3 модуля NetWorker PLUS орієнтована на Oracle і дозволяє виконувати повністю автоматичне резервне копіювання і відновлення баз даних за розкладом. З розширенням для пристроїв NAS — модулем NetWorker PLUS для Oracle

для NDMP (Network Data Management Protocol) — це рішення є єдиним присутнім в даний час на ринку програмним засобом, який здатний повністю інтегрувати моментальні знімки баз даних в концепцію резервного копіювання, що дозволяє створювати резервні копії протягом декількох секунд, значно прискорює процеси відновлення і знімає з серверів баз даних навантаження по резервному копіюванню. В результаті забезпечується безперервний захист даних підприємства і підвищення ефективності процесів резервного копіювання і відновлення.

До складу пакету NetWorker Backup Suite входять також модулі FireWall eXtension і SAN Registration Service (SRS). Програмне рішення Fujitsu Siemens Computers FireWall eXtension дозволяє використовувати єдиний порт брандмауера для взаємодії з всіма компонентами NetWorker (серверами, клієнтами і вузлами зберігання даних). Модуль SAN Registration Service забезпечує необхідну синхронізацію при динамічному підключенні дисків в системі зберігання до серверів. В результаті підвищується ефективність використовування існуючої системи зберігання даних і скорочується її період окупності [35].

Отже, для мінімізації ризику втрат необхідна реалізація комплексу нормативних, організаційних і технічних захисних мір, у першу чергу: уведення рольового управління доступом, організація доступу користувачів по пред'явленню цифрового сертифіката, а в найближчій перспективі - промислове рішення щодо вибіркового шифрування і застосування алгоритмів ДСТ для шифрування обраних сегментів бази.

Для повного рішення проблеми захисту даних адміністратор безпеки повинний мати можливість проводити моніторинг дій користувачів, у тому числі з правами адміністратора.

Питання для самоперевірки

1. Основні критерії оцінки надійності комп’ютерної системи.
2. Основні механізми захисту ПК від несанкціонованого доступу.
3. Традиційні способи захисту баз даних.
4. Апаратні засоби захисту інформації.
5. Сучасні програмні продукти безпеки баз даних.
6. Поняття захисту інформації.
7. Поняття політики безпеки.
8. Поняття ролі в безпеці БД.
9. Ключ захисту.
10. Захист даних шифруванням.
11. Компоненти класу доступу.

КОРОТКИЙ ТЕРМІНОЛОГІЧНИЙ СЛОВНИК

Атака на комп'ютерну систему - це дія, що починається зловмисником, що полягає в пошуку і використанні тієї або іншої уразливості.

Атомарність - виражається в тім, що трансакція повинна бути виконана в цілому або не виконана зовсім.

Атрибут (властивість) - це однозначний факт про деяку сутність, тобто дані про сутність, які потрібно зберегти.

Вихідні дані — це повідомлення й результати, видані системою.

Внутрішня модель системи - фізична модель, що визначає розміщення даних, методи доступу й техніку індексування.

Вхідні дані — це інформація, передана системі.

Вузол — це сукупність атрибутів даних, що описують деякий об'єкт.

Дигітайзер - пристрій для вводу в комп'ютер координат крапок із прикріпленого до планшету листа за допомогою рухомого курсору з клавіатурою

Довговічність - якщо трансакція завершена успішно, то ті зміни, у даних, які були нею зроблені, не можуть бути загублені ні за яких умов.

Домен - безліч атомарних значень того самого типу.

Ізольованість - означає, що конкуруючі за доступ до БД трансакції фізично обробляються послідовно, ізольовано друг від друга, але для користувачів це виглядає так, начебто вони виконуються паралельно.

Загроза безпеки комп'ютерної системи - це потенційно можлива подія, що може зробити небажаний вплив на саму систему, а також на інформацію, що зберігається в ній.

Запис — сукупність логічно зв'язаних полів.

Запис, що зберігається, — це набір зв'язаних полів, що зберігаються.

Запит - це процес звернення користувача до БД із метою введення, одержання або зміни інформації в БД.

Зв'язок - асоціювання двох або більше сутностей.

Клас - це безліч предметів реального миру, зв'язаних спільністю структури й поводженням.

Ключ – мінімальний набір атрибутів, за значеннями яких можна однозначно знайти необхідний екземпляр сутності.

Користувач БД - це програма або людина, що звертаються до БД мовою маніпулювання даними.

Логічна модель - версія концептуальної моделі, що може бути забезпечена конкретною СУБД**.**

Логічна структура БД - це визначення БД на фізично незалежному рівні, ближче всього відповідної концептуальної моделі БД.

Можливість реалізації вилученої трансакції - це обробка однієї трансакції, що складає із множини SQL-запитів, на одному вилученому вузлі.

Найменша одиниця даних реляційної моделі – це окреме атомарне (нерозкладне) для даної моделі значення даних. Так, в одній предметній

області прізвище, ім'я та по батькові можуть розглядатися як єдине значення, а в іншій - як три різних значення.

Об'єкт - це абстракція безлічі предметів реального миру, що володіють однаковими характеристиками й законами поводження.

Погодженість - гарантує, що в міру виконання трансакцій, дані переходять із одного погодженого стану в інше, тобто трансакція не руйнує взаємної погодженості даних.

Поле — елементарна одиниця логічної організації даних.

Поле, що зберігається, — це якнайменша одиниця даних, що зберігаються.

Предметна область — це частина реального світу, в межах якої визначається набір даних і методів, маніпулювання ними для вирішення конкретних завдань або виконання досліджень.

Простий ключ - поле, кожне значення якого однозначно визначає відповідний запис.

Роль — це пойменований набір повноважень.

Сканер - пристрій для сканування, т.є. автоматизованого оцифрування растрових зображень: а) периферійний пристрій комп'ютера; б) сенсор дистанційного зондування.

Стрижнева сутність – незалежна сутність.

Структурування — це введення угод про способи подання даних.

Схема бази даних - опис бази даних засобами МОД.

Сутність - це відмінний об'єкт, де об'єкт, про який йде мова, може бути настільки конкретним або абстрактним, наскільки нам подобається. Відомості про сутності мають вигляд атрибутів і/або зв'язків.

Тематичні дані — групування даних за характеристиками основних компонентів предметної області, що здійснюються за принципом змістовної організації інформації

Топологія БД (структура РБД) - це схема розподілу фізичних БД по мережі. Локальна автономність означає приналежність локальному власникові інформації локальної БД і пов'язаних з нею певних даних. Вилучений запит - це запит, що виконується з використанням модемного зв'язку.

Трансакція - це послідовність операцій модифікації даних у БД, що переводить БД із одного несуперечливого стану в інший несуперечливий стан.

Уразливість комп'ютерної системи - це якась її невдала характеристика, що уможливлює виникнення загрози.

Файл (таблиця) — сукупність екземплярів записів однієї структури.

ТЕСТОВІ ЗАВДАННЯ ДЛЯ САМОКОНТРОЛЮ

1	**Проектування бази даних – побудова...**
	1. взаємозалежних об'єктів 2. комплексу растрових даних 3. комплексу взаємозалежних моделей даних 4. взаємопов'язаних баз даних

2	**Типи даних**
	1. логічні, інформаційні 2. векторні, растрові 3. зовнішні, внутрішні 4. реальні, моделі

3	**Рівні подання даних**
	1. основний, допоміжний, середній 2. концептуальний, внутрішній, зовнішній 3. нормальний, перемінний, структурний 4. логічний, натурний, модельний

4	**Система баз даних**
	1. комп'ютеризована система зберігання записів 2. комп'ютеризована однокористувальницька система 3. комп'ютеризована система вхідних даних 4. комп'ютеризована система централізованого управління

5	**Дані в БД у загальному випадку**
	1. централізовані, децентралізовані 2. постійні, транзитні 3. інтегровані, загальні 4. збережені, розрізнені

6	**Моделі даних**
	1. ієрархічна, мережна, реляційна 2. інфологічна, даталогічна, фізична 3. симетрична, уфологічна, часова 4. системна, вибіркова, гібридна

7	**Структури БД**
	1. деревоподібна, мережна, таблична 2. багатошарова, одношарова, тришарова 3. реляційна, ієрархічна, мережна 4. атрибутивна, векторна, растрова

8	**Помилки в БД ГІС**
	1. систематичні, грубі, випадкові 2. візуальні, скриті, кодовані 3. графічні, атрибутивні, кодування 4. одноразові, систематичні, часові

9	**База даних**
	1. пошаровій збір інформації 2. організований набір взаємозалежних файлів даних 3. централізований набір інформаційних файлів 4. набір реляційних та табличних даних

10	**Основні роботи з БД**
	1. введення, оцінювання, кодування 2. введення, зміна, оцінювання 3. введення, збереження, редагування 4. збирання, введення, редагування

11	**Кількість етапів проектування БД**
	1. 4 2. 6 3. 8 4. 10

12	**Встановити відповідність**	
	1. відношення 2. кортеж 3. атрибут	A. запис B. поле C. файл

13	**Що є характеристикою реляційної системи**
	1. дані для користувача передаються у вигляді таблиць 2. користувач застосовує просторову геометрію 3. користувачеві надаються оператори 4. централізований набір файлів користувачем

14	**Способи уведення даних в ГІС**
	1. прозорова сітка на карті 2. дигітайзер 3. сканер 4. комп'ютер 5. квадрофон

15	**Прилади цифрового уведення в ГІС**
	1. комп'ютер 2. дигітайзер 3. квадрофон 4. селектор

16	**Перший етап розробки ГІС**
	1. фізична модель організації даних 2. концептуальна модель організації даних 3. логічна модель організації даних

17	Основні підсистеми агромеліоративної геосистеми
	1. природно-ресурсна основа 2. еколого-меліоративний моніторинг 3. виробничо-технологічна структура території 4. територіальна прив'язка об'єктів 5. нормативно-регламентуюча база прийняття рішень

18	База знань в ГІС
	1. засіб точного відображення уявлень людини про системи та процеси предметної області реального світу 2. сукупність знань для обґрунтування та оптимізації прийняття рішень 3. засіб централізованого управління базами даних

19	Подання всього вмісту бази даних...

1. концептуальна схема 2. концептуальне подання 3. концептуальне відображення 4. концептуальна модель

20	**Реляційна модель баз даних – вид...**
	1. фізичної моделі 2. міфологічної моделі 3. логічної моделі 4. ієрархічної моделі

21	**Засоби апаратного забезпечення СУБД**
	1. накопичувані інформації 2. процесори 3. принтери 4. сканери 5. утиліти

22	**Системи управління базами даних**
	1. ASSIST 2. Paradox 3. Oracle 4. Fox BASE 5. Macintosh

23	**Адміністратор баз даних**
	1. людина, відповідальна за стратегію і політику прийняття рішень, пов'язаних з даними об'єкта. 2. людина, що забезпечує необхідну технічну підтримку виконання рішень, пов'язаних з даними об'єкта. 3. людина, що забезпечує безпеку та зберігання даних об'єкта, для прийняття рішень.

24	**Безліч предметів реального миру, зв'язаних спільністю структури й поводженням - ...**
	1. об'єкт 4. набір 2. блок 5. модель 3. клас

25	**Мінімальний набір атрибутів, за значенням яких можна однозначно знайти необхідний екземпляр сутності**
	1. первинний ключ 2. зовнішній ключ 3. ключ 4. можливий ключ 5. вторинний ключ

26	**SQL – мова...**
	1. структурованих запитів 2. інформаційних принципів 3. концептуальних термінів 4. реляційних атрибутів

27	**Запити мови обробки даних бувають**
	1. плановані 2. своєчасні 3. неплановані 4. операційні 5. тотожні

28	**Користувачі ГІС**
	1. прикладні програмісти 2. адміністратори бази даних 3. оперативні програмісти 4. кінцеві користувачі 5. системні провайдери

29	**Опис бази даних засобами МОД**
	1. мова визначення даних 2. схема бази даних 3. непроцедурна мова 4. трансакція

30	**Пойменована характеристика сутності**
	1. ключ 2. зв'язок 3. клас 4. атрибут 5. мова

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Карпінський Ю., Лященко А. Концептуальні створення національної інфраструктури геопросторових даних України / Сучасні досягнення геодезичної науки та виробництва, - 2005., С. 295.
2. Карпінський Ю., Лященко А. Аналіз міжнародного досвіду створення інфраструктури геопросторових даних / Сучасні досягнення геодезичної науки та виробництва. – 2006. Вип.1. –С.151-164.
3. Растоскуев В.В. Експертна система для обробки даних контролю забруднень атмосфери. - Спб.: 1997. - 261 с.
4. Геоинформатика. Толковый словарь основных терминов. - М.: ГИС-Ассоциация, 1999. – 204 с.
5. Інформаційно-обчислювальне забезпечення моніторингу меліорованих земель.Ч.1: Методика організації системи інформаційного забезпечення моніторингових робіт на зрошуваних землях. Посібник 3 до ВБН 33-5.5-01-97 "Організація і ведення еколого-меліоративного моніторингу".Ч.1: Зрошувані землі. – К., 2002. – 65 с.
6. Тикунов В.С., Цапук Д.А. Устойчивое развитие территорий: картографо-геоинформационное обеспечение. – М.; Смоленск: Изд-во СГУ, 1999. – 176 с.
7. Юрченко И.Ф. Информационные технологии обоснования мелиораций. – М., 2000. – 238 с.
8. Ромащенко М.І., Рачинська Е.С., Шевченко А.М. Інформаційне забезпечення зрошуваного. Концепція, структура, методологія організації/ За ред. М.І. Ромащенка. – К.: Аграрна наука, 2005. – 196 с., 8 карт.
9. ДеМерс М.Н. Географические информационные системы. Основа.: Пер. с англ. – М.: Дата+, 1999. - 490с.
10. Дейт К.Дж. Введение в системы баз данных. 6-е изд.: Пер. с англ. – К; М.; СПб: Изд. Дом Вильямс, 1999. – 848 с.: илл.
11. Healey R.G "Database Management Systems." In Geographical Information Systems: Principles and Application, D.J. Maguire, M.F. Goodchild, and D.W. Rhind, Eds. Essex: Longman Scientific & Technical, 1991
12. Ullman J.D. Principles of Database Systems. Rockville, MD: Computer Science Press. U.S. Department of Commerce, Bureau of the Census, 1969. The DIME Geocoding System. Iri Report No. 4, Census Use Study. Washington, DC: Government Printing Office. 1982.
13. Codd E.F. "A Relational Model of Data for Large Shared Data Banks." Communications of the Association for Computing Machinery, 1970. - 13(6):377-387.
14. Kent W. A simple guide to five normal forms in relational database theory. Communications of the Association for Computing Machinery –1983.-26(2): 120-125.
15. Dangermond J. "A Classification of Software Components Commonly Used in Geographic Information Systems." In Proceedings of the U.S.-Australia

Workshop on the Design and Implementation of Computer-Based Geographic Information Systems, Honolulu, HI, - 1982. - pp. 70-91.
16. Aronson P. "Applying Software Engineering to a General Purpose Geographic Information System," In Proceedings of A UTOCARTO 7. Falls Church, VA:ASPRS, - 1985. - pp. 23-31.
17. Morehouse S. "The Architecture of ARC/INFO." In Proceedings of AVTOCARTO 9. Falls Church, VA: ASPRS, - 1989. - pp. 266-277.
18. Guptill, S. "Desirable Characteristics of a Spatial Database Management System." In Proceedings ofAUTOCARTO 8. Falls Church, VA: ASPRS, - 1987. - pp. 278-281.
19. DeMers M.N., Fisher P.P. "Comparative Evolution of Statewide Geographic Information Systems in Ohio." InternationalJoumal of Geographical Information Systems, - 1991. - 5(4):469-485.
20. Burrough P.A. Geographical Information Systems for Natural Resources Assessment. New York: Oxford University Press. 1986.
21. Банки географических данных для тематического картографирования/ Под ред. К.А. Салищева, С.Н. Сербенюка. – М.: Изд-во Моск. Ун-та, 1987.- 188 с.
22. Кириллов В.В. Основы проектирования реляционных баз данных .Учебное пособие. - СПб.: ИТМО, 1994. - 90 с.
23. Кауфельд Д. Access 2003 для “чайников”.: Пер. с англ.. – М.: Издательский дом “Вильямс”, 2005. – 320 с.:ил.
24. Дейт К. Руководство по реляционной СУБД DB2. - М.: Финансы и статистика, 1988. - 320 с.
25. http://www.citforum.ru/database/sql_kg/index.shtml “Основы проектирования реляционных баз данных ”
26. Зейлер М. Моделирование нашого мира. Руководство ESRI по проектированию базы геоданных. – Нью-Йорк, ESRI Press, 1999. – 254 с.
27. ARC/INFO Управление данными. Концепции, модели данных, разработка баз данных и хранение данных. – М.: Дата+, 1998. – 300 с.
28. Харитонова И.А. Михеева В.Д. Microsoft Access 2000. – СПб.: БХВ – Санкт-Петербург, 1999. – 1088 с., ил.
29. Shekhar S., Chawla S. Spatial databases: a tour – Upper Saddle River, NJ: Prentice Hall, 2003. – 262 p.
30. Thew D. Essential Access 2000 fast: how to create databases using Access 2000. - London: Springer, 2000. – 208 p., ill.
31. Mueck T.A., Polaschek M.L. Index data structures in object-oriented databases – Boston: Kluwer Academic, 1997. – 177 p., ill.
32. Connolly T.M., Begg C.E. Database solutions: a step-by-step guide to building databases – Harlow: Addison-Wesley, 2004. - 528 p., ill.
33. Річчуті М. Microsoft укрепляет защиту базы даннях, 2004 // www.zdnet.ru.
34. Лелі С. Oracle и IBM сосредоточились на защите баз данных, 2001 // www.optim.su.

35. Корпорація Fujitsu Siemens Computers представляет новую версию пакета NetWorker Backup Suite. // www.vd.verysell.ru
36. Мацко П.В., Голубєв А.М. Вступ до геотроніки. Навчальний посібник. Херсон, ХДУ, 2006.–100с.
37. Плоткін С.Я. Програмне забезпечення ГІС. Частина 1. Початок роботи з ArcView GIS 3.x - Херсон, Вид-во ХДУ, 2006. - 32 с.
38. Морозов В.В. ГІС в управлінні водними і земельними ресурсами. Навчальний посібник. – Херсон, Вид-во ХДУ, 2006. - 91 с.
39. Управління ІТ вищих навчальних закладів: як інформаційні технології допомагають зробити управління ефективним / О.В. Співаковський, Д.Є. Щедролосьєв, Я.Б. Федорова та ін.: Методичний посібник. - Херсон: Айлант, 2006. – 356 с.: іл..
40. Multiply control of your IT world // www.intel.com.
41. Лисогоров К.С. Математичне моделювання і створення автоматизованих систем управління в зрошуваному землеробстві: монографія. - Херсон: Айлант, 2003. – 184 с.

ДОДАТКИ

Можливі варіанти баз даних для Гідромеліорації

Таблиця А.1 - Забезпеченість (%) річної суми опадів у Кримському Присивашші за 1977 - 1996 рр.*

Роки	Опади, мм	Р, %	Роки	Опади, мм	Р, %
1977	438	32,0	1987	428	36,0
1978	408	48,0	1988	471	20,0
1979	286	80,0	1989	396	52,0
1980	489	12,0	1990	239	96,0
1981	452	28,0	1991	413	40,0
1982	381	56,0	1992	464	24,0
1983	471	16,0	1993	270	84,0
1984	354	68,0	1994	307	76,0
1985	374	60,0	1995	409	44,0
1986	337	72,0	1996	364	64,0

* - за даними В.В. Морозова, Д.О. Ладичука

Таблиця А.2 - Показники для визначення меліоративної ефективності горизонтального дренажу (нормативна база даних)*

Показники меліоративної ефективності (посилання на літературне джерело)	Одиниця вимірю-вання	Оцінка роботи дренажу за ефективністю			
		висока	задовільна	низька	незадовільна
Середньорічний модуль дренажного стоку (Укргіпроводгосп,1975)	л/с з 1 га	0,05-0,15 >0,15	0,01-0,15	0,01-0,05	<0,01
Співвідношення річного дренажного стоку до водоподання (Д.М. Кац,1976)	%	>30	15-30	0-15	0
Зміна мінералізації дренажних вод (Укрдіпроводгосп,1975)	г/дм^3	зниження	зниження	стабільна, підвищення	стабільна, підвищення
Співвідношення об′єму солей, що відводяться дренажним стоком до об′єму солей, що поступають з поливними і атмосферними водами (Укрдіпроводгосп,1975)	б/р	1,5	1,2-1,5	1,0-1,2	1,0
Зміна засоленості грунтів шару 0-100 см (Укрдіпроводгосп,1975)	%	зниження	зниження	стабільна, підвищення	стабільна, підвищення
Зміна урожайності (Укрдіпроводгосп,1975)	т/га	ріст, стабільна	стабільна	зниження	зниження
Показник використання земель Еф (Б.В. Лютаєв, 1978)	б/р	>1	≥1	≤1	<1

* - за даними В.В. Морозова

Таблиця А.3 - Вміст гіпотетичних солей у дренажній воді на дослідно-виробничій ділянці у Кримському Присивашші*

Роки	Мінералізація, г/дм^3	Гіпотетичні солі, %						% токсичних солей
		$Ca(HCO_3)_2$	$CaSO_4$	Na_2SO_4	$MgSO_4$	NaCl	$MgCl_2$	
B** = 240 м								
1977-1981	14,28	3,9	16,6	22,0	-	31,2	26,3	79,5
1982-1989	12,87	5,2	16,0	26,8	-	27,5	24,5	78,8
1990-1996	7,20	3,5	25,9	31,4	-	29,4	9,8	70,6
середнє	12,16	4,2	19,5	26,7	-	29,4	20,2	76,3
B = 300 м								
1977-1981	12,60	4,0	17,0	27,9	-	25,2	25,9	79,0
1982-1989	11,38	5,1	17,6	32,0	-	20,3	25,0	77,3
1990-1996	7,48	4,4	25,6	35,2	4,3	7,5	23,0	70,1
середнє	10,60	4,5	20,1	31,7	1,4	17,7	24,6	75,5
B = 400 м								
1977-1981	12,82	4,3	17,5	24,6	-	28,3	25,3	78,2
1982-1989	11,54	5,6	16,0	35,8	3,0	18,0	21,6	78,4
1990-1996	8,08	3,6	25,3	27,4	0,9	15,1	27,7	71,1
середнє	11,05	4,4	19,6	29,3	1,3	20,5	24,9	76,0

* - за даними В.В. Морозова, Д.О. Ладичука

** - міждренна відстань.

Таблиця А.4. - Статистичні характеристики залежності вмісту основних іонів від суми солей (S, %) у грунті**

№ об'єкту*	Рівняння регресії	Коефіцієнт кореляції r та його похибка
1	HCO_3^-= 0,014S + 0,603	0,025 ± 0,058
2	HCO_3^-= 10,053S – 0,308	0,86 ± 0,032
3	HCO_3^-= 0,147S + 0,893	0,127 ± 0,148
1	Cl^-= 2,058S + 0,209	0,524 ± 0,042
2	Cl^-= 0,012S + 0,155	0,016 ± 0,123
3	Cl^-= 2,007S – 0,01	0,71 ± 0,075
1	SO_4^{2-}=13,555S – 0,968	0,978 ± 0,003
2	SO_4^{2-}=2,535S + 0,295	0,355 ± 0,108
3	SO_4^{2-}= 13,329S – 1,28	0,981 ± 0,006
1	Ca^{2+}= 7,061S – 0,460	0,877 ± 0,013
2	Ca^{2+}= 1,016S + 0,361	0,309 ± 0,111
3	Ca^{2+}= 2,06S + 0,078	0,8 ± 0,054
1	Mg^{2+}= 3,400S – 0,011	0,874 ± 0,014
2	Mg^{2+}= 0,169S + 0,487	0,028 ± 0,123
3	Mg^{2+}= 2,72S + 0,099	0,924 ± 0,022
1	Na^++K^+= 5,038S + 0,348	0,771 ± 0,024
2	Na^++K^+= 14,086S – 0,971	0,916 ± 0,020
3	Na^++K^+= 10,225S – 0,341	0,969 ± 0,009

* 1 – Кримське Присивашшя (зрошення + дренаж),
2 – Асканія – Нова (зрошення)
3 - Асканія – Нова (цілина)

** - за даними В.В. Морозова, Д.О. Ладичука

НАВЧАЛЬНО – МЕТОДИЧНЕ ВИДАННЯ

Ладичук Дмитро Олександрович – кандидат сільськогосподарських наук, доцент кафедри ГІС - технологій Херсонського державного аграрного університету.

Пічура Віталій Іванович – асистент кафедри ГІС - технологій Херсонського державного аграрного університету.

Бази геоінформаційних даних

Навчальний посібник

За редакцією професора В.В. Морозова

ISBN 966

Технічний редактор – Яковлєва Н.І.

Підписано до друку 14.02.2007 р. Формат 60 х 84 1/16. Друк ризографія.
Гарнітура Arial. Ум. друк. арк. 5,2. Наклад 100 примірників.
Видавництво Херсонського державного університету.