МІЖРЕГІОНАЛЬНА
АКАДЕМІЯ УПРАВЛІННЯ ПЕРСОНАЛОМ

МАУП

МЕТОДИЧНІ МАТЕРІАЛИ ЩОДО ЗАБЕЗПЕЧЕННЯ САМОСТІЙНОЇ РОБОТИ СТУДЕНТІВ

з дисципліни

“СУЧАСНІ СИСТЕМИ ПРОГРАМУВАННЯ БАЗ ДАНИХ”

(для бакалаврів)

Київ
ДП «Видавничий дім «Персонал»
2009

Підготовлено доцентом кафедри прикладної математики та програмування
Н. М. Москальковою

Затверджено на засіданні кафедри інформатики
та інформаційних технологій (протокол № 19 від 29.02.08)

Схвалено Вченою радою Міжрегіональної Академії управління персоналом

Москалькова Н. М. Методичні матеріали щодо забезпечення самостійної роботи студентів з дисципліни "Сучасні системи програмування баз даних" (для бакалаврів). — К.: ДП «Видавничий дім «Персонал», 2009. — 52 с.

Методична розробка містить пояснювальну записку, тематику самостійної роботи з дисципліни та список літератури.

ПОЯСНЮВАЛЬНА ЗАПИСКА

Використання баз даних є однією із характерних рис більшості сучасних інформаційних систем. Бази даних є тим ядром, навколо якого і будується інформаційна система будь-якого підприємства. Тому теорії створення та практиці використання баз даних приділяється достатня увага на протязі всього періоду функціонування інформаційних систем. Тривалий час основним типом були реляційні бази даних, які на сьогодні вже вважаються класичними. Проте розвиток інформаційних систем поставив перед сучасними базами даних завдання, вирішення яких неможливе в межах використання тільки реляційних баз даних. Крім класичних завдань, сучасні бази даних повинні забезпечувати багатомашинну обробку та зберігання великих обсягів інформації, оперативний аналіз даних, інтеграцію із мережею Інтернет, розмежування доступу користувачів до інформації, захист інформації під час її передачі мережею. Хоча на практиці і використовують велику кількість різноманітних баз даних, але для більшості з них існує велика кількість спільних рис і як для розробки, і для використання. Це дає можливість вивчати сучасні бази даних і відповідне прикладне та системне програмне забезпечення на прикладах які попри свою новизну вже стали класичними.

Актуальною є підготовка спеціалістів, які знають основні характеристики сучасних баз даних та методику їх інтеграції в інформаційні системи, володіють засобами програмування і клієнтської, і серверної частини СУБД, вміють ефективно використовувати всі можливості сучасних баз даних, мають достатню кваліфікацію для проектування, розробки та використання перспективних баз даних. Саме для підготовки таких спеціалістів і призначена дисципліна "Сучасні системи програмування баз даних".

Метою вивчення дисципліни "Сучасні системи програмування баз даних" є ознайомлення студентів із сучасними системами програмування баз даних, а також здобуття практичних навичок проектування баз даних та побудови систем управління базами даних та розробки баз даних в архітектурі клієнт-сервер.

Завданням курсу є поглиблення теоретичних знань, що необхідні для вирішення задач автоматизації обробки інформації у різних предметних областях, а також оволодіння практичними навичками використання та проектування систем управління базами даних, розробки програмних засобів збереження та маніпулювання даними.

Під час вивчення дисципліни передбачається систематична практична робота студентів і під керівництвом викладача, і самостійно. Предметом вивчення курсу "Сучасні системи програмування баз даних" є реляційна модель даних, засоби маніпулювання реляційними базами даних та засоби їх створення.

Після вивчення дисципліни студент повинен *знати*:

- тенденції та перспективи розвитку систем управління базами даних;
- технології збереження, пошуку та обробки інформації, які використовуються в сучасних системах управління базами даних;
- основні концепції роботи бази даних в архітектурі клієнт-сервер;
- правила розробки структури баз даних та створення прикладного програмного забезпечення з використанням сучасних систем управління базами даних;
- принципи побудови та технологію проектування баз даних;
- мову побудови запитів SQL.

Після вивчення дисципліни студент має *володіти* такими *вміннями та навичками*:

- проектування інформаційних систем та баз даних засобами *MS Access*, *MS SQL Server*, *Oracle Database Express Edition*;
- розробки інформаційних систем в архітектурі клієнт-сервер;
- створення програмного забезпечення для доступу до баз даних у сучасних середовищах візуального програмування;
- здійснення аналізу даних засобами сучасних систем управління базами даних.

Для розуміння тематики курсу "Сучасні системи програмування баз даних" студенти мають володіти базовими знаннями з основ програмування та алгоритмічних мов, володіти мовами об'єктно-орієнтованого програмування, мати досвід використання систем та інструментальних засобів програмування. Підсумкова перевірка знань студентів передбачена у формі іспиту. Для вивчення конкретних навчальних тем треба використовувати рекомендовану літературу з поданого у програмі списку.

Самостійна робота студентів є надзвичайно важливою складовою підготовки спеціалістів з напряму "Комп'ютерні науки". Теоретичний матеріал з програмування баз даних потребує багаторазового підкріплення практичними прикладами. Студенти мають здобути навички

самостійного виконання усіх етапів розробки програмного забезпечення (проектування, створення, тестування тощо). Це вимагає від студента систематичного виконання практичних завдань протягом семестру та підготовки до кожного практичного заняття. Студент, який хоче якомога краще оволодіти професією, має добре розуміти: на занятті викладач подає основи знань, навчає, як учити, виділяє ті ключові істини дисципліни, які пробуджують у молодої людини потяг до поглиблення й удосконалення знань. Лише постійне самостійне навчання дає можливість ближче підійти до вершини знань певної галузі, оволодіти такою сумою знань і вмінь, які б дали змогу заявити про себе як про професіонала.

Самостійна робота з дисципліни "Сучасні системи програмування баз даних" має такі складові і форми їх оцінювання:

- опрацювання програмного матеріалу зі змістового модуля (за матеріалами конспекту і підручників) та оцінка результатів під час проміжного контролю;
- підготовка та власне аудиторна робота під час практичних і лабораторних занять; результати оцінюються під час поточного контролю;
- виконання самостійних робіт у формі есе, рефератів з конкретних проблем та складання письмових звітів на електронних або паперових носіях та усних доповідей;
- виконання практичних завдань до змістовного модуля та оцінка результатів під час проміжного контролю;
- виконання письмової контрольної роботи або тестування;
- підготовка до складання іспиту.

Навчальний час, відведений для самостійної роботи, регламентується робочим навчальним планом і повинен згідно з Болонською декларацією становити не менше 50 % загального обсягу навчального часу студента, відведеного для вивчення конкретної дисципліни.

Самостійна робота студента під час лекції. Належне ведення конспекту під час лекції сприяє збереженню необхідної інформації та дає студенту змогу в подальшому проаналізувати її. При підготовці до практичних занять студент має спиратися на складений ним конспект лекції. При опрацюванні матеріалу лекції слід порівнювати законспектований матеріал з планом практичного заняття, що міститься у методичних матеріалах для практичних занять або у навчально-методичному комплексі. Якщо у конспекті бракує матеріалу з окремих питань лекції або недостатньо розкриті деякі питання практичного

заняття, або вони винесенні на самостійне опрацювання, студент повинен звернутися до рекомендованих підручників, навчальних посібників і відповідних методичних матеріалів.

Вивчення навчального матеріалу за підручниками, навчальними посібниками, методичними вказівками, опрацювання матеріалу за першоджерелами, науковою і спеціальною літературою. При роботі з цими джерелами студент насамперед повинен ознайомитись з їх змістом, щоб визначити чи необхідно опрацьовувати це джерело, чи має воно відношення до навчального курсу, що вивчається, і тільки після цього визначити послідовність опрацювання, відібрати необхідний для вивчення матеріал з цього джерела (глави, розділи тощо). При вивченні матеріалу необхідно аналізувати прочитане, порівнюючи з прослуханою та законспектованою лекцією, робити логічні висновки, позначати незрозумілі положення з метою їх подальшого з'ясування на практичному занятті. Бажано відпрацювати зручну для себе певну систему позначень (позначки на полях конспекту, підкреслення маркерами різних кольорів, доповнення конспекту альтернативними формулюваннями та посиланнями на інші джерела тощо) та фіксації опрацьованого матеріалу.

Робота з бібліотечними фондами та дистанційними джерелами з метою пошуку необхідної інформації. Сервіси мережі *Internet* надають унікальні можливості знаходження літературних джерел у географічно віддалених фондах та архівах, а також шляхом участі у мережевих конференціях, де можна отримати відповіді та поради щодо питань з розшукуваної інформації. Серед пошукових систем існують і обширні з тематики метапошукові системи, і вузькоспеціалізовані. Найбільш відомі з них: http://www.google.com, http://www.altavista.com, http://www.askjeeves.com, http://www.lycos.com, http://www.sciseek.com, http://www.msn.com, http://meta.ua http://www.rambler.ru, http://www.yandex.ru, http://www.aport.ru, http://www.metabot.ru, http://newsgroups.langenberg.com, uk.wikipedia.org, www.bukinist.agava.ru.

Матеріали щодо методів підвищення ефективності пошуку інформації в *Internet* містяться у статях: http://www.yandex.ru/info/search.html, http://www.searchengines.ru/, http://www.zodchiy.ru/links/search/, http://www.citforum.ru/internet/search/index.shtml, http://websearch.report.ru/, http://www.kokoc.com/search-engines/index.shtml, http://www.zhurnal.ru/search-r.shtml.

ТЕМАТИКА САМОСТІЙНОЇ РОБОТИ з дисципліни "СУЧАСНІ СИСТЕМИ ПРОГРАМУВАННЯ БАЗ ДАНИХ"

Змістовий модуль І. Теоретичні основи сучасних баз даних

Тема 1. Сучасні технології проектування інформаційних систем

Опрацювання програмного матеріалу (за матеріалами конспекту, підручників)

Представлення даних за допомогою ЕОМ. Концепція інтегрованої обробки даних: скорочення надлишковості, цілісність даних, незалежність прикладних програм від даних. Поняття предметної області. Поняття бази даних. Моделі даних: інфологічна модель даних, даталогічна модель даних, фізична модель даних. Поняття систем управління базами даних. Архітектура систем управління базами даних. Функції систем управління базами даних.

Реляційні інформаційні структури та системи управління базами даних. Загальні властивості відношень у реляційній базі даних. Ефективність використання пам'яті. Принципи нормалізації. Нормальні форми та їх види. Ключі та індекси. Цілісність даних.

Локальні та розподілені бази даних, їх переваги та недоліки. Однота багаторівневі типи архітектур баз даних. Поняття файл-сервер. Використання архітектури клієнт-сервер. Схема взаємодії її елементів. Сервер бази даних і сервер додатків. "Тонкий" та "товстий" клієнти. Використання браузеру як клієнта бази даних.

Функції систем керування базами даних: забезпечення секретності, захист цілісності даних, синхронізація, захист від відмов та відновлення. Особливості інструментальних засобів керування сучасних базам даних. Приклади та порівняльна характеристика функціональних можливостей сучасних баз даних.

Загальна характеристика СУБД *MS Access*. Основні технічні параметри промислової СУБД *Microsoft SQL Server*. Характерні відмінності від інших промислових баз даних. Модель зберігання даних. Зв'язок з клієнтськими програмами. Сервіси *Microsoft SQL Server*. Принципи використання програмних додатків в якості клієнтів *Microsoft SQL Server*. Принципи адміністрування *Microsoft SQL Server*. Задачі адміністрування. Використання графічних утиліт. Порядок

інсталяції. Вимоги до апаратного та програмного забезпечення. Початкова конфігурація *Microsoft SQL Server*. Визначення контексту та механізму безпеки.

Література [3; 6–8; 10; 11; 14–16; 19–21]

Основні визначення

Предметна область зазвичай розглядається як деяка сукупність реальних об'єктів (сутностей), кожний з яких має певний набір *властивостей*. Для відображення предметної області будують *інформаційну модель*, яка містить описи інформаційних сутностей (об'єктів) та тих їх властивостей (атрибутів), що є значимими для вирішення задачі. Загальний опис усіх інформаційних об'єктів та їх взаємозв'язків називається зовнішньою (інфологічною) моделлю предметної області. Сучасна технологія роботи з базами даних передбачає, що доступ користувачів та прикладних програм до бази даних здійснюється за допомогою спеціальних засобів, які називають системою управління базою даних (СУБД). Розрізняють три рівня абстракції для представлення даних, які зберігаються у базі даних:

- зовнішня інформаційна модель даних;
- логічна модель даних (логічна схема збереження даних);
- фізична модель даних (фізична схема збереження даних).

Більшість сучасних СУБД використовують реляційну модель даних. У реляційній моделі предметна область представляється у вигляді сукупності взаємопов'язаних відношень (таблиць), кожне з яких описує деякий клас однотипних об'єктів предметної області. В реляційній таблиці, стовпці називаються *атрибутами* (*полями*), оскільки вони характеризують одну із властивостей (аспектів) об'єктів. Список назв усіх стовпців (атрибутів) називається *схемою відношення*.

Для однозначної ідентифікації рядків таблиці можна використовувати деяку підмножину її атрибутів, яка називається *ключем*. Ключ — це мінімальний набір атрибутів, за значенням яких можна однозначно ідентифікувати окремий екземпляр об'єкта (рядок у таблиці). Підмножина полів таблиці є ключем тоді і тільки тоді, коли виконуються такі умови:

1. Рядки таблиці мають відрізнятись значенням хоча б одного ключового атрибута (унікальність значень ключових атрибутів);

2. Жоден з атрибутів не може бути виключений з підмножини ключових атрибутів без порушення умови унікальності (мінімальність).

У реляційній моделі даних між таблицями встановлюють зв'язки, які відображають взаємозв'язки між відповідними інформаційними об'єктами предметної області. Зв'язок „первинний ключ — зовнішній ключ" породжує зв'язок типу „один до багатьох" між таблицями. Якщо зовнішній ключ є одночасно внутрішнім ключем, то зв'язок має тип „один до одного". В цьому випадку між векторами обох таблиць встановлюється взаємооднозначна відповідність.

Тематика практичних завдань

1. Спроектувати реляційну схему даних "Відправлення грошових переказів у гривнях".
2. Спроектувати реляційну схему даних "Відправлення грошових переказів у валюті".
3. Спроектувати реляційну схему даних "Облік використання фондів стаціонару медичної установи".
4. Спроектувати реляційну схему даних "Облік бюджету медичної установи".
5. Спроектувати реляційну схему даних "Облік призначень для лікування пацієнта та вартості лікування у медичних закладах санаторно-курортного типу".
6. Спроектувати реляційну схему даних "Облік процедур, що відпускаються у медичних закладах санаторно-курортного типу".
7. Спроектувати реляційну схему даних "Облік замовлень на використання виробничих ліній для таблетування різних лікарських форм".
8. Спроектувати реляційну схему даних "Аптечна база".
9. Спроектувати реляційну схему даних "Дистриб'юторська мережа іноземної фармацевтичної компанії".
10. Спроектувати реляційну схему даних "Нарахування заробітної плати дистриб'ютора іноземної мережі".

Питання для самоконтролю та співбесіди

1. Що таке реляційна база даних?
2. Поясніть відмінність понять "домен" та "тип даних".
3. Що таке відношення?
4. Що таке ключ?

5. Для чого використовуються зв'язки між таблицями?
6. Для чого виконують нормалізацію бази даних?
7. Дайте визначення першої нормальної форми.
8. Дайте визначення другої нормальної форми.
9. Дайте визначення третьої нормальної форми.
10. Дайте визначення четвертої нормальної форми.

Тема 2. Сучасні засоби проектування баз даних

Опрацювання програмного матеріалу
(за матеріалами конспекту, підручників)

Проектування баз даних засобами *MS Access*. Створення заголовка відношення. Типи даних *MS Access*. Призначення властивостей полів таблиці. Підстановка значень до поля таблиці. Використання властивостей вкладинки *Подстановка* для визначення способу підстановки значень. Використання засобу *Мастер подстановки*. Визначення зв'язків між таблицями у *MS Access*. Засіб *Схема данных*. Зміна типу зв'язку між таблицями.

Проектування баз даних засобами *Microsoft SQL Server*. Системні бази даних та бази даних користувачів. Об'єкти баз даних *Microsoft SQL Server*. Принципи використання об'єктів програмних додатків в якості клієнтів *Microsoft SQL Server*. Системні таблиці. Перегляд метаданих. Методика зберігання даних. Номенклатура та призначення файлів бази даних. Структура даних. Визначення трансакцій. Створення бази даних за допомогою утиліти *Server Enterprise Manage*r та запитів *Query Analyzer*. Визначення характеристик бази даних під час її створення.

Проектування баз даних засобами СУБД *Oracle*. Огляд редакцій СУБД *Oracle Database*. Об'єкти баз даних. Проектування бази даних. Настройка користувачів бази даних. Моніторинг під'єднань до бази даних. Перегляд та створення об'єктів бази даних. Побудова запитів. Створення представлень. Використання послідовностей. Основи роботи в *Oracle SQL*Plus*. Підключення до серверу бази даних. Проектування застосування бази даних. Огляд інструментів програмування застосування бази даних. Основи мови *PL/SQL*. Використання збережених процедур та тригерів.

Література [1–3; 6; 8–10; 13; 17; 18; 22; 28–30]

Основні визначення

Для створення нової бази даних СУБД Access необхідно при завантаженні середовища у вікні, що з'явиться, вибрати пункт *Новая база данных* та натиснути кнопку *OK*. Далі на запит СУБД необхідно вказати назву файла (розширення файла — *mdb*), в якому буде розміщено базу, після чого з'явиться головне вікно керування базою даних. Для створення реляційних таблиць в СУБД *MS Access* у головному вікні бази даних вибрати вкладинку *Таблицы*, натиснути кнопку *Создать*, в діалоговому вікні *Новая таблица* обрати режим *Конструктор* і натиснути кнопку *OK*. В результаті на екрані з'явиться вікно для конструювання таблиці. В першому стовпчику *Имя поля* необхідно ввести ім'я поля таблиці, яке далі можна використовувати для звернення до відповідних даних таблиці. Для кожного поля таблиці необхідно вказати тип даних, який буде зберігатися в цьому полі. Тип даних вибирається зі списку можливих типів даних:

- *Текстовый* — алфавітно-цифрові дані (обсягом до 255 байтів);
- *Поле MEMO* — алфавітно-цифрові дані (обсягом до 64000 байтів);
- *Числовой* — числові дані (обсягом до 8 байтів);
- *Дата/время* — дані у вигляді дати та часу (обсягом до 8 байтів);
- *Денежный* — дані про грошові суми з 4 знаками після коми;
- *Счетчик* — унікальне ціле число від 0 до 2147483 647;
- *Логический* — логічні дані (Істина або Хиба);
- *Поле объекта OLE* — об'єкти програмних продуктів Windows (до 1 Гбайт);
- *Гиперссылка* — URL адреса;
- *Мастер подстановок* — текстова інформація зі списку або поля зі списком.

Нижня частина вікна формування структури таблиці дозволяє для кожного заданого поля описати його додаткові властивості, а саме: *Размер поля*, *Формат поля*, *Маска ввода*, *Значение по умолчанию*, *Условие на значение*, *Сообщение об ошибке*, *Обязательное поле*, *Пустые строки*, *Индексированное поле* тощо.

У *MS SQL Server* бази даних зберігаються на жорсткому диску у файлах із розширенням *mdf*. Журнали трансакцій зберігаються у файлах з розширенням *ldf*. Модель бази даних містить такі стандартні об'єкти *SQL Server*:

- *Database Users* — користувачі бази даних;

- *Database Roles* — ролі бази даних;
- *Tables* — таблиці;
- *Views* — представлення, які дозволяють відображати інформацію, що вибирається з таблиць;
- *Stored Procedures* — збережені процедури тощо.

Для створення бази даних за допомогою програми *SQL Server Enterprise Manager* необхідно виконати:

1) у контекстному меню *Databases* вікна *Object Inspertor* обрати команду *New Database...*;
2) визначити параметри нової бази даних та натиснути *OK*.

Для створення нової таблиці за допомогою програми *SQL Server Enterprise Manager* необхідно виконати:

3) у контекстному меню *Tables* розділу *Databases* вікна *Object Inspertor* обрати команду *New Table...*;
4) визначити назви полів та тип даних для кожного поля;
5) визначити властивості полів на вкладниці *Column properties*;
6) визначити ключ для таблиці за допомогою кнопки *Set Primary key* панелі інструментів *Table Designer* або діалогового вікна *Indexes/ Keys* (кнопка *Manage Indexes and Keys* панелі інструментів *Table Designer*);
7) визначити зв'язки з іншими таблицями за допомогою кнопки *Relationships* панелі інструментів *Table Designer*;
8) зберегти таблицю з деяким ім'ям.

Після створення таблиці для неї можна визначити обмеження та тригери, обравши команди *New constraint...* або *New trigger...* у контекстному меню відповідного розділу.

Типи даних, які можна використовувати для визначення полів таблиць в *MS SQL Server*:

1) цілочисельні: *int* в діапазоні від –231 до +231, *smallint* у діапазоні від -32768 до +32767, *tinyint* — додатні числа в інтервалі від 0 до 255;
2) числові з плаваючою крапкою: *real* — дійсні з точністю до 7 знаків, *float* — дозволяє представляти десяткові числа з точністю до 15 знаків, *decimal* [(p [, s])] и *numeric* [(p [, s])] (на відміну від *float* та *real*) дозволяють визначати допустимий інтервал значень (параметр *p*) та точність (параметр *s*) і використовувати для цього від 2 до 17 байт;

3) текстові типи даних : *char* (*n*) — *n* — розмір пам'яті, може містити до 8000 ANSI-символів, *nchar* може містити не більше 4000 символів Unicode, *varchar* — дозволяє зберігати значення змінної довжини, що містять до 8000 ANSI-символів, *nvarchar* — дозволяє зберігати значення змінної довжини, що містять не більше 4000 символов *Unicode*;
4) *datetime* и *smalldatetime* використовуються для представлення дати та часу;
5) *bit* використовуються для збереження двох значень: 0 або1;
6) *timestamp* — лічильник, значення поля такого типу не можна визначити явним способом;
7) *text* та *image* використовуються для збереження великих обсягів текстових та двійкових даних;
8) *money* дозволяє зберігати грошові значення тощо.

Створення бази даних в *Oracle* здійснюється за допомогою *Database Configuration Assistant*. При інсталяції *Oracle Database 10g Express Edition* база даних створюється автоматично. Таблиці в базі даних можуть бути створені різними способами, але в результаті застосування будь-якого візуального інструменту буде згенеровано SQL-команду, яку виконає сервер *Oracle*.

Для створення таблиці засобами *Oracle Database 10g Express Edition* необхідно:

1) обрати команду *Create Table* у меню *Object Browser;*
2) визначити ім'я таблиці у полі *Table Name*;
3) визначити імена полів у стовпці *Column Name* та типи даних у стовпці *Type*;
4) визначити властивості *Precision* , *Scale* та *Not Null* та натиснути *Next*;
5) визначити первинний ключ таблиці та послідовність, яка буде для нього використовуватись (*No Primary Key*, *Populated from a new sequence*, *Populated from an existing sequence*, *Not populated*), та натиснути *Next*;
6) визначити зовнішні ключі таблиці та натиснути *Next*;
7) визначити обмеження та натиснути *Finish*;
8) для створення таблиці натиснути *Create*.

Для зміни даних в таблиці натиснути кнопку *Data*. Для визначення полів таблиці в *Oracle* використовують такі типи даних:

- *CHAR* — символьний рядок фіксованої довжини (до 255 символів);
- *LONG* — символьний рядок змінної довжини розміром до 2 Гб;
- *DATE* — дозволяє зберігати рік, місяць, день, години, хвилини та секунди;
- *LONG RAW* — бінарні дані змінної довжини розміром до 2 Гб;
- *NUMBER* — число як з фіксованою, так і з плаваючою комою;
- *VARCHAR2* — символьний рядок змінної довжини від 1 до 4000 символів;
- *VARRAY* — складений тип даних, який дозволяє зберігати набір однотипних даних, що ідентифікуються індексом (масив) тощо.

Тематика практичних завдань

1. Описати структуру таблиць бази даних *Борей*.
2. Дати загальну характеристику схеми *HumanResources* бази даних *AdventureWorks* у СУБД *MS SQL Server*.
3. Дати загальну характеристику схеми *Person* бази даних *AdventureWorks* у СУБД *MS SQL Server*.
4. Дати загальну характеристику схеми *Production* бази даних *AdventureWorks* у СУБД *MS SQL Server*.
5. Дати загальну характеристику схеми *Sales* бази даних *AdventureWorks* у СУБД *MS SQL Server*.
6. Дати загальну характеристику схеми *Purchasing* бази даних *AdventureWorks* у СУБД *MS SQL Server*.
7. Описати типи даних, властивості полів, ключі, індекси та обмеження таблиці *Address* схеми *Person* бази даних *AdventureWorks* у СУБД *MS SQL Server*. Додати до таблиці рядок даних.
8. Описати типи даних, властивості полів, ключі, індекси та обмеження таблиці *ProductPhoto* схеми *Production* бази даних *AdventureWorks* у СУБД *MS SQL Server*. Додати до таблиці рядок даних.
9. Описати типи даних, властивості полів, ключі, індекси, обмеження та тригери таблиці *Employee* схеми *HumanResources* бази даних *AdventureWorks* у СУБД *MS SQL Server*. Додати до таблиці рядок даних.

10. Описати типи даних, властивості полів, ключі, індекси, обмеження та тригери таблиці *Store* схеми *Sales* бази даних *AdventureWorks* у СУБД *MS SQL Server*. Додати до таблиці рядок даних.
11. Описати типи даних, властивості полів, ключі, індекси, обмеження та тригери таблиці *WorkOrder* схеми *Purchasing* бази даних *AdventureWorks* у СУБД *MS SQL Server*. Додати до таблиці рядок даних.
12. Дати загальну характеристику схеми *HR* (*Human Resources*) СУБД *Oracle Database 10g Express Edition*.
13. Описати типи даних, властивості полів, ключі, індекси та обмеження таблиці *COUNTRIES* схеми *HR* СУБД *Oracle Database 10g Express Edition*. Додати до таблиці інформацію про Україну.
14. Описати типи даних, властивості полів, ключі, індекси та обмеження таблиці *DEPARTMENTS* схеми *HR* СУБД *Oracle Database 10g Express Edition*. Додати до таблиці рядок даних.
15. Описати типи даних, властивості полів, ключі, індекси та обмеження таблиці *EMPLOYEES* схеми *HR* СУБД *Oracle Database 10g Express Edition*.
16. Опишіть тригери, які використовуються для таблиці *EMPLOYEES* схеми *HR* СУБД *Oracle Database 10g Express Edition*.
17. Опишіть зміни, які відбудуться при додаванні до таблиці *EMPLOYEES* схеми *HR* СУБД *Oracle Database 10g Express Edition* нового рядка даних.
18. Опишіть, за яких умовах не можна внести зміни у таблицю *EMPLOYEES* схеми *HR* СУБД *Oracle Database 10g Express Edition* нового рядка даних.
19. Описати типи даних, властивості полів, ключі, індекси та обмеження таблиці *LOCATIONS* схеми *HR* СУБД *Oracle Database 10g Express Edition*. Додати до таблиці рядок даних.
20. Описати послідовності, які використовуються для схеми *HR* СУБД *Oracle Database 10g Express Edition*.
21. Описати послідовності, які використовуються для схеми *OE* (*Order Entry*) СУБД *Oracle Database 10g Express Edition*.
22. Провести інсталяцію та настройку СУБД *Microsoft SQL Server*.
23. Створити базу даних у СУБД *Microsoft SQL Server* для обліку товарів інтернет-магазину.
24. Створити базу даних у СУБД *Microsoft SQL Server* для обліку автомобілів на автостоянці.

25. Створити базу даних у СУБД *Microsoft SQL Server* для обліку кредитування на навчання.
26. Створити базу даних у СУБД *Microsoft SQL Server* для обліку кредитування на придбання квартири.
27. Провести інсталяцію на настройку СУБД *Oracle Database 10g Express Edition.*
28. Створити базу даних у СУБД *Oracle Database 10g Express Edition* для обліку кредитування на придбання квартири з щомісячною виплатою.
29. Створити базу даних у СУБД *Oracle Database 10g Express Edition* для обліку товарів інтернет-магазину.
30. Створити базу даних у СУБД *Oracle Database 10g Express Edition* для обліку автомобілів на автостоянці.

Питання для самоконтролю та співбесіди

1. Перерахуйте основні об'єкти бази даних Microsoft SQL Server.
2. Яким чином можна переглянути метадані в базі даних Microsoft SQL Server?
3. Що таке системні бази даних Microsoft SQL Server?
4. Що таке користувацькі бази даних Microsoft SQL Server?
5. Яким чином можна використати програмні додатки в якості клієнта бази даних Microsoft SQL Server?
6. Перерахуйте сервіси бази даних Microsoft SQL Server.
7. Як за допомогою запитів Query Analyzer створити базу даних Microsoft SQL Server?
8. Як за допомогою Server Enterprise Manager створити базу даних Microsoft SQL Server?
9. Перерахуйте завдання керування базами даних Microsoft SQL Server?
10. Як провести перегляд параметрів бази даних Microsoft SQL Server?
11. Як змінити параметри бази даних Microsoft SQL Server після її створення?
12. Які параметри бази даних Microsoft SQL Server можна встановити під час її створення?
13. Як визначити параметри бази даних Microsoft SQL Server під час її створення?

14. Які параметри бази даних Microsoft SQL Server можна змінити після її створення?
15. Функціональні можливості утиліти Server Enterprise Manager бази даних *Microsoft SQL Server*?
16. Які цілочисельні типи даних можна використовувати у таблицях СУБД *Microsoft SQL Server*?
17. Які текстові типи даних можна використовувати у таблицях СУБД *Microsoft SQL Server*?
18. Які типи даних можна використовувати у таблицях СУБД *Microsoft SQL Server*?
19. Як представляється дата та час у таблицях СУБД *Microsoft SQL Server*?
20. Що означає властивість типів даних *Length*?
21. Що означає властивість типів даних *Precision*?
22. Що означає властивість типів даних *Scale*?
23. Для чого використовується обмеження *PRIMARY KEY* ?
24. Для чого використовується обмеження *FOREIGN KEY* ?
25. Як створити таблицю засобами *Oracle Database 10g Express Edition*?
26. Які об'єкти можна створювати засобами *Oracle Database 10g Express Edition*?
27. Які цілочисельні типи даних можна використовувати у таблицях СУБД *Oracle Database 10g Express Edition*?
28. Які текстові типи даних можна використовувати у таблицях СУБД *Oracle Database 10g Express Edition*?
29. Які типи даних можна використовувати у таблицях СУБД *Oracle Database 10g Express Edition*?
30. Як представляється дата та час у таблицях СУБД *Oracle Database 10g Express Edition*?

Тема 3. Аналіз даних засобами мови SQL

Опрацювання програмного матеріалу
(за матеріалами конспекту, підручників)

Запити. Класифікація запитів. Запити на вибірку. Вибірка даних із декількох таблиць. Способи об'єднання запитів з декількох таблиць. Повне об'єднання. Внутрішнє та зовнішнє об'єднання. Ліве зовніш-

нє об'єднання та праве зовнішнє об'єднання. Зміна типу об'єднання таблиць.

SQL і управління реляційними базами даних. Використання *SQL* для побудови і наповнення бази даних. Оператор *SELECT*. Сортування, видалення інформації, що повторюється, використання спеціальних функцій для обчислень. Групування даних і побудова звітів. Обробка невизначених значень. Об'єднання таблиць, аналіз даних. Запити SQL, структуровані запити.

Визначення умов відбору записів за допомогою логічних виразів. Логічні оператори *AND*, *OR*, *BETWEEN*, IN, *LIKE*. Правила побудови складних логічних умов за декількома полями запиту. Обчислення у запитах. Обчислювальні поля. Правила побудови обчислювальних полів. Групування даних у запитах та використання агрегатних функцій. Використання підзапитів. Обчислення підсумкових значень. Перехресні запити. Запити-дії. Запити на оновлення даних. Запити на знищення записів. Запити на додавання записів. Запити на створення нової таблиці.

Література [1; 2; 4–8; 11; 14; 16; 27–30]

Основні визначення

Запит — це об'єкт, за допомогою якого здійснюється перегляд, зміна та аналіз бази даних так, як це визначено користувачем. Результат виконання запиту має вигляд таблиці та називається динамічним набором записів. У запитах часто використовуються арифметичні та логічні вирази для обчислення значень та визначення умов відбору записів.

Найчастіше застосовуються запити на вибірку деякої інформації з однієї або декількох взаємопов'язаних таблиць. Результатом виконання такого запиту є набір фактичних даних, що задовольняються визначеним умовам. Запити на вибірку здійснюються за допомогою оператора *SELECT*.

SELECT <предикат> <вирази над іменами стовбців>
FROM <список таблиць звідки здійснюється вибірка даних>
WHERE <умова вибірки>
GROUP BY <список стовбців поля для групування записів>
HAVING <умова вибірки в межах групи >
(використовується тільки якщо є розділ GROUP)
ORDER BY <ім'я стовбця ASC | DESC >
Предикат приймає значення:

ALL по замовченню,

DISTINCT — якщо з результату треба видалити рядки, що повторюються,

TOP — якщо потрібні верхні *n* рядків таблиці,

TOP n PERCENT — якщо потрібні верхні *n* % рядків таблиці.

Вирази над іменами стовбців:

1) скалярний вираз — може складатись з імен полів таблиць, що входять до розділу *FROM* (* — для позначення всіх полів), з простих функцій, що повертають скалярні значення;
2) агрегатні функції: *COUNT*, *MAX*, *MIN*, *SUM*, *AVG*. Агрегатні функції обчислюються за всіма рядками, що задовольняють умовному виразу розділу *WHERE*. Якщо розділ *GROUP BY* присутній, то агрегатні функції обчислюються окремо для кожної групи, що визначена в розділі *GROUP BY*.

Відібрані у запиті записи можуть бути згруповані за допомогою виразу *GROUP BY* з метою обчислення агрегатних функцій для кожної групи записів. Наприклад, співробітників можна згрупувати за посадою і для кожної групи записів обчислити сумарну заробітну плату. Агрегатні функції, які найчастіше використовуються наведені у таблиці.

Функція	Результат
Sum	Сума значень групи записів
Avg	Середнє значення групи записів
Min	Найменше значення групи записів
Max	Найбільше значення групи записів
Count	Кількість значень групи записів (без врахування порожніх значень)

Наприклад, у запиті "*Продажи товаров* в 1997" бази даних "*Борей*" використовується групування за полями "*Категория*", "*Марка*", "*Квартал*" з сумуванням за полем "*ПродажиТоваров*".

Умовний вираз у розділі *WHERE* обчислюється для кожного рядка, до результату потрапляють лише записи, які задовольняють цю умову. Умовний вираз у розділі *WHERE* може містити підзапити. Розділ *HAVING* містить умовний вираз, що обчислюється для кожної групи,

яка утворюється згідно зі списком групування розділу *GROUP BY*. Ця умова може містити агрегатні функції, що обчислюються для кожної групи. Умова розділу *WHERE* використовується для визначення умов для рядків, умова розділу *HAVING* використовується для визначення умов для груп. До логічних операторів належать:

- оператори *AND* та *OR*, які використовуються для об'єднання логічних виразів за допомогою логічних зв'язок *і* та *або*;
- логічний оператор *Not* дозволяє перевірити протилежну умову;
- оператор *BETWEEN* використовується для визначення належності значення виразу вказаному діапазону.

Синтаксис: вираз [*Not*] *Between* значення_1 *And* значення_2,

- Якщо значення виразу знаходиться у діапазоні між значення_1 та значення_2 (включно), то результатом є значення *True*; інакше — *False*.
- За допомогою оператора *In* здійснюється перевірка, чи збігається значення виразу з одним з елементів вказаного списку.

Синтаксис: вираз [*Not*] *In*(значення_1, значення_2, ...),

- Якщо вираз міститься у списку виразів значення_1, значення_2, **...,** то оператор повертає значення *True*; у протилежному випадку — значення *False*.
- За допомогою оператора *Like* можна задати шаблон, якому має відповідати значення текстового виразу.

Синтаксис: вираз [*Not*] *Like* шаблон

Для аргумента *шаблон* можна задавати повне значення або використовувати підставкові знаки.

У більшості випадків запити використовуються для відбору даних з декількох таблиць. При цьому запит виконуються у такій послідовності:

1) з кожної таблиці здійснюється відбір визначених полів або обчислення обчислюваних полів;
2) здійснюється об'єднання сформованих динамічних наборів записів відповідно до зв'язків, які визначені між таблицями;
3) до об'єднаного набору записів застосовуються умови відбору, операції групування, здійснюються обчислення агрегатних функцій тощо.

Операція об'єднання записів декількох таблиць або динамічних наборів записів є одною з найважливіших у реляційній алгебрі. За-

лежно від типу зв'язків, які встановлено між таблицями, розрізняють декілька способів об'єднання:

1) повне об'єднання (декартовий добуток), коли між таблицями зв'язки не встановлено;
2) внутрішнє об'єднання (*INNER JOIN*)- цей тип використовується за замовченням, коли одне з полів зв'язку є ключовим;
3) зовнішнє об'єднання.

Повне об'єднання (декартовий добуток) таблиць (відношень) є найпростішим способом об'єднання, коли всі записи першої таблиці комбінуються з усіма записами з другої таблиці. Таким чином, якщо у першій таблиці було *m* записів, а у другій *n* записів, то до результуючий набір буде містити *m*n* записів, кожна з яких містить усі обрані поля вихідних таблиць. Наприклад, повне об'єднання таблиць *Поставщики* та *Клиенты* буде містити 2639 = 91*29 записів. Такий спосіб об'єднання використовують, якщо необхідно сформувати перелік можливих комбінацій, що задовольняють певну умову.

Найчастіше у запиті на вибірку необхідно отримати об'єднання таблиць, між якими існує зв'язок типу "один до багатьох". При цьому лівою таблицею називається та з таблиць, у якій поле зв'язку є ключовим, інша таблиця називається правою таблицею. Таким чином, ліва таблиця знаходиться на стороні „один" зв'язку, а права таблиця — на стороні „багато". Внутрішнє об'єднання здійснюється з перевіркою умови на значення полів, за якими встановлено зв'язок: до результуючого набору додаються лише ті комбінації записів, для яких значення у пов'язаних полях і лівої і правої таблиць не є порожніми та збігаються.

Використовується також і зовнішнє об'єднання таблиць:

- ліве зовнішнє об'єднання (*LEFT JOIN*) — до результуючого набору додаються усі записи з першої (лівої) таблиці та з ними комбінуються лише ті записи другої (правої) таблиці, для яких значення у пов'язаних полях збігаються;
- праве зовнішнє об'єднання (*RIGHT JOIN*) — до результуючого набору додаються усі записи з другої (правої) таблиці та з ними комбінуються лише ті записи першої (лівої) таблиці, для яких значення у пов'язаних полях збігаються.

Відмінність внутрішнього та зовнішнього об'єднання полягає в тому, що при внутрішньому об'єднанні до результуючого набору додаються лише ті записи лівої (головної) таблиці, які мають відповідні записи у правій (підпорядкованій) таблиці. До результату зовнішньо-

го об'єднання будуть додані усі записи лівої (для лівого зовнішнього об'єднання) або правої (для правого зовнішнього об'єднання) таблиці, навіть якщо значення полів з іншої таблиці будуть порожніми. Наприклад, результат внутрішнього об'єднання таблиць *Клиенты* та *Заказы* буде містити перелік клієнтів (всього записів — 830), які справді здійснювали замовлення, тобто значення *КодКлиента* яких присутнє і в таблиці *Клиенты*, і в таблиці *Заказы*. До результуючого набору (всього записів — 832) виконання запиту з лівим зовнішнім об'єднанням будуть включені в тому числі і клієнти (*КодКлиента* — *PARIS*, *FISSA*), які не здійснювали замовлення (значення поля *КодЗаказа* для цих записів буде порожнім).

Так звані, запити-дії використовуються для внесення зміни в базу даних. За допомогою таких запитів можуть бути оновлені або видалені групи записів, додані дані в таблицю або створена нова таблиця.

Запити на додавання записів дозволяють додати окремі записи до таблиці:

INSERT INTO "table_name" ("column1", "column2", ...)
VALUES ("value1", "value2", ...)
або групу записів до таблиці
INSERT INTO "table1" ("column1", "column2", ...)
SELECT "column3", "column4", ... *FROM* "table2"

Запити на оновлення даних дозволяють оновити деякі дані:

UPDATE "table_name"
SET column_1 = [value1], column_2 = [value2]
WHERE {condition}

Запити на знищення даних дозволяють видалити деякі записи з таблиці

DELETE FROM "table_name"
WHERE {condition}

Запити на створення таблиці дозволяють створити таблицю у базі даних

CREATE TABLE "table_name"
("column 1" "data_type_for_column_1",
"column 2" "data_type_for_column_2",...)

Для створеної таблиці можна визначити первинний ключ, вторинні ключі, обмеження тощо.

Тематика практичних завдань

Запит 1. Для схеми даних *HR* (*Human Resources*) бази даних *Oracle Database 10g Express Edition* написати запит на вибірку інформації щодо співробітників, які працюють на фірмі більше 15 років.

Запит 2. Для схеми даних *HR* (*Human Resources*) бази даних *Oracle Database 10g Express Edition* написати запит на визначення середнього терміну працевлаштування співробітників для кожного відділу.

Запит 3. Для схеми даних *HR* (*Human Resources*) бази даних *Oracle Database 10g Express Edition* написати запит на визначення співробітника, який працює на фірмі найдовше.

Запит 4. Для схеми даних *HR* (*Human Resources*) бази даних *Oracle Database 10g Express Edition* написати запит на визначення для кожного співробітника кількості посад, на яких він працював на фірмі.

Запит 5. Для схеми даних *HR* (*Human Resources*) бази даних *Oracle Database 10g Express Edition* написати запит на визначення для кожного співробітника посади, на якій він працював на фірмі найкоротший термін.

Запит 6. Для схеми даних *HR* (*Human Resources*) бази даних *Oracle Database 10g Express Edition* написати запит на визначення переліку країн, у яких є філіали.

Запит 7. Для схеми даних *HR* (*Human Resources*) бази даних *Oracle Database 10g Express Edition* написати запит на визначення переліку країн, кількість співробітників у яких більше 5.

Запит 8. Для схеми даних *HR* (*Human Resources*) бази даних *Oracle Database 10g Express Edition* написати запит на визначення кількості співробітників у кожному регіоні.

Запит 9. Для схеми даних *HR* (*Human Resources*) СУБД *Oracle Database 10g Express Edition* написати запит на визначення для кожного менеджера переліку його підлеглих.

Запит 10. Для схеми даних HR (Human Resources) СУБД Oracle Database 10g Express Edition написати запит на визначення переліку співробітників, які працюють у Техасі.

Запит 11. Для схеми даних *Борей* написати SQL-запит для визначення того, на яку суму (з урахуванням знижки) офор-

мив замовлень товарів типу "*молочные продукты*" кожний із співробітників, що пропрацював на фірмі хоча б 15 років.

Запит 12. Для схеми даних *Борей* написати SQL-запит для визначення категорій товарів, що поставлялись у банках з *США* або з *Канади*.

Запит 13. Для схеми даних *Борей* написати SQL-запит для визначення десяти найдорожчих замовлень, що були зроблені у 1998 році.

Запит 14. Для схеми даних *Борей* написати SQL-запит для визначення середньої суми замовлень товарів (з урахуванням знижки та вартості доставки) для кожної країни замовника та типу товару (перехресний запит).

Запит 15. Для схеми даних *Борей* написати SQL-запит для визначення переліку товарів типу "*Кондитерские изделия*", які повторно були поставлені клієнтам з *Бразилії*.

Запит 16. Для схеми даних *Борей* написати SQL-запит для визначення марок товарів типу "*молочные продукты*" або "*Рыбопродукты*", які постачаються у пакетах.

Запит 17. Для схеми даних *Борей* написати SQL-запит для визначення того, на яку кількість замовлень обслужив кожний із співробітників, вік якого більше 35 років, та на яку суму (з урахуванням знижки).

Запит 18. Для схеми даних *Борей* написати SQL-запит для визначення 10 країн, до яких було поставлено поштою товарів на найбільшу суму (з урахуванням знижки та вартості доставки).

Запит 19. Для схеми даних *Борей* написати SQL-запит для визначення постачальників та співробітників з однакових країн.

Запит 20. Для схеми даних *Борей* написати SQL-запит для визначення того, на яку кількість замовлень обслужив кожний із співробітників, який поступив на роботу у другому півріччі 1993 року, та на яку суму (з урахуванням знижки).

Запит 21. Для схеми даних *Борей* написати SQL-запит для визначення марок товарів типу „*Кондитерские изделия*" або "*Мясо/птица*", які постачаються у коробках.

Запит 22. Для схеми даних *Борей* написати SQL-запит для визначення для кожного типу товарів розміру максимальної знижки.

Запит 23. Написати SQL-запит для визначення переліку товарів типу "*Молочные продукты*", які повторно були поставлені клієнтам з *Австрії*.

Запит 24. Для схеми даних *Борей* написати SQL-запит для визначення кількості та суми замовлень, які були розміщені у різні роки.

Запит 25. Для схеми даних *Борей* написати SQL-запит для визначення співробітників, які оформили замовлень товарів на суму (з урахуванням знижки) більше середнього.

Запит 26. Для схеми даних *Борей* написати SQL-запит для визначення кількості та суми замовлень, які доставлені кожним із видів доставки.

Запит 27. Для схеми даних *Борей* написати SQL-запит для визначення переліку клієнтів та категорій товарів, які вони не замовляли.

Запит 28. Для схеми даних *Борей* написати SQL-запит для визначення переліку марок товарів, які не доставляли до *США* та *Канади*.

Запит 29. Для схеми даних *Борей* написати SQL-запит для визначення постачальників та клієнтів з однакових міст.

Запит 30. Для схеми даних *Борей* написати SQL-запит для визначення того, на яку загальну суму замовлено товарів кожного типу до кожної країни клієнту (перехресний запит).

Запит 31. Для схеми даних *Борей* написати SQL-запит для визначення того, яка кількість товарів зберігається на складі для кожного типу товару та кожної країни постачальника (перехресний запит).

Запит 32. Для схеми даних *Борей* написати SQL-запит для створення таблиці *Таблиця1*, до якої додати записи про замовлення товарів (код замовлення, марка товару, категорія товару, назва постачальника та ціна товару), тип яких "*Кондитерские изделия*", ціна перевищує 2000 грн., та замовлення, які були розміщені у 1997 році.

Запит 33. Для схеми даних *Борей* написати SQL-запит для зміни у всіх записах таблиці *Таблица1* назви постачальника *Forots d'Erables* на *Forots Erables.*

Запит 34. Для схеми даних *Борей* написати SQL-запит для створення таблиці *Таблиця2*, до якої розмістити інформацію про товари (марка товару, категорія товару, назва постачальника та ціна товару), тип яких "*Фрукты*", ціна перевищує 2000 грн., та замовлення, які були розміщені у 1998 році.

Запит 35. Для схеми даних *Борей* написати SQL-запит для видалення з таблиці *Таблиця2* всіх записів про товари, ціна яких більше 2300 грн.

Питання для самоконтролю та співбесіди

1. Що таке запит?
2. Які види запитів існують у *MS Access*?
3. Як створити запит на основі фільтра?
4. Як створити запит на вибірку даних за допомогою майстра побудови простих запитів?
5. Що таке обчислюване поле?
6. Як формуються прості умови відбору записів?
7. Як відібрати 15 записів, що мають найбільші значення за деяким полем?
8. Як формуються складені умови відбору записів?
9. Як знайти унікальні значення деякого поля таблиці?
10. Як здійснюється сортування записів у запиті?
11. Які агрегатні функції можна використовувати в запитах?
12. Для чого використовується конструкція *SELECT*?
13. Для чого використовується службове слово *AS* ?
14. Для чого використовується службове слово *FROM* ?
15. Для чого використовується конструкція *UNION* ?
16. Для чого використовується конструкція *ORDER BY* ?
17. Для чого використовується конструкція *GROUP BY* ?
18. Як мовою *SQL* визначається поле, за яким здійснюється об'єднання таблиць?
19. Як мовою *SQL* визначається умова відбору записів?
20. Як мовою *SQL* визначається групування записів?
21. Що таке запит з параметром?
22. Як здійснюється побудова обчислювальних полів у запитах?

23. Наведіть приклади запитів з використанням логічного оператора *AND*
24. Наведіть приклади запитів з використанням логічного оператора *OR*.
25. Наведіть приклади запитів з використанням логічного оператора *BETWEEN*.
26. Наведіть приклади запитів з використанням логічного оператора *IN*.
27. Наведіть приклади запитів з використанням логічного оператора *LIKE*.
28. Як у запитах здійснюється групування даних?
29. Для чого використовуються перехресні запити?
30. Для чого використовуються запити-дії?
31. Назвіть типи запитів-дій.
32. Наведіть приклад запиту на створення таблиці.
33. Наведіть приклад запиту на додавання записів у таблицю.
34. Наведіть приклад запиту на видалення записів з таблиці.
35. Наведіть приклад запиту на оновлення записів у таблиці.

Змістовий модуль ІІ. Сучасні технології програмування баз даних в архітектурі клієнт-сервер

Тема 4. Забезпечення роботи інформаційної системи в мережі

Опрацювання програмного матеріалу (за матеріалами конспекту, підручників)

Механізми доступу до даних з клієнтських застосувань. Використання прикладного програмного інтерфейсу (*Application Programming Interface*, *API*) для доступу до даних серверної системи управління базами даних. Використання універсальних механізмів доступу до даних. Використання *Open Database Connectivity* (*ODBC*) для доступу до даних. Використання *OLE DB* для доступу до даних. Використання *ActiveX Data Objects* (*ADO*) для доступу до даних.

Планування роботи. Типи блокування. Встановлення таблиць для блокування. Блокування записів. Відміна блокування таблиць та записів. Використання сеансів роботи. Використання буферів при редагування даних. Використання трансакцій. Обробка мережевих помилок.

Загальні відомості про засоби захисту *MS Access*. Визначення пароля для бази даних. Створення *mde*-файлів. Використання майстра захисту. Настройка середовища *MS Access*. Визначення параметрів запуску програми у *MS Access*. Адміністрування бази даних. Стиснення та відновлення файлів *MS Access*.

Література [1; 2; 4–8; 11; 12; 14; 16; 27–30]

Основні визначення

Інформаційні системи, що використовуються архітектуру клієнт-сервер надають користувачам суттєві переваги. Зокрема, однією з найбільших переваг є зниження мережевого трафіку при виконанні запитів. Іншою важливою перевагою є можливість збереження бізнес-правил на сервері, що дозволяє уникнути дублювання коду у різних клієнтських застосуваннях, що використовують спільну базу.

Для доступу до даних у клієнтських застосуваннях частіше використовують універсальні механізми доступу до даних, які реалізуються у вигляді додаткових модулів (драйверів). Застосування, які створені у такий спосіб, легко модифікувати у разі необхідності зміни СУБД, причому модифікація стосується не зміни коду застосування, а лише настроювання доступу до даних. Найбільш поширеними механізмами доступу до даних є *Open Database Connectivity* (*ODBC*) та *ActiveX Data Objects* (*ADO*).

ODBC — універсальний механізм доступу до баз даних будь-якого формату, що вбудований до операційної системи *Windows*. Для того, щоб отримати доступ до даних за допомогою *ODBC*, спочатку треба створити так зване джерело даних *ODBC*, яке будуть використовувати програми для доступу до бази даних. Джерело даних містить відомості про те, де знаходиться файл бази даних та в якому форматі зберігається база даних. Настроювання джерела даних *ODBC* здійснюється за допомогою спеціальної програми *ODBC*-адміністратора (odbcad32.exe) у вікні "*Администратор источников данных ODBC*", яке можна викликати за допомогою команди "*Источники данных (ODBC)*" розділу *Администрирование* Панелі управління. Для того, щоб додати нове джерело даних необхідно натиснути кнопку "*Добавить*".

Фізично *ODBC* представляє собою набір динамічних бібліотек *DLL*, які обслуговують підключення та роботу з конкретним типом бази даних. При запиті на підключення до деякої заздалегідь описаної базі активізується деяка *DLL* — драйвер цього типу бази даних.

Звернення до бази даних відбувається за ім'ям джерела даних *ODBC* (*DSN* — data source name). *DSN* є об'явленням бази даних на цьому комп'ютері. Управління джерелами даних *ODBC* можна здійснювати програмно.

ADO (*ActiveX Data Objects*) — інтерфейс програмування застосувань для доступу до даних, розроблений компанією *Microsoft* та заснований на технології компонентів *ActiveX*. Об'єктна модель *ADO* складається з таких об'єктів: *Connection* — для підключення до віддаленого джерела даних, *Recordset* — набір рядків, що отримані з джерела даних, *Command* — використовується для виконання команди SQL-запитів з параметрами, *Record* — представляє один запис об'єкта *Recordset*, *Stream* –використовується для читанин та запису потокових даних, наприклад, документів XML або двійкових об'єктів, *Errors* представляє помилки, *Fields* — представляє стовпці таблиці бази даних тощо.

ADO.NET (*ActiveX Data Objects .NET*) є набором класів, які реалізують програмні інтерфейси для полегшення підключення до баз даних із застосування незалежно від особливостей реалізації конкретної системи управління базами даних та від структури самої бази даних. Цей механізм дозволяє також встановлювати доступ до даних незалежно від місця розташування бази даних (при розробці клієнт-серверного застосування). *ADO.NET* широко використовуються спільно з технологією web-програмування *ASP.NET*.

У *Microsoft Access* реалізовано такі механізми керування управління доступом до бази даних:

- кодування та декодування — файл бази даних стискається та стає недоступним для читання за допомогою службових програм та текстових редакторів;
- відображення та приховування об'єктів в вікні бази даних ;
- використання параметрів запуску — визначають настройки запуску застосування бази даних ;
- встановлення пароля для відкриття бази даних;
- використання захисту на рівні користувачів.

Захист на рівні користувачів дозволяє встановити різні рівні доступу до важливих даних та об'єктів у базі даних. Інформація щодо користувачів зберігається у файлі робочої групи.

Тематика практичних завдань

1. Створити джерело даних *ODBC* користувача для бази *HR* (*Human Resources*) СУБД *Oracle Database 10g Express Edition*.
2. Створити джерело даних *ODBC* користувача для бази *AdventureWorks* СУБД *MS SQL Server.*
3. Створити користувача *study* у СУБД *Oracle Database 10g Express Edition.*
4. Створити джерело даних *ODBC* користувача для бази даних *Борей*.
5. Реалізувати систему заходів для запобігання несанкціонованого доступу до бази даних *Борей*.
6. Для бази даних *Борей* створити групу користувачів *Клієнти*, для яких визначити доступ до об'єктів бази даних, де представлена інформація про наявні товари та заборонити доступ до об'єктів бази даних, де представлена інформація про службову інформацію торгової фірми *Борей.*
7. Для бази даних *Борей* створити групу користувачів *Відділ кадрів*, для яких визначити доступ до об'єктів бази даних, де представлена інформація про співробітників фірми та обсяг виконаної ними роботи та заборонити доступ до об'єктів бази даних, де представлена інформація про товари та клієнтів торгової фірми *Борей.*
8. Створити роль бази даних *Microsoft SQL Server.*
9. В базі даних *Microsoft SQL Server* створити скрипт для резервного копіювання бази даних при критичному розмірі журналу трансакцій.
10. Провести відновлення бази даних *Microsoft SQL Server* із резервної копії.
11. Створити прилад резервування бази даних *Microsoft SQL Server.*
12. Створити резервну копію журналу трансакцій бази даних *Microsoft SQL Server.*

Питання для самоконтролю та співбесіди

1. Які функції виконує *ODBC*?
2. Як створити джерело даних *ODBC*?
3. Які механізми доступу до даних використовуються при створенні клієнтських застосувань?
4. Охарактеризуйте використання технології *OLE DB* .
5. Охарактеризуйте використання технології *ADO.*

6. Які засоби захисту даних реалізовано у *MS Access*?
7. Як визначити пароль бази даних у *MS Access*?
8. Як створити *mde*-файл?
9. Які параметри запуску програми можна визначити засобами *MS Access*?
10. Як здійснити стиснення даних у *MS Access*?

Тема 5. Розробка інтерфейсу клієнта засобами Access

Опрацювання програмного матеріалу
(за матеріалами конспекту, підручників)

Форми. Класифікація форм. Призначення форм. Режими відображення форм. Використання форм для введення даних у режимі *Режим формы*. Проектування форм у режимі *Конструктор*. Відображення джерела даних за допомогою режиму *Режим таблицы*. Призначення звітів. Розділи звітів. Елементи управління, що використовуються у звітах.

Використання форм для введення і редагування даних. Панель інструментів *Режим формы*. Створення макета форми. Розділи форм. Елементи управління. Властивості елементів управління. Майстри побудови елементів управління. Майстер побудови списків, полів зі списками, майстер створення групи перемикачів, майстер створення кнопок.

Підпорядковані форми. Зв'язані форми. Представлення пов'язаних таблиць за допомогою підпорядкованих форм. Майстер побудови підпорядкованих форм. Створення підпорядкованої форми на основі декількох таблиць або запитів. Додавання підпорядкованої форми. Використання обчислювальних елементів управління у формах. Розрахунок підсумкових значень у групах записів підпорядкованої форми. Створення кнопкових форм. Використання засобу *Диспетчер кнопочных форм*. Побудова діалогових форм. Введення параметрів запитів за допомогою діалогових форм.

Сортування та групування даних у звітах. Зміна параметрів сортування та групування у діалоговому вікні *Сортировка и группировка*. Створення звітів за допомогою майстрів побудови звітів. Редагування звіту у режимі *Конструктор*. Обчислення у звітах. Обчислювальні поля за одним записом. Обчислювальні поля для груп записів. Використання властивості *Сумма с накоплением* для групи та для всіх записів звіту.

Література [3; 6; 8–10; 13; 17; 18]

Основні визначення

Форми є основою інтерфейсу користувача. Форми виконують дві важливих функції в базах даних:

1) дозволяють вводити та редагувати дані в таблицях та запитах (форми даних);
2) дозволяють здійснювати управління роботою програми (кнопкові та діалогові форми).

Підпорядкована форма — це форма, яка пов'язана з іншою (головною) формою за деякою сукупністю полів. Підпорядковані форми використовуються для представлення даних таблиць та запитів, які пов'язані відношенням "один до багатьох", причому головна форма представляє сторону "один", а підпорядкована форма — сторону "багато". Відображення даних у підпорядкованій формі синхронізується з головною формою таким чином, що у підпорядкованій формі відображаються тільки ті записи, які пов'язані з поточним записом головної форми.

Структура форми складається з таких розділів:

- *заголовок формы* — відображається у верхній частині форми, зазвичай використовується для розміщення тексту заголовка форми та інструкцій щодо роботи з формою;
- *примечание формы* — відображається у нижній частині форми, призначення аналогічне до заголовка форми;
- *область данных* — використовується для відображення даних.

Кнопкова форма — це форма, яка містить кнопки, при натисканні яких здійснюється виконання визначених на етапі розробки дій, таких як виконання запиту, відкриття звіту, виконання макросу тощо. Для створення кнопкових форм використовується *Диспетчер кнопочных форм* (*Сервис*, *Служебные программы*, *Диспетчер кнопочных форм*).

Звіт — це об'єкт, який призначено для представлення даних з бази у вигляді зручному для їх друку. Роздрукувати дані без обробки можна безпосередньо з таблиці (кнопка "*Печать*" на панелі інструментів). Звіти використовуються, коли перед друком необхідно згрупувати дані за деякими полями, обчислити проміжні та підсумкові значення, коли необхідно оформити колонтитули сторінок тощо. Джерелом даних для звіту є таблиці та запити. Зв'язок між звітом та джерелом даних, подібно до форм, здійснюється за допомогою елементів управ-

ління. Зміна структури звіту здійснюється у режимі *Конструктор*. У режимі конструктора звіт містить такі розділи:

- *Заголовок отчета* — вміст розділу виводиться лише на першій сторінці звіту;
- *Верхний колонтитул* — вміст розділу виводиться зверху кожної сторінки звіту;
- *Заголовок группы* — необов'язковий розділ, який використовується при групуванні даних, виводиться зверху кожної групи;
- *Примечание группы* — необов'язковий розділ, який використовується при групуванні даних, виводиться знизу кожної групи;
- *Область данных* — у цьому розділі розміщують елементи управління, які забезпечують зв'язок звіту з джерелом даних;
- *Нижний колонтитул* — вміст розділу виводиться знизу кожної сторінки звіту;
- *Примечание отчета* — вміст розділу виводиться лише наприкінці звіту.

Для відображення заголовка та приміток звіту, колонтитулів необхідно у вікні конструктора у меню "*Вид*" увімкнути перемикачі "*Колонтитулы*" та "*Заголовок/примечание отчета*". Зв'язок між формами та звітами та джерелом даних реалізується за допомогою е*лементів управління*, які розміщують в області даних. Для додавання елементів до форми використовується *Панель элементов*.

Тематика практичних завдань

1. Створити базу даних обліку інформації про співробітників фірми. У базі даних має бути така інформація про співробітника:
   - прізвище, ім'я, по батькові,
   - посада, оклад для визначеної посади, персональна надбавка для працівника, дата найму співробітника, робочий телефон,
   - номер паспорта, дата народження, місто проживання, адреса проживання, домашній телефон, фотографія,
   - назва філіалу, в якому працює співробітник, ідентифікаційний код філіалу, місто, де розташований філіал, адреса філіалу, телефон офіса.

   Створити форми даних :

- форму *Співробітник* для представлення інформації про співробітників фірми (поля розмістити на трьох вкладках *Дані про співробітника*, *Особисті дані*, *Фото*),
- форму *Філіали* для представлення інформації про філіали фірми,
- у формі *Філіали* розмістити підпорядковану форму *підпорядкована форма Співробітник* для представлення службової інформації про співробітників філіалу (за допомогою кнопки *Додаткова інформація* реалізувати відкриття форми *Співробітник* для обраного співробітника),
- форму *Посади* для представлення інформації про посади фірми,
- форму *Міста* для представлення переліку міст.

У формі *Співробітник* та *підпорядкована форма Співробітник* розмістити обчислювальне поле для заробітної плати, яка складається з окладу посади, надбавки за вислугу років — 50 грн за кожні 5 років служби та персональної надбавки.

У формі *Філіали* розмістити обчислювальне поле для представлення сукупного фонду заробітної плати філіалу.

Створити головну кнопкову форму, з якої здійснювати виклик форм *Філіали*, *Співробітник*, *Посади*, *Міста*. Заповнити базу даних інформацією про 7–10 філіалів з різних міст, у кожному із яких працює 7–10 співробітників (на 5–7 посадах).

Створити звіт *Кадри*, в якому навести перелік співробітників у кожному із філіалів (Назва та ідентифікаційний код філіалу, прізвище, ім'я, по батькові співробітника, посада, вік та стаж роботи, впорядкувати за прізвищем за зростанням), з нумерацією співробітників для кожного із філіалів. Визначити середній вік та стаж співробітників у кожному із філіалів та в цілому у фірмі.

Створити звіт *Зарплата* (параметром звіту виступає дата видачі), в якому навести перелік співробітників у кожному із філіалів (Прізвище, ім'я, по батькові, номер паспорта, заробітна плата, з урахуванням надбавок, податок (13 % на всю суму), сума на руки, впорядкувати за прізвищем за зростанням), з нумерацією співробітників для кожного із філіалів. Визначити загальний фонд заробітної плати у кожному філіалі та у фірмі в цілому.

Створити діалогову форму *Звіти* для друку звітів, у якій користувач обирає тип звіту *Кадри* або *Зарплата*, а також може обрати назву

філіалу, для якого буде створено звіт (якщо назву філіалу не обрано, то звіт формується для всіх філіалів). Якщо обрано тип звіту — *Зарплата*, то користувач має визначити місяць та рік видачі. Форма також містить кнопки *Просмотр* для перегляду звіту, *Друк* — мовлення для друку звіту та *Отмена* — для виходу у головну кнопкову форму. Реалізувати виклик форми *Звіти* із головної кнопкової форми.

2. Створити базу даних обліку інформації про поставки товарів на склад. У базі даних має бути така інформація про товар:
   - назва, категорія товару, ціна товару, одиниця виміру товару, назва постачальника, країна, місто та адреса постачальника, телефон постачальника та адреса Web-сторінки,
   - номер поставки, дата поставки товару, кількість товару у поставці, знижка при поставці, вартість доставки.

   Створити форми даних :
   - форму *Товари* для обліку інформації про товари,
   - форму *Постачальник* для обліку інформації про постачальників,
   - форму *Поставки* для обліку інформації про поставки та підпорядковану форму *Поставлено* для представлення інформації про товари, що надійшли згідно із визначеною поставкою (за допомогою кнопки *Інформація про постачальників* реалізувати відкриття форми *Постачальник* для всіх постачальників обраної поставки),
   - форму *Категорії* для представлення інформації про категорії товарів.

   У формі *Поставлено* розмістити обчислювальне поле, в якому обчислювати ціну продажу товару з урахуванням знижки.

   У формі *Поставки* розмістити обчислювальне поле для представлення суми поставки з урахуванням знижки та вартості доставки товару.

   Створити головну кнопкову форму, з якої здійснювати виклик форм *Товари*, *Постачальник*, *Поставки*, *Категорії*. Заповнити базу даних інформацією про 7–10 поставок, кожна з яких включає поставку 7–10 товарів (7–10 різних категорій). Поставки здійснюються 7–10 постачальниками.

   Створити звіт *Постачальники* (параметром звіту виступає дата), в якому навести перелік постачальників та перелік товарів, що поставлені кожним із постачальників (назва постачальника, адреса, те-

лефон, номер замовлення, дата, назва товару, вартість, впорядкувати за назвою за зростанням), з нумерацією товарів для кожної категорії. Визначити сумарну вартість поставлених товарів кожного із постачальників та частку (у процентах) сумарної вартості поставок кожним постачальником від загальної вартості поставлених товарів.

Створити звіт *Розрахунки* (параметром звіту виступає дата), в якому навести перелік поставок товарів за категоріями (назва, ціна, номер замовлення, дата, кількість, вартість, вартість з урахуванням знижки, вартість з урахуванням доставки, впорядкувати за вартістю за зростанням), з нумерацією товарів для кожної із категорій. Визначити загальну сплачену суму за кожною із категорій та всіх товарів у цілому.

Створити діалогову форму *Звіти* для друку звітів, в якій користувач обирає тип звіту *Постачальники* або *Розрахунки* та визначає дату. Якщо обрано тип звіту — *Розрахунки*, то користувач має визначити категорію товару, для якого формується звіт. Форма також містить кнопки *Просмотр* для перегляду звіту, *Друк* — для друку звіту та *Отмена* — для виходу у головну кнопочну форму. Реалізувати виклик форми *Звіти* із головної кнопкової форми.

3. Створити базу даних обліку інформації про клієнтів фірми. У базі даних має бути така інформація про клієнта:
   - назва клієнта, прізвище, ім'я, по батькові директора, контактний телефон, номер рахунка клієнта,
   - країна, місто та юридична адреса клієнта, ідентифікаційний код клієнта,
   - вид робіт, виконання яких замовив клієнт, опис замовлення, сума замовлення, знижка, дата замовлення, дата виконання замовлення, номер акта прийняття робіт,
   - сума оплати за замовлення (оплата може здійснюватись за кількома рахунками), дата оплати, метод оплати, номер платіжного рахунка.

   Створити форми даних :
   - форму *Клієнт* для представлення інформації про клієнтів фірми,
   - форму *Замовлення* для представлення інформації про замовлення фірми,
   - форму *Оплата* для представлення інформації про оплату, що надійшла на рахунок фірми,

- форму *ВидиРобіт* для представлення інформації про види робіт, що виконує фірма,
- у формі *Замовлення* розмістити підпорядковану форму *Оплата* для представлення інформації про здійснену оплату,
- у формі *Клієнт* розмістити підпорядковану форму *Замовлення* для представлення інформації про замовлення, які здійснив клієнт (за допомогою кнопки *Інформація про оплату* здійснити відкриття форми *Замовлення* для замовлень, які зробив клієнт).

У формі *Клієнт* розмістити обчислювальне поле *сума*, в якому обчислити загальну суму замовлень, які зробив клієнт (з урахуванням знижки).

У формі *Замовлення* розмістити обчислювальне поле *Сплачено* для представлення сукупної сплаченої суми клієнтом за цим замовленням та поле *Борг* для представлення заборгованості за цим замовленням.

Створити головну кнопкову форму, з якої здійснювати виклик форм *Клієнт*, *Замовлення*, *Оплата*, *ВидиРобіт*. Заповнити базу даних інформацією про 7–10 клієнтів, кожний з яких зробив 5–7 замовлень, а оплату здійснював за 3–5 операцій. Фірма виконує замовлення на 3–5 видів робіт.

Створити звіт *Клієнти*, в якому навести перелік клієнтів за кожним із видів замовлених робіт (назва клієнта, ідентифікаційний код, номер рахунка, робочий телефон, сума замовлення, сума, яку вже сплачено, сума боргу, впорядкувати за назвою за зростанням), з нумерацією клієнтів для кожного із видів замовлених робіт. Визначити середній розмір замовлення за кожним із видів робіт та загальний розмір боргу.

Створити звіт *Оплата* про кошти, що були сплачені за звітний період (параметром звіту виступає період, протягом якого здійснювалась оплата — початкова та кінцева дата). Вказати назву організації платника, дату замовлення, суму замовлення, суму оплати, дату оплати, метод оплати, номер платіжного рахунка, групування за датою оплати потижнево з нумерацією оплат для кожного із замовлень. Визначити загальну суму потижневої оплати за вказаний період, а також процентне співвідношення сплачених сум протягом кожного тижня.

Створити діалогову форму *Звіти* для друку звітів, в якій користувач обирає тип звіту *Клієнти* або *Оплата*, а також може обрати вид

замовлених робіт, для якого буде створено звіт (якщо вид замовлених робіт не обрано, то звіт формується для всіх видів замовлених робіт). Якщо обрано тип звіту — *Оплата*, то користувач має визначити початкову та кінцеву дату звітного періоду. Форма також має кнопки *Просмотр* для перегляду звіту, *Друк* — для друку звіту та *Отмена* — для виходу у головну кнопкову форму. Реалізувати виклик форми *Звіти* із головної кнопкової форми.

4. Створити базу даних обліку інформації про відліт літаків з аеропорту. У базі даних має бути така інформація про рейс:
   - номер рейсу, пункт призначення, пункт відльоту, дата та час відправлення, час прибуття, кількість відправлень на тиждень, вартість квитка першого класу, вартість квитка другого класу, вартість квитка третього класу, кількість проданих квитків першого класу, кількість проданих квитків другого класу, кількість проданих квитків третього класу, собівартість вильоту,
   - тип літака, кількість місць у літаку першого класу, кількість місць у літаку другого класу, кількість місць у літаку третього класу,
   - прізвище, ім'я, по батькові пілота, стаж пілота, категорія пілота, дата народження.

   Створити форми даних :
   - форму *Рейси* для представлення інформації про рейси,
   - форму *Відправлення* для представлення інформації про відліт літаків,
   - форму *Пілоти* для представлення інформації про пілотів, що обслуговують рейси,
   - форму *ТипиЛітаків* для представлення інформації про типи літаків,
   - форму *ПунктиОбслуговування* для представлення переліку пунктів призначення та прибуття авіарейсів,
   - у формі *Рейси* розмістити підпорядковані форми *Відліт* для представлення інформації про відправлені літаки за обраним рейсом та *Пілоти* для представлення інформації про пілотів, що літали за обраним рейсом.

У формі *Відліт* розмістити обчислювальне поле сума оплати, яке містить загальну суму, яка сплачена за квитки на рейс, та поле *збитки/прибутки*, в якому визначається сума збитків або прибутків від вильоту.

У формі *Рейси* розмістити обчислювальне поле для представлення суми прибутковості/збитковості рейсу.

Створити головну кнопкову форму, з якої здійснювати виклик форм *Рейси*, *Відліт*, *Пілоти*, *Типи літаків* та *ПунктиОбслуговування*. Заповнити базу даних інформацією про 7–10 рейсів у різні пункти призначення з кількістю вильотів на тиждень — 5–7. Ввести інформацію про 5–7 типів літаків та 7–10 пілотів.

Створити звіт *Пілоти*, в якому навести перелік пілотів за кожним із рейсів (номер рейсу, пункт відправлення, пункт призначення, прізвище, ім'я, по батькові, стаж пілота, категорія пілота, вік пілота, впорядкувати за прізвищем за зростанням), з нумерацією пілотів для кожного із рейсів. Визначити середній вік пілотів у кожному із рейсів.

Створити звіт *ПродажКвитків*, в якому навести перелік вильотів у кожному із рейсів (номер рейсу, дата та час відльоту , кількість проданих квитків першої категорії, кількість проданих квитків другої категорії, кількість проданих квитків третьої категорії, сума від продажу квитків, впорядкувати за датою вильоту за зростанням), з нумерацією вильотів для кожного із рейсів (параметром звіту виступає період, протягом якого здійснювався продаж квитків — початкова дата та кінцева дата). Визначити загальну суму від продажу квитків, суму прибутку/збитку за кожним рейсом та за весь період в цілому.

Створити діалогову форму *Звіти* для друку звітів, в якій користувач обирає тип звіту *Пілоти* або *Продаж квитків*, а також може обрати пункт призначення, для якого буде створено звіт (якщо пункт призначення не обрано, то звіт формується для всіх пунктів призначення). Якщо обрано тип звіту — *ПродажКвитків*, то користувач має визначити місяць та рік видачі. Форма також містить кнопки *Просмотр* для перегляду звіту, *Друк* — для друку звіту та *Отмена* — для виходу у головну кнопкову форму. Реалізувати виклик форми *Звіти* із головної кнопкової форми.

5. Створити базу даних обліку інформації про порушення при продажу товарів. У базі даних має бути така інформація про порушення:
   - назва підприємства-порушника, фізична та юридична адреса, телефон, прізвище керівника, розрахунковий рахунок, МФО, назва банку, ідентифікаційний код підприємства, тип підприємства (роздрібна_торгівля_на_ринку, магазин, кіоск, оптова_база),

- номер акта про виявлення порушення, дата, опис порушення, тип порушення (відсутність супровідних документів, відсутність інформації про товар, не відповідає нормативним документам, вичерпаний термін придатності), відмітка про сплату штрафу,
- вид товару (наприклад, електропобутові товари, парфумерно-косметичні вироби, іграшки, інші непродовольчі товари, продовольчі товари), кількість товару, при продажу якого допущено порушення, ціна товару.

Створити форми даних :

- форму *Акти* для представлення інформації про створені акти та підпорядковану форму *Порушення* для представлення інформації про порушення, що зафіксовані в акті (за допомогою кнопки *Інформація про підприємство* реалізувати відкриття форми *Постачальник* для підприємства, на якому виявлено порушення);
- форму *Підприємства* для представлення інформації про підприємства;
- форму *ТипПідприємства* для представлення інформації про типи підприємства;
- форму *ВидиТовару* для представлення інформації про види товарів.

У формі *Порушення* розмістити обчислювальне поле вартість товару, при продажу якого виявлено порушення, та поле штраф, значення якого розраховувати залежно від типу виявленого порушення:

- штраф становить 5 % від вартості товару при порушеннях: *відсутність інформації про товар* або *не відповідає нормативним документам*;
- штраф становить 10 % від вартості товару при порушенні: *відсутність супровідних документів*;
- штраф становить 20 % від вартості товару при порушені: *вичерпаний термін придатності*.

У формі *Акти* розмістити обчислювальне поле для представлення сукупної вартості товару, при продажу якого виявлено порушення, та обчислювальне поле для представлення суми штрафу за актом.

Створити головну кнопкову форму, з якої здійснювати виклик форм *Акти*, *Протокол*, *Підприємства*, *ТипПідприємства*, *ВидиТовару*. Заповнити базу даних інформацією про 7–10 актів про 3–5 порушень, що виявлено при роботі 7–10 підприємств.

Створити звіт *ШтрафиПідприємств*, в якому навести перелік виявлених порушень для кожного із підприємств (назва, ідентифікаційний код, прізвище керівника, номер акта, тип порушення, вартість товару, штраф, відмітку про сплату), з нумерацією порушень кожного із підприємств. Визначити сумарний розмір штрафу, що накладено на підприємство, та суму несплаченого штрафу.

Створити звіт *Порушення* (параметром звіту виступає період, протягом якого зафіксовано порушення — початкова дата та кінцева дата), в якому навести перелік порушень (тип порушення, вартість товару, при продажу якого виявлено порушення, штраф, вид товару, номер акта, назва підприємства і дата), з нумерацією порушень для кожного типу порушення. Визначити сумарний розмір штрафу, що отримано від штрафів при порушеннях продажу кожного із типів товарів, та суму несплаченого штрафу. Обчислити процентне співвідношення сум штрафів кожного типу порушення до загальної суми штрафу.

Створити діалогову форму *Звіти* для друку звітів, в якій користувач обирає тип звіту *ШтрафиПідприємств* або *Порушення*, а також може обрати тип порушення, для якого буде створено звіт (якщо тип порушення не обрано, то звіт формується для всіх типів порушення). Якщо обрано тип звіту — *Порушення*, то користувач має визначити початкову та кінцеву дату звітного періоду. Форма також містить кнопки *Просмотр* для перегляду звіту, *Друк* для друку звіту та *Отмена* для виходу у головну кнопкову форму. Реалізувати виклик форми *Звіти* із головної кнопкової форми.

6. Засобами *MS Access* створити засоби введення даних та підготовки звітів для бази даних “Облік аналізів крові пацієнтів” та кнопкову форму.
7. Засобами *MS Access* створити засоби введення даних та підготовки звітів для бази даних “Облік бюджету медичної установи” та кнопкову форму.
8. Засобами *MS Access* створити засоби введення даних та підготовки звітів для бази даних “Облік переліку та вартості процедур, що відпускаються у медичних закладах санітарно-курортного типу” та кнопкову форму.
9. Засобами *MS Access* створити засоби введення даних та підготовки звітів для бази даних “Облік замовлень на отримання лікарських розчинів лікарнею” та кнопкову форму.

10. Засобами *MS Access* створити засоби введення даних та підготовки звітів для бази даних “Облік роботи аптечної бази” та кнопкову форму.

Питання для самоконтролю та співбесіди

1. Для чого призначені форми?
2. Опишіть способи створення форм у *MS Access*.
3. Як виконується пошук у формах *MS Access*?
4. Як здійснити фільтрацію у формах *MS Access*?
5. Які розділи містить форма?
6. Для чого використовується заголовок форми?
7. Для чого використовується примітка форми?
8. Для чого використовується область даних у формах?
9. Чим відрізняються “*табличная форма*” та “*ленточная форма*”?
10. Чим відрізняються “*форма*: *в один столбец*” та “*ленточная форма*”?
11. Як створити форму з використанням полів декількох таблиць?
12. Як змінити джерело записів для форми?
13. Як зв’язуються головна та підпорядкована форма?
14. Що таке підпорядкована форма?
15. Опишіть способи створення підпорядкованих форм.
16. Що таке кнопкові форми? Як створити кнопкову форму у *MS Access*?
17. Як використовується “*Диспетчер кнопочных форм*”?
18. Що таке діалогові форми? Як створити діалогову форму у *MS Access*?
19. Назвіть розділи звіту та опишіть їх призначення.
20. Чим відрізняються розділи звіту: колонтитул та заголовок групи?
21. Чим відрізняються розділи звіту: колонтитул та примітка групи?
22. Чим відрізняються розділи звіту: верхній колонтитул та заголовок звіту?
23. Чим відрізняються розділи звіту: нижній колонтитул та приміткa заголовок звіту?
24. Які властивості можна встановити для області даних звіту?
25. Як створити обчислюване поле в звіті?
26. Як обчислити значення у групах записів у звіті?

27. Як вивести номер сторінки у звіті?
28. Як обчислити кількість записів у кожній групі звіту?
29. Як здійснити нумерацію записів у групі звіту?
30. Як додати групування у звіті?
31. Як на першій сторінці звіту вивести дату друку звіту?
32. Як визначити назву стовпця для групи у звіті?
33. Як обчислити середнє значення для деякого поля за всіма записами групи?
34. Як обчислити середнє значення для деякого поля за всіма записами звіту?
35. Як обчислити для кожного запису звіту процент, який становить значення поля до загальної суми за всіма записами звіту?
36. Назвіть етапи створення звітів за допомогою майстра.
37. Назвіть основні прийоми редагування звітів у режимі Конструктор.
38. Як здійснюються обчислення у звітах?
39. Як здійснюється групування записів у звітах?
40. Яка властивість елемента управління використовується для ідентифікації елемента?
41. Яка власність елемента управління визначає, чи буде елемент відображено на екрані у режимі форми?
42. Для чого призначена властивість *Вывод на экран*?
43. Як використовується властивість *Формат поля*?
44. Які формати полів можна задавати за допомогою властивості *Формат поля*?
45. Для чого призначена властивість *Данные* ?
46. Як використовується властивість *Значение по умолчанию*?
47. Як використовується властивість *Условие на значение*?
48. Що відбудеться при порушенні умови на значення, яка визначена у полі *Условие на значение*?
49. Для чого призначена властивість *Сообщение об ошибке*?
50. Як використовується властивість *Присоединенный столбец*?
51. Для якого елемента управління можна визначити властивість *Число столбцов* та як вона використовується?
52. Як використовується властивість *Ширина столбцов* ?
53. Як в *Access* можна визначити процедуру обробки події?
54. Наведіть приклад процедури обробки події для форми?
55. Наведіть приклад процедури обробки події для звіту?

Тема 6. Розробка інтерфейсу клієнта іншими засобами

Опрацювання програмного матеріалу (за матеріалами конспекту, підручників)

Огляд пакетів, які використовуються для створення клієнтського застосування для доступу до бази даних. Використання компонентів, що використовуються для встановлення зв'язку з базою даних. Компоненти встановлення зв'язку з множиною записів. Відображення полів таблиць та запитів у формі. Використання візуальних компонентів доступу до полів таблиць та запитів. Навігація за базою даних. Аналіз даних. Створення звітів.

Огляд засобів створення клієнтського застосування для доступу до бази даних засобами *Delphi*. Огляд засобів створення клієнтського застосування для доступу до бази даних засобами *C++ Builder*. Огляд засобів створення клієнтського застосування для доступу до бази даних засобами *PowerBuilder*.

Огляд засобів створення клієнтського застосування для доступу до бази даних засобами *Visual Studio*. Створення заготовки додатку з використанням класів бібліотек *MFC ODBC* та *DAO*. Методика програмної зміни джерела даних. Створення екранної форми для відображення вмісту бази даних за допомогою класів *MFC*. Використання властивостей об'єкта *m_pSet* класу *CRecordset* бібліотеки *MFC* для добавлення (знищення) записів у таблицях, сортування та фільтрації записів. Розробка програмного додатку на базі бібліотеки *MFC* для створення та реалізації явних *SQL* запитів до бази даних *Microsoft SQL Server* на вибірку, додання, знищення, фільтрацію, сортування та модифікацію записів таблиці бази даних.

Література [1; 2; 12; 22–30]

Основні визначення

Для створення клієнтського застосування для доступу до бази даних у середовищі *C++ Builder* використовують такі компоненти:

Компонент *TDataSource* виступає як посередник між компонентами *TDataSet* (*TTable*, *TQuery*, *TStoredProc*) та компонентами *Data Controls* — елементами управління, що забезпечують представлення даних у формі.

Зазвичай компонент *DataSource* зв'язаний з одним компонентом *TDataSet* (*TTable* або *TQuery*) та з одним або більше компонентів *Data Controls* (такими, як *DBGrid*, *DBEdit* тощо). Зв'язок *DataSource* з ком-

понентами *TDataSet* та DataControls реалізується за допомогою таких властивостей:

- *DataSet* компонента *DataSource* ідентифікує ім'я компонента *TDataSet*,
- *Enabled* компонента *DataSource* активізує або зупиняє взаємозв'язок між компонентами *TDataSource* та *Data Controls*.

Компонент *TTable* використовується для надання доступу до однієї таблиці. Події компонента *TTable* дозволяють будувати та контролювати поведінку застосування бази даних. Наприклад, подія *BeforePost* настає перед додаванням або зміною запису, подія *AfterPost* — після збереження доданого або зміненого запису, подія *AfterDelete* — після видалення запису тощо. Основні властивості *TTable*:

- *Active* — вказує відкрита (true) або ні (false) ця таблиця,
- *DatabaseName* — ім'я каталогу, що містить таблицю або псевдонім (*alias*) віддаленої бази даних,
- *Fields* — масив об'єктів *TField*, використовується для звернення до полів за номером.

Компонент *TField* дозволяє звертатись до окремих полів набору даних. Для виведення даних у формі за допомогою компонента *TField* необхідно:

1) розмістити компонент *TTable* або *TQuery* на формі;
2) встановити властивість *DatabaseName* для *TTable* або *TQuery*;
3) встановити властивість *TableName* компонента *TTable* або властивість *SQL* компонента *TQuery*;
4) обрати компонент *TDataSet* на формі та в контекстному меню обрати *Fields Editor*, у контекстному меню якого необхідно обрати команду *Add Fields*;
5) у вікні *Add Fields* обрати поля, які необхідно внести в список об'єктів;
6) для створення обчислюваного поля з контекстного меню *Fields Editor* обрати команду *New Field*.

Компонент *TDBGrid* забезпечує табличний спосіб відображення на екрані рядків даних з компонентів *TTable* або *TQuery*, а також виконання відображення, додавання, зниження, редагування даних у базі даних. Зовнішній вигляд таблиці може бути змінений за допомогою редактора властивостей *Columns Editor*.

Найпростішим способом реалізації навігації у записах таблиці є використання компонента *DBNavigator* разом, з компонентом *DBGrid*. Можна також використовувати інші інтерфейсні елементи, для яких

реалізувати обробники подій за допомогою методів *First*, *Last*, *Next*, *Ptior*, *Insert*, *Delete*, *Edit*, *Append*, *Post*, *Cancel* компонента *TTable*.

Компоненти *TDBLookup* (*TDBLookupListBox* та *TDBLookupCombo Box* сторінки Data Controls) використовуються для виведення переліку можливих значень, які знаходяться у деякій зв'язаній таблиці.

Компонент *TQuery,* як і компонент *TTable,* має усі властивості компонента *TDataSet*. Основною властивістю компонента *Query* є властивість *SQL*, що має тип *TString*. Властивість *SQL* може формуватись як у режимі розробки застосування (за допомогою Visual Query Builder), так і під час виконання застосування за допомогою методів класу *TString*.

Компоненти, які використовуються для побудови звітів, розташовані на сторінці *QReport* палітри компонентів. Для створення звіту у формі необхідно розмістити компонент *TQuickReport*.

Тематика практичних завдань

1. Створити базу даних для організації роботи відділу кадрів та клієнтське застосування для доступу до бази даних. У базі даних має бути така інформація: прізвище, ім'я, по батькові працівника, табельний номер, паспортні дані, ідентифікаційний код, посада, оклад, форма працевлаштування, структурний підрозділ, де працює співробітник, відомості про структурні підрозділи підприємства, керівників структурних підрозділів.
2. Створити базу даних для обліку руху товарів на складі та клієнтське застосування для доступу до бази даних. У базі даних має бути така інформація: код, найменування товару, виробник, адреса виробника та його реквізити, категорія товару, країна виробництва, обсяг продукту, одиниця виміру, відмітка про прибуття/вибуття, дата прибуття/вибуття, інформація про супроводжуючі документи про прибуття/вибуття, відповідальна особа за прийняття/відправку товару.
3. Створити базу даних обліку інформації про співробітників фірми та клієнтське застосування для доступу до бази даних. У базі даних має бути така інформація про співробітника: прізвище, ім'я, по батькові, посада, оклад для визначеної посади, персональна надбавка для цього працівника, дата найму співробітника, робочий телефон, номер паспорта, дата народження, місце проживання, адреса проживання, домашній телефон, фотографія, назва філіалу, в якому працює співробітник, ідентифікаційний код філіалу, місце розташування філіалу, адреса філіалу, телефон

офіса Створити форми для введення та перегляду інформації в таблицях. У формі *Співробітник* розмістити обчислювальні поля заробітна плата, яка складається з окладу посади, надбавки за вислугу років — 50 грн за кожні 5 років служби та персональної надбавки, податок (13% на всю суму), сума на руки, а також визначити середній вік та стаж співробітників у кожному із філіалів та в цілому у фірмі.

4. Створити базу даних обліку інформації про співробітників фірми та клієнтське застосування для доступу до бази даних. У базі даних має бути така інформація про товар: назва, категорія товару, ціна товару, одиниця виміру товару, назва постачальника, країна, місце та адреса постачальника, телефон постачальника та адреса Web-сторінки, номер поставки, дата поставки товару, кількість товару у поставці, знижка при поставці, вартість доставки. Створити форми даних для обліку інформації про товари, для обліку інформації про постачальників, для обліку інформації про поставки, для представлення інформації про категорії товарів. Заповнити базу даних інформацією про 7–10 поставок, кожна з яких включає поставку 7–10 товарів (7–10 різних категорій). Поставки здійснюються 7–10 постачальниками. У формах реалізувати такі обчислення:
   - ціни продажу товару з урахуванням знижки;
   - суми поставки з урахуванням знижки та вартості доставки товару;
   - сумарну вартість поставлених товарів за кожним із постачальників та частку (у процентах) сумарної вартості поставок кожним постачальником від загальної вартості поставлених товарів;
   - загальну сплачену суму за кожною із категорій та усіма товарам в цілому.
5. Створити базу даних обліку інформації про співробітників фірми та клієнтське застосування для доступу до бази даних. У базі даних має бути така інформація про клієнта: назва клієнта, прізвище, ім'я, по батькові директора, контактний телефон, номер рахунка клієнта, країна, місто та юридична адреса клієнта, ідентифікаційний код клієнта, вид робіт, виконання яких замовив клієнт, опис замовлення, сума замовлення, знижка, дата замовлення, дата виконання замовлення, номер акта прийняття робіт, сума оплати за замовлення (оплата може здійснюватись за кількома рахунками), дата оплати, метод оплати, номер платіж-

ного рахунку. Створити засоби введення інформації до бази даних (форми), в яких обчислити:

- середній розмір замовлення кожного із видів робіт та загальний розмір боргу,
- загальну суму оплати щотижнево та за вказаний період, а також процентне співвідношення сплачених сум протягом кожного тижня.

6. Створити базу даних для контролю параметрів процесу та клієнтське застосування для доступу до бази даних. У базі даних має бути така інформація: шифр параметра, найменування, розмірність, мінімальне значення, максимальне значення, поточне значення, шифр апарата, найменування апарата, лінійні розміри тощо.
7. Створити базу даних для контролю успішності студентів у різних групах та клієнтське застосування для доступу до бази даних. У базі даних має бути така інформація: номер залікової книжки, прізвище, ім'я, по батькові студента, рік народження, шифр групи, найменування дисципліни, оцінка, викладач, кафедра тощо. Забезпечити виконання таких обчислень:
   - середня оцінка студента, групи,
   - середній бал з дисципліни,
   - кількість студентів групи, які отримали оцінку "відмінно", "добре", "задовільно", "незадовільно".
8. Створити базу даних для розрахунку стипендії студентів у різних групах та клієнтське застосування для доступу до бази даних. У базі даних має бути така інформація: номер залікової книжки, прізвище, ім'я, по батькові студента, рік народження, шифр групи, рейтинг, коефіцієнт доплати, основна стипендія, сума доплати та інформація про здійснені виплати за місяцями. Забезпечити виконання таких обчислень:
   - частка студентів, що отримують підвищену стипендію,
   - частка студентів, що отримують звичайну стипендію,
   - частка студентів, що не отримують стипендію,
   - стипендіальний фонд із зазначенням за студентами, групами та загальний.
9. Створити базу даних для обліку роботи автотранспортного підприємства та клієнтське застосування для доступу до бази даних. У базі даних має бути така інформація: табельний номер водія, прізвище, ім'я, по батькові, клас, тариф оплати, дата виїзду, пробіг, обсяг вантажу, номер дорожнього листа, тип автомобіля, номер автомобіля тощо. Забезпечити виконання таких обчислень:

- частка водіїв, що отримують визначену у параметрі заробітну плату,
- частка водіїв, що не отримували у визначеному у параметрі місяці заробітну плату,
- фонд заробітної плати стосовно водіїв, класів водіїв, типів автомобілів та загальний.

10. Створити базу даних для обліку роботи комп'ютерної фірми та клієнтське застосування для доступу до бази даних. У базі даних має бути така інформація: марка комп'ютера, тип процесора, тактова частота, розмір оперативної пам'яті та жорсткого диска, ціна, початкова кількість, дата продажу, прізвище покупця, кількість проданих комп'ютерів тощо. Табельний номер водія, прізвище, ім'я, по батькові, клас, тариф оплати, дата виїзду, пробіг, обсяг вантажу, номер дорожнього листа, тип автомобіля, номер автомобіля тощо. Забезпечити виконання таких обчислень:
    - кількість проданих комп'ютерів кожної марки,
    - обсяг проданих комп'ютерів марки, яка визначена в параметрі,
    - обсяг замовлень кожний покупцем.
11. Створити базу даних для обліку роботи міської АТС та клієнтське застосування для доступу до бази даних. У базі даних має бути така інформація: прізвище, ім'я, по батькові абонента, домашня адреса, номер телефону, абонентна плата, тип з'єднання, інформація про розмови (кількість хвилин, тип розмови: міська, міжміська, міжнародна, на мобільний, номер абонента з'єднання тощо). Забезпечити виконання таких обчислень:
    - обсяг сплати щомісячно,
    - обсяг сплати за кожен вид розмови,
    - рахунок за місяць.
12. Створити базу даних для обліку роботи каси аеропорту та клієнтське застосування для доступу до бази даних. У базі даних має бути така інформація: номер рейсу, пункт призначення, час вильоту, дата вильоту, прізвище, ім'я, по батькові пасажира, номер паспорта, вартість білета, клас білета, відмітка про проходження реєстрації тощо.

Питання для самоконтролю та співбесіди

1. Програмна навігація за записами таблиці.
2. Назвіть методи, які використовуються для роботи з записами таблиці .
3. Як здійснюється доступ до значень полів даних у програмі?

4. Як забезпечується введення даних користувачем за допомогою компоненти *Edit* ?
5. Як здійснити виведення обчислюваних значень у форму застосування?
6. Опишіть механізм створення обробника події для компонента *Borland C++ Builder.*
7. Як здійснюється створення звітів у *Borland C++ Builder*?
8. Опишіть компоненти, що забезпечують виведення даних у звіт.
9. Опишіть компоненти доступу до даних, які використовуються у *C++ Builder.*
10. Опишіть компоненти доступу до даних, які використовуються у *Delphi.*
11. Назвіть компоненти відображення даних, які використовуються у *C++ Builder.*
12. Назвіть компоненти відображення даних, які використовуються у *Delphi.*
13. Які компоненти використовуються для зв'язку клієнтського застосування з базою даних у *C++ Builder*?
14. Які компоненти використовуються для зв'язку клієнтського застосування з базою даних у *Delphi*?
15. Які компоненти та методи використовуються для навігації за записами бази даних у *C++ Builder*?
16. Які компоненти та методи використовуються для навігації за записами бази даних у *Delphi*?
17. Створення та виконання запитів у середовищі *C++ Builde*r.
18. Створення та виконання запитів у середовищі *Delphi.*
19. Які компоненти використовуються для створення звітів засобами *C++ Builder*?
20. Які компоненти використовуються для створення звітів засобами *Delphi*?
21. Яким чином реалізується з'єднання додатку *VB* з базою даних *Microsoft SQL Server*?
22. Призначення класу *CDatabase* бібліотеки *MFC*?
23. Призначення класу *CRecordset* бібліотеки *MFC*?
24. Яким чином реалізувати з'єднання додатку VC з базою даних *Microsoft SQL Server*, що захищена паролем?
25. Яким чином реалізувати явний SQL-запит додатку VC до бази даних *Microsoft SQL Server*?

СПИСОК ЛІТЕРАТУРИ

Основна

1. *Oracle PL/SQL* для профессионалов / С. Фейерштейн. Б. – 3-е изд. — Изд-во: Питер. 2004 г .
2. *Артемов А.* Microsoft SQL Server для профессионалов. — М.: Издат. дом "Вильямс", 2002. — 576 с.
3. *Боровиков В. В.* Access 2002. — Изд-во: Солон-Р. — 560 с.
4. *Боуман Д.* та інші. Практическое руководство по SQL // Д. Боуман, С. Емерсон, М. Дарновски — К.: Диалектика, 1997. — 336 с.
5. *Грабер М.* Введение в SQL. — М.: Лори, 1996. — 379 с.
6. *Коннолли Т.* та інші. Базы данных. Проектирование, реализация и сопровождение. Теория и практика. — М: Изд-во "Вильямс". 2000. — 1120 с.
7. *Кузнецов С. Д.* Введение в стандарты языка баз данных SQL. — Центр информац. технол., 1998.
8. *Методичні* вказівки до виконання лабораторних робіт "Система управління базами даних Microsoft Access: Лабораторний практикум. — Ч. 1" / О. В. Вітюк, А. В. Кузьмін, Н. М. Москалькова, В. В. Попов, М. Є. Сініцький, Ю. А. Тарнавський. — К.: МАУП, 2003. — 166 с.
9. *Методичні* вказівки до виконання лабораторних робіт "Система управління базами даних Microsoft Access: лабораторний практикум — О. В. Вітюк, А. В. Кузьмін, Н. М. Москалькова, В. В. Попов, М. Є. Сініцький, Ю. А. Тарнавський. — К.: МАУП, 2004. — 168 с.
10. *Попов В. В.* та інші. Практикум і контрольні роботи з MS Access: Метод. вказівки до виконання контрол. і самостій. робіт / В. В. Попов, Л. О. Левченко, Н. М. Москалькова. — К.: МАУП, 2006. — 136 с.

Додаткова

11. *Хомоненко А. Д.* та інші. Базы данных: Учеб. для высш. учеб. заведений / Под ред. проф. А. Д. Хомоненко. — СПб.: КОРОНА-принт, 2000. — 416 с.
12. *Роберт Сигнор, Михаэль О. Стегман.* Использование ODBC для доступа к базам данных. — М.: БИНОМ, 1995. — 384 с.
13. *Послед Б. С.* Access 2002. Приложения баз данных: Лекции и упражнения. — Изд-во: DiaSoft UP. — 656 с.

14. *Дейт К.* Введение в системы баз данных. — 7-е изд. — М.: Издат. дом "Вильямс", 2001. — 1072 с.
15. *Диго С. М.* Проектирование и использование баз данных. — М.: Финансы и статистика, 1995. — 208 с.
16. *Дрибас В. П.* Реляционные модели данных. — М.: Мир, 1992. — 192 с.
17. *Каратыгин А.* Access 2000. Руководство пользователя с примерами. — Изд-во: ЛБЗ-ЮМС. — 376 с.
18. *Коупстейк С.* Access 97 шаг за шагом. — М.: Бином, 1998. — 208 с.
19. *Мартин Дж.* Организация баз данных в вычислительных системах — М.: Мир, 1980. — 662 с.
20. *Мейер М.* Теория реляционных баз данных. — М.: Мир, 1987. — 608 с.
21. *Ульман Д.* Основы систем баз данных. — М.: Финансы и статистика, 1983. — 334 с.
22. *Глушаков С. В., Зорянский В. Н., Хоменко С. Н.* Программирование в среде Borland C++ Builder 6. — Х.: Фолио, 2003. — 508 с.
23. *Фаронов В. В.* Программирование баз данных в Delphi 7. — 2003. — 464 с.
24. *Грейвс М.* Проектрование баз данных на основе XML. — М.: Изд. дом "Вильямс", 2002. — 639 с.
25. *Секунов Н.* Visual C++.NET. — СПб.: БХВ-Петербург, 2002. — 736 с.
26. *Фролов А. В., Фролов Г. В.* Базы данных в Интернете: Практ. руководство по созданию Web-приложений с базами данных. — 2-е Изд. испр. — М.: Изд.-торг. дом "Русская редакция", 2000. — 448 с.
27. *Грофф Дж., Вайнберг П.* Энциклопедия SQL. — 3-е изд. — СПб.: Питер, 2003.
28. *Горев А., Макашарипов С., Владимиров Ю.* Microsoft SQL Server 6.5 для профессионалов. — СПб.: Питер, 1998.
29. *Каучмэн Дж.С., Швинн У.* Oracle 8 Certified Professional DBA. Подготовка администраторов баз данных: Пер. с англ. — М.: Лори, 2002.
30. *Oracle 9i.* Программирование на языке PL/SQL: Разработка эффективных приложений с помощью PL/SQL / Скотт Урман. — Изд-во "Лори", 2004.

ЗМІСТ

Відповідальний за випуск *А. Д. Вегеренко*
Редактор *С. М. Толкачова*
Комп’ютерне верстання *О. А. Залужна*

Зам. № ВКЦ-3869

Підп. до друку 22.03.2009. Формат 60×84/16. Папір офсетний. Друк ротаційний трафаретний. Ум.-друк. арк. 3,02. Обл.-вид. арк. 2,87. Наклад 50 пр.

Міжрегіональна Академія управління персоналом (МАУП)
03039 Київ-39, вул. Фрометівська, 2, МАУП
ДП «Видавничий дім «Персонал»
03039 Київ-39, просп. Червонозоряний, 119, літ. ХХ

Свідоцтво про внесення до Державного реєстру суб’єктів видавничої справи ДК № 3262 від 26.08.2008