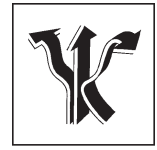
МІЖРЕГІОНАЛЬНА
АКАДЕМІЯ УПРАВЛІННЯ ПЕРСОНАЛОМ



МАУП

МЕТОДИЧНІ МАТЕРІАЛИ ЩОДО ЗАБЕЗПЕЧЕННЯ САМОСТІЙНОЇ РОБОТИ СТУДЕНТІВ

з дисципліни

“ПРАКТИЧНІ АСПЕКТИ ПОБУДОВИ БАЗ ДАНИХ”

(для бакалаврів)

Київ
ДП «Видавничий дім «Персонал»
2010

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

Затверджено на засіданні кафедри інформатики та інформаційних технологій (протокол № 17 від 29.02.08)

Схвалено Вченою радою Міжрегіональної Академії управління персоналом

Москалькова Н. М. Методичні матеріали щодо забезпечення самостійної роботи студентів з дисципліни "Практичні аспекти побудови баз даних" (для бакалаврів). — К.: ДП «Вид. дім «Персонал», 2010. — 49 с.

Методична розробка містить пояснювальну записку, тематику для самостійного опрацювання програмного матеріалу (за матеріалами конспекту, підручників), основні визначення за темою, практичні завдання, питання для самоконтролю та співбесіди, а також список літератури.

ПОЯСНЮВАЛЬНА ЗАПИСКА

Метою дисципліни "Практичні аспекти побудови баз даних" є ознайомлення студентів з практичними аспектами проектування баз даних та побудови програмних систем, які використовують бази даних, а також набуття практичних навичок використання сучасних технологій проектування баз даних, систем управління базами даних та розробки баз даних в архітектурі клієнт-сервер.

Завданням курсу є поглиблення теоретичних знань, необхідних для розв'язання завдань автоматизації обробки інформації у різних предметних областях, а також набуття практичних навичок використання та проектування інформаційних систем, розробки програмних засобів збереження та маніпулювання даними. Під час вивчення дисципліни передбачається систематична практична робота студентів як під керівництвом викладача, так і самостійно. Предметом вивчення курсу "Практичні аспекти побудови 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 у 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 систем управління базами даних;
- принципи побудови та технологію проектування баз даних;
- мову побудови запитів SQL.

Після вивчення дисципліни студент повинен набути навичок:

- проектування інформаційних систем та баз даних;
- 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 "Практичні аспекти побудови баз даних" студенти повинні мати базові знання і навички роботи на

персональному комп'ютері в операційній системі *Microsoft Windows*, із стандартними програмами *Microsoft Windows* і додатками з пакету *Microsoft Office* (зокрема *MS Access*), а також мати знання 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 вигляді заліку.

Самостійна робота студентів є надзвичайно важливою складовою підготовки спеціалістів з напрямку "Прикладна математика". Теоретичний матеріал з програмування 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. Самостійна робота студентів є основним засобом опанування навчального матеріалу у позааудиторний час. Лише постійне самостійне навчання дає можливість оволодіти такою сумою знань і вмінь, які б дали змогу заявити про себе як про професіонала.

Самостійна робота з дисципліни "Практичні аспекти побудови баз даних" передбачає такі види робіт та форми їх оцінювання:

- опрацювання програмного матеріалу зі змістового модуля (за матеріалами конспекту і підручників) та оцінювання результатів під час проміжного контролю;
- підготовку та власне аудиторну роботу 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 або усних доповідей;
- виконання практичних завдань до змістовного модуля та оцінювання результатів під час проміжного контролю;
- виконання письмової контрольної роботи або тестування;
- підготовку до складання заліку.

Згідно із державними стандартами навчальний матеріал навчальної дисципліни, передбачений робочим навчальним планом для засвоєння студентом у процесі самостійної роботи, виноситься на підсумковий контроль поряд з навчальним матеріалом, який опрацьовувався при проведенні навчальних занять. Навчальний час, від-

ведений для самостійної роботи, регламентується робочим навчальним планом і згідно із Болонською декларацією повинен становити не менше 50 % загального обсягу навчального часу студента, відведеного для вивчення конкретної дисципліни. У деяких випадках ця робота проводиться відповідно до заздалегідь складеного графіка, що гарантує можливість індивідуального доступу студента до необхідних дидактичних засобів. Графік доводиться до відома студентів на початку поточного семестру. При організації самостійної роботи студентів з використанням складного обладнання чи устаткування, складних систем доступу до інформації (наприклад, комп'ютерних баз даних, систем автоматизованого проектування тощо) передбачається можливість отримання необхідної консультації або допомоги з боку фахівця.

Самостійна робота студента під час лекції. Лекційний матеріал призначається для спрямування студентів у найбільш раціональному напрямі щодо вивчення навчальної дисципліни і акцентування уваги на найбільш складних, вузлових питаннях навчальної дисципліни. Належне ведення конспекту під час лекції сприяє збереженню необхідної інформації та дає студенту змогу в подальшому проаналізувати її. За умови подання лекційного матеріалу в усній формі одночасно засвоюється до 20 % інформації.

Робота над конспектами лекцій, планами практичних занять. При підготовці до практичних занять студент має спиратися на складений ним конспект лекції. При опрацюванні матеріалу лекції слід зіставити законспектований матеріал з планом практичного заняття, що міститься у методичних матеріалах для практичних занять або у навчально-методичному комплексі.

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 ознайомитися з їх змістом, щоб визначити, чи необхідно опрацьовувати це джерело і чи має воно відношення до навчального курсу, що вивчається, і тільки після цього визначити послідовність його опрацювання і відібрати необхідний для вивчення матеріал з цього джерела (глави, розділи тощо). При опрацюванні матеріалу необхідно з'ясувати суть питання, що вивчається, не уникаючи при цьому визначення суті незрозумілих чи незнайомих слів, термінів. 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. З позицій випереджаючої освіти навчання тільки за конспектом лекцій і основною літературою, вказаною у навчальній програмі, є недостатнім. У більшості випадків належна підготовка потребує вміння швидко знаходити та опрацьовувати необхідний матеріал за першоджерелами, науковою і спеціальною літературою та коректно цитувати знайдене. Тому завдання студента полягають у самостійному знаходженні цих матеріалів шляхом пошуку у паперових або електронних фондах бібліотек, а також у різноманітних файлових архівах, базах даних та базах знань, доступ до яких здійснюється за допомогою відповідних сервісів *Internet* (в основному — *Word Wide Web, FTP та UseNet newsgroups*).

Серед пошукових систем існують як великі за тематикою метапошукові системи, так 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 статтях: 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 "ПРАКТИЧНІ АСПЕКТИ ПОБУДОВИ БАЗ ДАНИХ"

Змістовий модуль І. Практичні аспекти організації та аналізу даних

Тема 1. Організація баз даних у СУБД MS Access

Опрацювання програмного матеріалу (за матеріалами конспекту, підручників).

Проектування інформаційних систем 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.

Література [1; 3; 4; 6—8; 12—15; 18—21]

Основні визначення

У *MS Access* таблиці використовуються для збереження даних. Кожне поле таблиці має набір властивостей, які впливають на спосіб введення та відображення даних у цьому полі. Перелік властивостей поля залежить від типу поля та представляється на вкладці *Общие* у вікні конструктора таблиць. Властивість *Формат* поля слугує для визначення формату представлення значень поля. Властивість *Маска ввода* дозволяє відображати дані у загальноприйнятій формі (наприклад, номер телефону відображати з символами тире). Властивість *Подпись* слугує для підпису, який буде відображатись у заголовку поля (за замовчуванням підпис поля співпадає з його назвою). Властивість *Значение по умолчанию* дає змогу задати значення, яке буде встановлено за замовчуванням для кожного нового запису таблиці. Властивість *Условие на значение* використовується для перевірки належності значення, що вводиться у поле, заданому діапазону. Якщо значення є помилковим, то видається повідомлення, текст якого задається у властивості *Сообщение об ошибке*. Поле можна визначити як обов'язкове (властивість *Обязательное поле*) або дозволити залишати його значення порожнім (властивість *Пустые строки*). За допо-

могою властивості *Индексированное поле* можна визначити індексоване поле, тобто поле, за яким система здійснює прискорений пошук записів. Можна також вимагати унікальність значень індексованого поля, встановивши цій властивості значення *Да (Совпадения не допускаются)*. На вкладці *Подстановка* можна визначити підстановку для заданого поля, яка дозволяє вводити дані в поле за допомогою вибору їх зі списку або поля зі списком.

Окремі таблиці не дають можливості представити структуру та взаємозв'язки об'єктів предметної області. Тому наступним етапом створення схеми даних є встановлення зв'язків. Зв'язок — це асоціація (посилання), що встановлюється між таблицями та дає змогу запобігти надлишковості даних. Зв'язок між таблицями встановлюється за полями, які мають однаковий тип даних. Встановлення зв'язків у *Access* здійснюється у вікні "*Схема данных*", для відкриття якого необхідно виконати команду *Сервис →Схема данных*. У цьому вікні зв'язки між таблицями відображаються у вигляді ліній, які проведено між полями таблиці.

Зв'язки між таблицями можуть мати тип "один до одного" або "один до багатьох". Зв'язок "один до одного" (1:1) встановлюється, коли кожному запису у першій таблиці відповідає один запис з другої таблиці та навпаки (взаємооднозначна відповідність). У цьому разі обидва поля, за якими встановлюється зв'язок, мають бути ключовими або принаймні унікальними. Зв'язки такого типу використовуються, коли необхідно розбити велику таблицю на декілька менших таблиць з метою, наприклад, відокремлення конфіденційної інформації. Наприклад, в базі даних "*Борей*" таблицю "*Сотрудники*" можна розбити на дві таблиці "*Сотрудники личные данные*" та "*Сотрудники служебные данные*" та пов'язати їх за полем "*Код сотрудника*".

Зв'язок "один до багатьох" (1:М) використовується, коли кожному запису першої таблиці (головна таблиця) може відповідати декілька записів другої таблиці (підпорядкованої таблиці) з тим самим значенням пов'язаного поля. Наприклад, в базі даних "*Борей*" зв'язок "один до багатьох" встановлено між таблицями "*Поставщики*" та "*Товары*" за полем "*Код поставщика*". Постачальник може поставляти декілька товарів, однак кожний товар може мати тільки одного постачальника. Поле "*Код поставщика*" в таблиці "*Поставщики*" и "*Товары*" має сумісні типи, а саме у першій таблиці воно має тип "*Счетчик*", а у другій — числовий тип.

Між об'єктами предметної області може існувати співвідношення "багато до багатьох", коли кожному екземпляру одного об'єкта відповідає багато об'єктів іншого типу та навпаки. Такий тип відношення моделюється за допомогою додаткової таблиці, яка пов'язується з першими двома зв'язками типу "один до багатьох". Наприклад, в базі даних "*Борей*" зв'язок між об'єктами "*Товары*" та "*Заказы*" типу "багато до багатьох" реалізовано за допомогою проміжної таблиці "*Заказано"*, в якій ключ складається з двох полів "*Код заказа*" та "*Код товара*". Ключове поле "Код заказа" таблиці "*Заказы*" пов'язане з одноіменним полем з таблиці "*Заказано*" зв'язком типу "один до багатьох".

Для встановлення зв'язку між таблицями необхідно виконати такі дії:

1) закрити всі таблиці та виконати команду *Сервис →Схема данных;*
2) додати необхідні таблиці до вікна схеми даних;
3) перетягнути мишею поле зв'язку з однієї таблиці в іншу;
4) у вікні "*Изменение связей*" (два рази ліва кнопка миші на зв'язку) визначити властивості встановленого зв'язку.

При введенні даних та внесенні змін до таблиць існує необхідність підтримувати цілісність (несуперечливість) даних. Умовами цілісності даних називають набір правил, які використовуються для підтримки міжтабличних зв'язків та заборони на випадкове поновлення або знищення пов'язаних таблиць. Встановлення прапорця "*Обеспечение целостности данных*" дає змогу забезпечити виконання цих умов, а саме:

- додавати запис у підпорядковану таблицю (з боку "багато") можна лише за наявності відповідного запису у головній таблиці;
- заборонено видаляти запис з головної таблиці, якщо у підпорядкованій таблиці є відповідний запис.

У разі введення даних, що порушують цілісність даних, буде видано повідомлення. Пом'якшити правила зміни та видалення записів у пов'язаних таблицях можливо, якщо встановити прапорці "*Каскадное обновление связанных полей*" та "*Каскадное удаление связанных записей*", а саме:

- встановлення прапорця *"Каскадное обновление связанных полей"* включає режим автоматичної зміни пов'язаних записів у підпорядкованій таблиці при зміні значень первинного ключа у головній таблиці;

- встановлення прапорця “*Каскадное удаление связанных записей*” включає режим автоматичного видалення пов’язаних записів з підпорядкованої таблиці при видаленні значення первинного ключа з головної таблиці.

Тематика практичних завдань

1. Загальна характеристика схеми даних навчальної бази даних *Борей*.
2. Загальна характеристика таблиць навчальної бази даних *Борей*.
3. Загальна характеристика форм навчальної бази даних *Борей*.
4. Загальна характеристика запитів навчальної бази даних *Борей*.
5. Загальна характеристика звітів навчальної бази даних *Борей*.
6. Засобами *MS Access* створити схему даних для предметної області “Кредитування на придбання автомобіля” з таблицями “Автомобілі”, “Позичальник”, “Кредити” та відповідними полями: “Автомобілі” (Код автомобіля, Назва автомобіля, Дата випуску, Об’єм двигуна, Дата продажу автомобіля), “Позичальник” (Ідентифікаційний номер, Прізвище, Адреса, Серія та номер паспорта, Місце роботи, Код автомобіля, Сума кредиту, Дата видачі кредиту, Дата погашення кредиту), “Кредити” (Код кредиту, Ідентифікаційний номер, Сума боргу, Термін виплати за місяцями, Процентна ставка, Щомісячний внесок).
7. Засобами *MS Access* створити схему даних для предметної області “Відправлення грошових переказів у гривнях” з таблицями “Відправник”, “Отримувач”, “Пункт” переказу з відповідними полями: Таблиця “Відправник” (Код відправника, Прізвище відправника, Країна, Місто, Адреса, Сума переказу, Плата за переказ, Дата відправлення), Таблиця “Отримувач” (Код отримувача, Код відправника, Прізвище отримувача, Країна, Місто, Адреса, Серія та номер паспорта, Дата отримання переказу), Таблиця “Пункт перекази” (Код пункту, Назва пункту, Контрольний номер грошового переказу, Прізвище оператора, Податок на переказ, Код відправника, Код отримувача).
8. Засобами *MS Access* створити схему даних для предметної області “Облік використання фондів стаціонару медичної установи” з таблицями “Відділення”, “Лікар”, “Пацієнт”, “Діагноз” з відповідними полями: Таблиця “Відділення” (Код, Найменування, Кількість Місць, Добова Вартість Утримання Пацієнта), Таблиця “Лікар” (Код, Прізвище, Ім’я, По батькові, Посада, Відділення,

Кількість Планових Хворих), таблиця "Пацієнт" (Код, Прізвище, Ім'я, По батькові, Номер Картки, Діагноз, Відділення, Дата Госпіталізації, Дата Виписки, Лікар), таблиця "Діагноз" (Код Діагнозу, Діагноз).

9. Засобами *MS Access* створити схему даних для предметної області "Облік аналізів крові пацієнтів" та створити таблиці "Клінічний Аналіз Крові", "Біохімічний Аналіз Крові", "Паспорт Пацієнта", "Діагноз" з відповідними полями: Таблиця "Діагноз" (Код, Найменування), Таблиця "Клінічний Аналіз Крові" (Код, Код Пацієнта, Дата, Еритроцити, Hb, Лейкоцити, Лейк Пал Ядрові, Лейк Сегм Ядрові, Лімфоцитів, Моноцитів, СОЕ, Висновок), Таблиця "БіохімічнийАналізКрові" (Код, Код Пацієнта, Дата, Загальний Білок, Альбуміни, Глобулін Альфа1, Глобулін Альфа2, Глобулін Бета, Глобулін Гамма, Тімолова Проба, Сулемова Проба, АлАТ, Білірубін Загальний, Протромбіновий Індекс, Фібріноген, Висновок), Таблиця "Паспорт Пацієнта" (Номер Картки, Прізвище, Ім'я, По батькові, Дата Народження, Стать, Адреса, МісцеРоботи, Дата Госпіталізації, Дата Виписки, Діагноз При Госпіталізації, Клінічний Діагноз, Супутній Діагноз).
10. Засобами *MS Access* створити схему даних для предметної області "Облік бюджету медичної установи" та створити таблиці "Відділення", "Співробітники", "Посади" з відповідними полями: Таблиця "Відділення" (Код, Найменування, Кількість Місць, Добова Вартість Утримання Пацієнта), Таблиця "Співробітники" (Код, Прізвище, Ім'я, По батькові, Посада, Перснадбавка, Відділення, Стать), Таблиця "Посади" (Код, Найменування, Оклад).
11. Засобами *MS Access* створити схему даних для предметної області "Облік переліку та вартості процедур, що відпускаються у медичних закладах санітарно-курортного типу" та створити таблиці "Лікарські Засоби", "Ліки На Процедуру", "Процедура" з відповідними полями.
12. Засобами *MS Access* створити схему даних для предметної області "Облік замовлень на використання виробничих ліній для таблетування різних лікарських форм" та створити таблиці "Упаковка", "Замовлення", "Склад", "Субстанції" з відповідними полями.
13. Засобами *MS Access* створити схему даних для предметної області "Облік замовлень на отримання лікарських розчинів лікарнею" та

створити таблиці “Розчини”, “Замовлення”, “Пацієнт”, “Діагноз” з відповідними полями.

14. Засобами *MS Access* створити схему даних для предметної області “Облік роботи аптечної бази” та створити таблиці “Склад”, “Фармацевт”, “Замовлення”, “Товар” з відповідними полями.
15. Засобами *MS Access* створити схему даних для предметної області “Облік роботи дистриб’юторської мережі іноземної фармацевтичної компанії (NSP), яка розробляє власні біологічні добавки” та створити таблиці “Склад”, “Валюта”, “Дистриб’ютор”, “Спонсор”, “Товар” з відповідними полями.

Питання для самоконтролю та співбесіди

1. У чому полягає концепція інтегрованої обробки даних?
2. Що таке база даних? Що таке реляційна база даних?
3. Назвіть функції систем управління базами даних.
4. Поясніть відмінність понять “домен” та “тип даних”.
5. Що таке відношення? Що таке ключ?
6. Для чого використовуються зв’язки між таблицями?
7. Для чого виконують нормалізацію бази даних?
8. Дайте визначення першої нормальної форми.
9. Дайте визначення другої нормальної форми.
10. Дайте визначення третьої нормальної форми.
11. Дайте визначення четвертої нормальної форми.
12. Як здійснюється зв’язування таблиць?
13. Для чого використовується майстер підстановки?
14. Що визначає властивість поля “*Присоединенный столбец*”?
15. Що визначає властивість поля “*Число столбцов*” ?
16. Що визначає властивість поля “*Ширина столбцов*” ?
17. Як визначити для підстановки поле зі списком з виведенням на екран двох стовпців?
18. Як приховати виведення стовпця ключа на екран при використанні підстановки?
19. Як визначити тип зв’язку між таблицями у *MS Access*?
20. Як визначити зв’язок між таблицями у *MS Access*?
21. Що таке цілісність даних у таблицях *MS Access*?
22. Що таке каскадне відновлення та каскадне видалення даних у таблицях *MS Access*?

23. Що таке каскадне відновлення та віддалення даних у таблицях *MS Access*?
24. Опишіть процедуру створення таблиці у *MS Access*.
25. Назвіть типи даних, які можна використовувати при визначенні таблиць у *MS Access*.
26. Як визначити маску вводу даних для поля?
27. Як здійснити перевірку правильності значень, що користувач вводить у таблицю?
28. Як визначити повідомлення при введенні неправильного значення, що користувач вводить у таблицю?
29. Що таке індекс? Як визначити індексоване поле?
30. Назвіть способи об'єднання таблиць у запитах.
31. Дайте визначення внутрішнього об'єднання таблиць.
32. Дайте визначення лівого об'єднання таблиць.
33. Дайте визначення правого об'єднання таблиць.
34. Наведіть приклади таблиць, для яких результат виконання лівого об'єднання не співпадатиме з результатом виконання внутрішнього об'єднання.
35. Наведіть приклади таблиць, для яких результат виконання правого об'єднання не співпадатиме з результатом виконання внутрішнього.

Тема 2. Аналіз даних у MS Access

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 зовнішнє 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.

Режими перегляду запитів. Правила конструювання запиту за допомогою режиму *Конструктор*. Перегляд результатів виконання

запиту у режимі *Режим таблицы*. Перегляд запиту у вигляді SQL-конструкції — режим *в виде SQL*. Створення запитів за допомогою майстрів побудови запитів.

Визначення умов відбору записів за допомогою логічних виразів. Логічні оператори *AND, OR, BETWEEN, IN, LIKE*. Правила побудови складних логічних умов за кількома полями запиту.

Обчислення у запитах. Обчислювальні поля. Правила побудови обчислювальних полів. Визначення імен обчислювальних полів. Побудова запитів з параметрами. Використання властивостей запиту.

Групування даних у запитах та використання агрегатних функцій. Використання підзапитів. Використання майстрів для побудови запитів. Побудова простого запиту. Обчислення підсумкових значень за допомогою майстра побудови простих запитів.

Перехресні запити. Побудова перехресного запиту за допомогою майстра. Формування та редагування перехресного запиту у режимі *Конструктор*. Запит *Повторяющиеся записи*. Побудова запиту *Повторяющиеся записи* за допомогою майстра. Засоби побудови запиту *Повторяющиеся записи* у режимі *Конструктор*. Пошук записів без підпорядкованих. Побудова запиту *Записи без подчиненных* за допомогою майстра. Засоби побудови запиту *Записи без подчиненных* у режимі *Конструктор*. Запити-дії. Запити на оновлення даних. Запити на знищення записів. Запити на додавання записів. Запити на створення нової таблиці.

Література: [1—8; 14; 16; 17; 21]

Основні визначення

Запит — це об'єкт, за допомогою якого здійснюється перегляд, зміна та аналіз бази даних так, як це визначено користувачем. Результат виконання запиту виглядає як таблиця та називається динамічним набором записів. У запитах часто використовуються арифметичні та логічні вирази для обчислення значень та визначення умов відбору записів. Для побудови виразів використовується засіб *Построитель выражений*. У рядку "*Условие отбора*" визначають деякий логічний вираз відбору записів. Логічна умова відбору даних за кількома полями складається з кількох простих умов, які об'єднані за допомогою операторів *And* або *Or*, причому якщо умови задані в одному рядку, то буде використано оператор *And*, якщо умови задані в різних рядках, то буде використано оператор *Or*. Логічні оператори можуть вико-

ристовуватись для об'єднання логічних виразів для одного поля. До логічних операторів належать:

- Оператори *AND* та *OR* використовуються для об'єднання логічних виразів за допомогою логічних зв'язок *і* та *або*.
- Логічний оператор *Not* дає змогу 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 перебуває у діапазоні між *значення_1* та *значення_2* (включно), то результатом є значення *True*; у противному разі — *False*.
- За допомогою оператора *In* здійснюється перевірка, чи співпадає 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 противному разі — 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 (наприклад, Like "Иваненко") або використовувати знаки:

- знак зірочка (*) або знак процента (%) — позначає довільну послідовність символів,
- знак питання (?) або символ підкреслення (_) — позначає довільний один символ,
- знак номера (#) — позначає довільну одну цифру,
- квадратні дужки ([]) — довільний символ, який включено до квадратних дужок,
- знак оклику (!) — довільний символ, який не включений до списку,
- дефис (-) — використовується для позначення діапазону символів.

Відібрані у запиті записи можуть бути згруповані з метою обчислення агрегатних функцій для кожної групи записів. Наприклад, співробітників можна згрупувати за посадою та для кожної групи записів обчислити сумарну заробітну плату. Найчастіше використовують такі агрегатні функції.

Функція	Результат	Дозволені типи полів
Sum	Сума значень групи записів	Числовий, Дата/час, Грошовий, Лічильник
Avg	Середнє значення групи записів	
Min	Найменьше значення групи записів	Текстовий, Числовий, Дата/час, Грошовий, Лічильник
Max	Найбільше значення групи записів	
Count	Кількість значень групи записів (без урахування пустих значень)	Текстовий, Числовий, Дата/час, Грошовий, Лічильник, Логічний, Об'єкт OLE

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

Для визначення способу групування записів та обчислення агрегатних функцій у режимі *Конструктор* використовується рядок *Групповая операция*. Для відображення рядку *Групповая операция* необхідно натиснути кнопку *Групповые операции* панелі інструментів *Конструктор запросов*. У рядку *Групповая операция* для кожного поля запиту необхідно у списку обрати одне із значень:

- *Группировка* — якщо за полем необхідно здійснити групування,
- *Выражение* — якщо необхідно здійснити обчислення виразу для групи записів,
- *Условие* — якщо необхідно визначити умову на обчислене агрегатне значення,
- назву агрегатної функції.

Запит за параметром — це запит, для виконання якого необхідно вказати значення одного або кількох параметрів. Значення параметра може бути визначено у стандартному вікні *Введите значение параметра* або у формі (формі даних чи діалоговій формі). Зокрема, запит за параметром дає змогу застосувати багато разів умову для різних значень параметра або параметрів. Запити на вибірку за параметрами часто використовуються як основа для звітів.

Ім'я параметра береться у квадратні дужки та використовується при побудові умовного виразу запиту. Ім'я параметра буде відображатись у запрошенні ввести значення параметра. Наприклад, запит *Продажи по сотрудникам и странам* виконується для значень початкової дати та кінцевої дати, які вводяться користувачем. Умова на значення поля *ДатаИсполнения* має вигляд Between [Начальная дата] And [Конечная дата], де *Начальная дата*, *Конечная дата* — імена параметрів.

Запит *Продажи по сотрудникам и странам* є джерелом даних для звіту *Продажи по сотрудникам и странам*. Таким чином, стає можливим формування звіту про продажі товарів для різних звітних періодів. Для введення значення параметра з діалогової форми або форми даних необхідно при створенні запиту як умову відбору вказати назву елемента управління форми, в яке буде введено значення параметра.

У більшості випадків запити використовуються для відбору даних з кількох таблиць. При цьому запити 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 кількох таблиць або динамічних наборів записів є однією з найважливіших у реляційній алгебрі. Залежно від типу зв'язків, які встановлені між таблицями, розрізняють кілька способів об'єднання:

1) повне об'єднання (декартовий добуток), коли між таблицями зв'язки не встановлено;
2) внутрішнє об'єднання — цей тип використовується за замовчуванням, 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 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 певній умові.

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 боці "один" зв'язку, а права таблиця — на боці "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 співпадають.

Використовується також і зовнішнє об'єднання таблиць:

- ліве зовнішнє об'єднання — 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 співпадають;
- праве зовнішнє об'єднання — 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 співпадають.

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 зовнішнього 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 дійсно здійснювали замовлення, тобто значення *КодКлиента* яких присутнє як в таблиці *Клиенты*, так і в таблиці *Заказы*. До результуючого набору (всього записів — 832) виконання запиту з лівим зовнішнім об'єднанням будуть включені у тому числі і клієнти (*КодКлиента* — PARIS, FISSA), які не здійснювали замовлення (значення поля *КодЗаказа* для цих записів буде порожнім).

Зміна типу зв'язку здійснюється у діалоговому вікні "*Параметры объединения*", яке можна відкрити, натиснувши два рази ліву кнопку миші на зв'язку, або за допомогою команди "*Параметры объединения*" контекстного меню зв'язку. У цьому вікні можна визначити тип об'єднання:

1. Об'єднання тільки тих записів, у яких пов'язані поля обох таблиць співпадають (внутрішнє об'єднання).

2. Об'єднання всіх записів з лівої таблиці та тільки тих записів з правої таблиці, у яких значення пов'язаних полів співпадають (ліве зовнішнє об'єднання).
3. Об'єднання всіх записів з правої таблиці та тільки тих записів з лівої таблиці, у яких значення пов'язаних полів співпадають (праве зовнішнє об'єднання).

Перехресні запити використовуються для аналізу даних за кількома полями. Перехресні запити є аналогом зведених таблиць у програмі *Microsoft Excel.* У перехресному запиті агрегатна (підсумкова) функція застосовується до деякого числового поля з одночасним групуванням даних за кількома (зазвичай двома) іншими полями, причому значення одного поля групування розміщуються у заголовках стовпців результуючого набору записів, а значення інших полів — у назвах рядків. На перетині рядків та стовпців розміщуються результати обчислення агрегатної функції.

Перехресний запит може бути створений за допомогою майстра або в режимі конструктора. Для створення перехресного запиту за допомогою майстра необхідно:

1) створити простий запит на вибірку полів, за якими у перехресному запиті буде здійснюватись групування або обчислення агрегатних функцій, та за необхідності вказати умови відбору записів. Якщо усі необхідні поля знаходяться в одній таблиці, то цей крок можна пропустити;
2) натиснути кнопку "*Создать*" на вкладинці "*Запросы*" та обрати пункт "*Перекрестный запрос*";
3) у вікні "*Создание перекрестных таблиц*" обрати створений на першому кроці запит або потрібну таблицю та натиснути кнопку "*Далее*";
4) обрати за допомогою кнопки ">" одне або кілька полів групування, значення яких будуть розташовані у назвах рядків, та натиснути "*Далее*";
5) обрати одне поле, значення якого будуть розташовані як назви стовпців, та натиснути кнопку "*Далее*";
6) визначити поле, за значенням якого буде обчислено значення агрегатної функції, та обрати назву агрегатної функції;
7) якщо необхідно обчислити підсумкове значення для кожного з рядків, то слід встановити прапорець "*Вычислять итоговое значение для каждой строки*" та натиснути кнопку "*Далее*";
8) визначити ім'я запиту.

У режимі конструктора для перехресного запиту відображається додатковий рядок *Перекрестная таблица*, в якому для кожного поля запиту визначається спосіб його відображення: *Заголовки строк*, *Заголовки столбцов*, *Значение* або (*не отображается*). Лише для одного поля може бути визначено значення *Заголовки столбцов* та лише для одного поля може бути визначено значення *Значение*.

Серед полів таблиці можна виділити поля, значення яких дозволяють однозначно ідентифікувати записи таблиці, тобто не існує наборів значень цих полів, які зустрічаються у кількох записах. Наприклад, для ідентифікації авіарейсу достатньо визначити номер рейсу та дату вильоту. Один з таких наборів полів визначається як первинний ключ таблиці. Однак для інших полів таблиці або запитів допускаються повторення. Наприклад, фірма може має багато клієнтів з однієї країни та одного міста. Тоді в таблиці *"Клиенты"* можуть бути присутні набори записів з однаковими значеннями в полях "*Страна"* та "*Город*". Запит типу "*Повторяющиеся записи*" використовується для вибірки з таблиці тих записів, у яких значення визначених полів співпадають.

Для забезпечення цілісності даних *Microsoft Access* автоматично проводить аналіз з метою запобігання суперечливості даних у таблицях. Наприклад, у таблиці "*Заказы*" бази даних "*Борей*" в полі "*Код клиента*" не можна ввести значення, якого немає в такому самому полі в таблиці "*Клиенты*", тобто неіснуючий клієнт не може зробити замовлення. Однак у таблиці клієнтів можуть бути записи про клієнтів, які не робили замовлень. Для пошуку таких записів використовується майстер побудови записів типу "*Записи без подчиненных*".

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. У режимі "*Таблица*" з меню "*Вид*" можна переглянути записи, над якими будуть виконані дії. Для виконання запиту на зміну даних необхідно запустити його за допомогою кнопки "Запуск" панелі інструментів *Конструктор запросов*.

Для створення запитів на зміну даних необхідно створити запит на вибірку даних та змінити тип запиту за допомогою кнопки "*Тип запроса*" панелі інструментів *Конструктор запросов*. Далі процес побудови запиту залежить від обраного типу запиту. Зокрема, для запиту на оновлення даних у рядку "*Обновление*" необхідно визначити

вираз, згідно із яким буде обчислено нове значення поля. Для запиту на додавання у рядку "*Добавление*" необхідно вказати поле, у якому будуть розміщені дані. Для запиту на створення нової таблиці необхідно визначити ім'я створюваної таблиці тощо.

Тематика практичних завдань

Запит 1. Знайти марки товарів типу "*Молочные продукты*" або "*Рыбопродукты*", які постачаються у пакетах.

Запит 2. Яку кількість замовлень обслужив кожний із співробітників, вік якого більше 35 років, та на яку суму (з урахуванням знижки).

Запит 3. Визначити 10 країн, до яких було поставлено поштою товарів на найбільшу суму (з урахуванням знижки та вартості доставки).

Запит 4. Знайти постачальників та співробітників з однакових країн.

Запит 5. Яку кількість замовлень обслужив кожний із співробітників, які були прийняті на роботу у другому півріччі 1993 року, та на яку суму (з урахуванням знижки).

Запит 6. Знайти марки товарів типу "*Кондитерские изделия*" або "*Мясо/птица*", які постачаються у коробках.

Запит 7. Для кожного типу товарів визначити розмір максимальної знижки.

Запит 8. Визначити перелік товарів типу "*Молочные продукты*", які повторно були поставлені клієнтам з Австрії.

Запит 9. Визначити кількість та суму замовлень, які були розміщені у різні роки.

Запит 10. Знайти співробітників, які оформили замовлень товарів на суму (з урахуванням знижки) більше середнього.

Запит 11. Визначити кількість та суму замовлень, які доставлені кожним із видів доставки.

Запит 12. Сформувати перелік клієнтів та категорій товарів, які вони не замовляли.

Запит 13. Сформувати перелік марок товарів, які не доставляли до *США* та *Канади*.

Запит 14. Знайти постачальників та клієнтів з однакових міст.

Запит 15. На яку загальну суму замовлено товарів кожного типу до кожної країни клієнта (перехресний запит).

Запит 16. Яка кількість товарів зберігається на складі для кожного типу товару та кожної країни постачальника (перехресний запит).

Запит 17. Створити таблицю *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 р., та замовлення на які були розміщені у 1997 р.

Запит 18. В усіх записах таблиці *Таблица1* змінити назву постачальника *Forots d'Erables* на *Forots Erables*.

Запит 19. Створити таблицю *Таблиця2*, до якої розмістити інформацію про товари (марка товару, категорія товару, назва постачальника та ціна товару), тип яких "*Фрукты*", ціна перевищує 2000 р. та замовлення на які були розміщені у 1998 році.

Запит 20. З таблиці *Таблиця2* видалити всі записи про товари, ціна яких більше 2300 р.

Питання для самоконтролю та співбесіди

1. Що таке запит?
2. Які види запитів існують в *MS Access*?
3. Як створити запит на основі фільтра?
4. Як створити запит на вибірку даних за допомогою майстра побудови простих запитів?
5. Як створити запит в режимі *Конструктор*?
6. Що таке обчислюване поле?
7. Правила побудови виразів у *MS Access*.
8. Побудова виразів засобами *Построителя выражений*.
9. Використання вбудованих функцій у майстрі побудови виразів.
10. Використання фінансових функцій у майстрі побудови виразів.
11. Використання функцій *Дата/Время* у майстрі побудови виразів.
12. Використання статистичних функцій у майстрі побудови виразів.
13. Використання статистичних функцій у майстрі побудови виразів.
14. Використання текстових функцій у майстрі побудови виразів.
15. Як формуються прості умови відбору записів?
16. Як відібрати 15 записів, що мають найбільші значення за деяким полем?
17. Як формуються складені умови відбору записів?
18. Як знайти унікальні значення деякого поля таблиці?
19. Як здійснюється сортування записів у запиті?

20. Які агрегатні функції можна використовувати в запитах?
21. Для чого використовується конструкція *SELECT?*
22. Для чого використовується службове слово *AS?*
23. Для чого використовується службове слово *FROM?*
24. Для чого використовується конструкція *UNION?*
25. Для чого використовується конструкція *ORDER BY?*
26. Для чого використовується конструкція *GROUP BY?*
27. Як мовою *SQL* визначається поле, за яким здійснюється об'єднання таблиць?
28. Як мовою *SQL* визначається умова відбору записів?
29. Як мовою *SQL* визначається групування записів?
30. Що таке запит з параметром?
31. Як здійснюється побудова обчислювальних полів у запитах?
32. Наведіть приклади запитів з використанням логічного оператора *AND*.
33. Наведіть приклади запитів з використанням логічного оператора *OR*.
34. Наведіть приклади запитів з використанням логічного оператора *BETWEEN*.
35. Наведіть приклади запитів з використанням логічного оператора *IN*.
36. Наведіть приклади запитів з використанням логічного оператора *LIKE*.
37. Як у запитах здійснюється групування даних?
38. Опишіть створення простого запиту за допомогою майстра.
39. Опишіть створення підсумкового запиту за допомогою майстра.
40. Для чого використовуються перехресні запити?
41. Опишіть створення перехресного запиту за допомогою майстра.
42. Як створити перехресний запит у режимі конструктора?
43. Опишіть створення запиту на пошук записів, що повторюються, за допомогою майстра.
44. Опишіть створення запиту на пошук записів без підлеглих за допомогою майстра.
45. Для чого використовуються запити-дії?
46. Назвіть типи запитів-дій.
47. Наведіть приклад запиту на створення таблиці.
48. Наведіть приклад запиту на додавання записів у таблицю.
49. Наведіть приклад запиту на видалення записів з таблиці.
50. Наведіть приклад запиту на оновлення записів у таблиці.

Змістовний модуль ІІ. Практичні аспекти побудови клієнтських застосувань

Тема 3. 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 редагуванні даних. Використання трансакцій. Обробка мережних 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; 6—8; 10; 14; 16; 17; 21—24]

Основні визначення

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

Для доступу до даних у клієнтських застосуваннях частіше за все 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. Найпоширенішими ме-

ханізмами 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 являє собою набір динамічних бібліотек DLL, які обслуговують підключення та роботу з конкретним типом бази даних. При запиті на підключення до деякої заздалегідь описаної бази активізується деякий DLL — драйвер цього типу бази даних. Звернення до бази даних відбувається за ім'ям джерела даних ODBC (DSN — data source name). DSN є оголошенням 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 команд и SQL-запитів з параметрами, *Record* — представляє один запис об'єкта *Recordset*, *Stream* – використовується для читання та запису потокових даних, наприклад документів XML або двійкових об'єктів, *Errors* — наводить помилки, *Fields* — наводить 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 структури самої бази даних. Цей механізм дає змогу 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 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 у 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 дає змогу 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 даних "*Борей*".
2. Реалізувати систему заходів для запобігання несанкціонованого доступу до бази даних *"Борей"*.
3. Для бази даних "*Борей"* створити групу користувачів *Клієнти*, для яких визначити доступ до об'єктів бази даних, де подана інформація про наявні товари та заборонити доступ до об'єктів бази даних, де подана інформація про службову інформацію торгової фірми "*Борей"*.
4. Для бази даних "*Борей"* створити групу користувачів *Відділ кадрів*, для яких визначити доступ до об'єктів бази даних, де подана інформація про співробітників фірми та обсяг виконаної ними роботи, та заборонити доступ до об'єктів бази даних, де подана інформація про товари та клієнтів торгової фірми "*Борей"*.
5. Для бази даних "*Борей"* встановити параметри запуску програми для клієнтів торгової фірми "*Борей"*.

Питання для самоконтролю та співбесіди

1. Які засоби захисту даних реалізовано у *MS Access*?
2. Як визначити пароль бази даних у *MS Access*?
3. Як створити *mde*-файл?

4. Які параметри запуску програми можна визначити засобами *MS Access*?
5. Як здійснити стиснення даних у *MS Access*?

Тема 4. Розробка інтерфейсу клієнта засобами Access

Опрацювання програмного матеріалу (за матеріалами конспекту, підручників).

Форми. Класифікація форм. Використання форм для введення даних у режимі *Режим формы*. Проектування форм у режимі *Конструктор*. Відображення джерела даних за допомогою режиму *Режим таблицы*. Панель інструментів *Режим формы*. Пошук, сортування та фільтрація даних у формі. Запис фільтра у вигляді запиту. Експорт результатів виконання фільтрації. Створення макету 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 кількох таблиць або запитів. Додавання підпорядкованої форми.

Використання обчислювальних елементів управління у формах. Розрахунок підсумкових значень за групою записів підпорядкованої форми. Створення кнопочних форм. Використання засобу *Диспетчер кнопочных форм*. Побудова діалогових форм. Введення параметрів запитів за допомогою діалогових форм.

Призначення звітів. Розділи звітів. Елементи управління, що використовуються у звітах. 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 [1—8; 14; 16; 17; 21]

Основні визначення

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

1) дають можливість вводити та редагувати дані в таблицях та виводити результат виконання запитів (форми даних);
2) дають можливість здійснювати управління роботою програми (кнопочні 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 — 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 заголовку форми;
- *область данных* — використовується для відображення даних.

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

Звіт — це об'єкт, який призначений 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 управління. Звіти зручніше створювати за допомогою майстра з подальшим редагуванням у режимі *Конструктор*.

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

- *Заголовок отчета* — вміст розділу виводиться лише на першій сторінці звіту;
- *Верхний колонтитул* — вміст розділу виводиться угорі кожної сторінки;
- *Заголовок группы* — необов'язковий розділ, який використовується при групуванні даних, виводиться угорі кожної групи;
- *Примечание группы* — необов'язковий розділ, який використовується при групуванні даних, виводиться унизу кожної групи;
- *Область данных* — у цьому розділі розміщують елементи управління, які забезпечують зв'язок звіту з джерелом даних;
- *Нижний колонтитул* — вміст розділу виводиться унизу кожної сторінки;
- *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 *елементів управління*, які розміщують в області даних. Для додавання елементів до форми використовується *Панель элементов*.

Для автоматизації дії у *Access* використовуються макроси. Макросом називають набір із однієї або більше макрокоманд, що виконують певні операції. Макрокомандою називають замкнену інструкцію, яка визначає виконання деякої операції. Макроси можуть бути об'єднані у групи. Ім'я макроса задається у стовпці *Имя макроса*, який виводиться на екран за допомогою команди *Вид, Имя макроса*. Для запуску макросу з групи вказується ім'я групи, а потім після точки — ім'я макросу. При виконанні макросу у групі макросів виконується команда у стовпці макрокоманд та всі наступні макрокоманди, для яких стовпець *Имя макроса* є порожнім.

Тематика практичних завдань

1. Описати візуальні елементи управління та їх властивостей форми *Клиенты* бази даних "*Борей*".
2. Описати візуальні елементи управління та їх властивостей форми *Типы* бази даних "*Борей*".

3. Описати візуальні елементи управління та їх властивостей форми *Сотрудники* бази даних "*Борей*".
4. Описати візуальні елементи управління та їх властивостей форми *Заказы* бази даних "*Борей*".
5. Описати візуальні елементи управління та їх властивостей форми *Отчеты о продажах* бази даних "*Борей*".
6. Описати візуальні елементи управління та їх властивостей звіту "*Список товаров*" бази даних "*Борей*".
7. Описати візуальні елементи управління та їх властивостей звіту "*Продажи по сотрудникам и странам*" бази даних "*Борей*".
8. Описати візуальні елементи управління та їх властивостей звіту "*Продажи по типам*" бази даних "*Борей*".
9. Описати візуальні елементи управління та їх властивостей звіту "*Суммы продаж по годам*" бази даних "*Борей*".
10. Описати візуальні елементи управління та їх властивостей звіту "*Счет*" бази даних "*Борей*".
11. Засобами *MS Access* створити засоби уведення даних та підготовки звітів для бази даних "Облік переліку та вартості процедур, що відпускаються у медичних закладах санаторно-курортного типу" та кнопкову форму.
12. Засобами *MS Access* створити засоби введення даних та підготовки звітів для бази даних "Облік замовлень на використання виробничих ліній для таблетування різних лікарських форм" та кнопкову форму.
13. Засобами *MS Access* створити засоби введення даних та підготовки звітів для бази даних "Облік замовлень на отримання лікарських розчинів лікарнею" та кнопкову форму.
14. Засобами *MS Access* створити засоби введення даних та підготовки звітів для бази даних "Облік роботи аптечної бази" та кнопкову форму.
15. Засобами *MS Access* створити засоби введення даних та підготовки звітів для бази даних "Облік роботи дистриб'юторської мережі іноземної фармацевтичної компанії, яка розробляє власні біологічні добавки" та кнопкову форму.
16. Описати засоби автоматизації дій користувача за допомогою групи макросів "*Итоги продаж*" бази даних *"Борей"*.
17. Описати засоби автоматизації дій користувача за допомогою групи макросів "*Клиенты*" бази даних *"Борей"*.

18. Описати засоби автоматизації дій користувача за допомогою групи макросів "*Наклейки для клиентов*" бази даних *"Борей"*.
19. Описати засоби автоматизації дій користувача за допомогою групи макросів "*Поставщики*" бази даних *"Борей"*.
20. Описати засоби автоматизації дій користувача за допомогою групи макросів "*Телефоны клиентов*" бази даних *"Борей"*.
21. Створити базу даних обліку інформації про співробітників фірми. У базі даних має бути подана наступна інформація про співробітника:
    - прізвище, ім'я, по батькові,
    - посада, посадовий оклад, 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 розташування філіалу, адреса філіалу, телефон офісу.

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

    У формі *Співробітник* та *підпорядкована форма Співробітник* розмістити обчислювальне поле *"Заробітна плата"*, яка складається з посадового окладу, надбавки за вислугу років — 50 грн. за кожні 5 років служби та персональної надбавки.

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

    Створити головну кнопкову форму, з якої здійснюється 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 з 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 загалом у фірмі.

Створити звіт *Зарплата* (параметром звіту є 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 філіалу та у фірмі загалом.

Створити діалогову форму *Звіти* для друку звітів, в 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* для друку звіту та *Отмена* для виходу у головну кнопкову форму. Реалізувати виклик форми *Звіти* із головної кнопкової форми.

22. Створити базу даних обліку інформації про поставки товарів на склад. У базі даних має бути подано наступна 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 наведення 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 подання 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 наведення 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 є дата), в якому навести перелік постачальників та перелік товарів, що поставлені кожним із постачальників (назва постачальника, адреса, телефон, номер замовлення, дата, назва товару, вартість, впорядкувати за назвою за зростанням) з нумерацією товарів для кожної категорії. Визначити сумарну вартість поставлених товарів кожним постачальником та частку (у процентах) сумарної вартості поставок кожним постачальником від загальної вартості поставлених товарів.

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

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 кнопкову форму. Реалізувати виклик форми *Звіти* із головної кнопкової форми.

23. Створити базу даних обліку інформації про клієнтів фірми. У базі даних має бути наведена наступна 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 наведення інформації про клієнтів фірми,
- форму *Замовлення* для наведення інформації про замовлення фірми,
- форму *Оплата* для наведення інформації про оплату, що надійшла на рахунок фірми,
- форму *ВидиРобіт* для наведення інформації про види робіт, що виконує фірма,
- у формі *Замовлення* розмістити підпорядковану форму *Оплата* для подання інформації про здійснену оплату,
- у формі *Клієнт* розмістити підпорядковану форму *Замовлення* для подання 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 наведення сукупної сплаченої суми клієнтом за даним замовленням та поле *Борг* для наведення заборгованості за даним 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 у 3–5 порцій. 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 є 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 по тижнях) з нумерацією оплат для кожного із замовлень. Визначити загальну суму оплати за кожним тижнем та 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 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.

24. Створити базу даних обліку інформації про відправлення літаків в аеропорту. У базі даних має бути подана наступна інформація про рейс:

- номер рейсу, пункт призначення, пункт відправлення, 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 наведення інформації про рейси,
- форму *Відправлення* для подання інформації про відправлення літаків,

- форму *Пілоти* для подання інформації про пілотів, що обслуговують рейси,
- форму *ТипиЛітаків* для подання інформації про типи літаків,
- форму *ПунктиОбслуговування* для наведення переліку пунктів призначення та прибуття авіарейсів.

У формі *Рейси* розмістити підпорядковані форми *Відправлення* для наведення інформації про відправлені літаки за обраним рейсом та *Пілоти* для подання інформації про пілотів, що літали за обраним рейсом.

У формі *Відправлення* 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 наведення суми прибутковості/збитковості рейсу.

Створити головну кнопкову форму, з якої здійснювати виклик форм *Рейси, Відправлення, Пілоти, ТипиЛітаків* та *ПунктиОбслуговування*. Заповнити базу даних інформацією про 7–10 рейсів до різних пунктів 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 рейсом (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 за кожним рейсом.

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

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.

25. Створити базу даних обліку інформації про порушення при продажі товарів. У базі даних має бути подана наступна 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 наведення інформації про створені акти та підпорядковану форму *Порушення* для подання 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 наведення інформації про підприємства, форму *ТипПідприємства* для подання інформації про типи підприємства, форму *ВидиТовару* для подання інформації про види товарів.

У формі *Порушення* розмістити обчислювальне поле *"Вартість товару"*, при продажі якого виявлено порушення, та поле *"Штраф"*, значення якого розраховувати залежно від типу виявленого порушення, а саме:

- 5 % від вартості товару при таких порушеннях: *відсутність інформації про товар або не відповідає нормативним документам,*
- 10 % від вартості товару через *відсутність супровідних документів,*
- 20 % від вартості товару у зв'язку з тим, що минув *термін придатності.*

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

Створити головну кнопкову форму, з якої здійснювати виклик форм *Акти, Протокол, Підприємства, Тип Підприємства, Види Товару.* Заповнити базу даних інформацією про 7–10 актів та 3–5 порушень, що виявлені при роботі 7–10 підприємств.

Створити звіт *Штрафи Підприємств*, в якому навести перелік виявлених порушень за кожним підприємством (назва, ідентифікаційний код, прізвище керівника, номер акта, тип порушення, вартість товару, штраф, відмітка про сплату) з нумерацією порушень за кожним підприємством. Визначити сумарний розмір штрафу, що накладений на підприємство, та суму несплаченого штрафу.

Створити звіт *Порушення* (параметром звіту є період, протягом якого зафіксовано порушення — початкова дата та кінцева дата), в якому навести перелік порушень (тип порушення, вартість товару, при продажі якого виявлено порушення, штраф, вид товару, номер акта, назва підприємства і дата) з нумерацією порушень за кожним типом 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 обраний, 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.

Питання для самоконтролю та співбесіди

1. Для чого призначені форми? Опишіть способи створення форм.
2. Які розділи містить форма?
3. Для чого використовується заголовок форми?
4. Для чого використовується примітка форми?
5. Для чого використовується область даних у формах?
6. Чим відрізняються "*табличная форма*" та "*ленточная форма*"?
7. Чим відрізняються "*форма: в один столбец*" та "*ленточная форма*"?
8. Як створити форму з використанням полів кількох таблиць?
9. Як змінити джерело записів для форми?
10. Як зв'язуються головна та підпорядкована форма?
11. Що таке підпорядкована форма?
12. Опишіть способи створення підпорядкованих форм.
13. Що таке кнопкові форми? Як створити кнопкову форму у *MS Access*?
14. Як використовується "*Диспетчер кнопкових форм*"?
15. Що таке діалогові форми? Як створити діалогову форму у *MS Access*?
16. Назвіть розділи звіту та опишіть їх призначення.
17. Чим відрізняються розділи звіту: колонтитул та примітка групи?
18. Чим відрізняються розділи звіту: верхній колонтитул та заголовок?
19. Чим відрізняються розділи звіту: нижній колонтитул та примітка і заголовок звіту?
20. Які властивості можна встановити для області даних звіту?
21. Як створити обчислюване поле у звіті?
22. Як обчислити значення за групою записів у звіті?
23. Як вивести номер сторінки у звіті?
24. Як обчислити кількість записів у кожній групі звіту?
25. Як здійснити нумерацію записів у групі звіту?
26. Як додати групування у звіті?
27. Як визначити назву стовпця для групи у звіті?
28. Як обчислити середнє значення для поля за всіма записами групи?
29. Як обчислити середнє значення для поля за всіма записами звіту?

30. Як обчислити для кожного запису звіту процент, який становить значення поля до загальної суми за всіма записами звіту?
31. Назвіть етапи створення звітів за допомогою майстра.
32. Назвіть основні прийоми редагування звітів у режимі *Конструктор*.
33. Як здійснюються обчислення у звітах?
34. Як здійснюється групування записів у звітах?
35. Для чого призначені елементи управління?
36. Які елементи управління використовуються при створенні форм в *MS Access?*
37. Яка властивість елемента управління використовується для ідентифікації елемента?
38. Яка властивість елемента управління визначає, чи буде елемент відображено на екрані у режимі форми?
39. Для чого призначена властивість *Вывод на экран*?
40. Як використовується властивість *Формат поля?*
41. Які формати полів можна задавати за допомогою властивості *Формат поля?*
42. Для чого призначена властивість *Данные* ?
43. Як використовується властивість *Значение по умолчанию*?
44. Як використовується властивість *Условие на значение*?
45. Що відбудеться при порушенні умови на значення, яка визначена у полі *Условие на значение*?
46. Для якого елемента управління можна визначити властивість *Число столбцов* та як вона використовується?
47. Як використовується властивість *Ширина столбцов* ?
48. Як в *Access* можна визначити процедуру обробки події?
49. Наведіть приклад процедури обробки події для форми?
50. Наведіть приклад процедури обробки події для звіту?
51. Що таке макрос? Для чого використовуються макроси?
52. Що таке макрокоманда? Назвіть основні макрокоманди.
53. Яка макрокоманда використовується для встановлення значення властивості об'єктів?
54. Як визначити умову виконання макрокоманди?
55. Як створити макрос для перевірки коректності уведених значень?

Тема 5. 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 безліччю 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*.

Література [1; 2; 6—8; 10; 14; 16; 17; 21—24]

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 наведення 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 компонентами 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)* таблиця,
- *DatabaseName* — ім'я каталога, що містить таблицю або псевдонім *(alias)* віддаленої бази даних,
- *Fields* — масив об'єктів *TField*, використовується для звернення до полів за номером.

Компонент *TField* дає можливість 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 до списку 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 за записами таблиці є використання компонента *DBNavigator* разом з компонентом *DBGrid*. Можна також використовувати й 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 *TDBLookupComboBox* 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 формува-

тись як в 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 обліку інформації про співробітників фірми та клієнтське застосування для доступу до бази даних. У базі даних має бути наведена наступна інформація про співробітника: прізвище, ім'я, по батькові, посада, посадовий оклад, персональна надбавка для працівника, дата найому співробітника, робочий телефон, номер паспорта, дата народження, місто проживання, адреса проживання, домашній телефон, фотографія, назва філіалу, в якому працює співробітник, ідентифікаційний код філіалу, місто розташування філіалу, адреса філіалу, телефон офісу. Створити форми для введення та перегляду інформації в таблицях. У формі *Співробітник* розмістити обчислювальні поля *"Заробітна плата"*, яка складається з посадового окладу, 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 за кожним із філіалів та загалом у фірмі.

2. Створити базу даних обліку інформації про співробітників фірми та клієнтське застосування для доступу до бази даних. У базі даних має бути наведена наступна 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 для обліку інформації про товари, про постачальників, про поставки, а також для наведення 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:

- продажна ціна товару з урахуванням знижки,
- сума поставки з урахуванням знижки та вартості доставки товару,

- сумарна вартість поставлених товарів кожним із постачальників та частка (у процентах) сумарної вартості поставок кожним постачальником від загальної вартості поставлених товарів,
- загальна сплачена сума за кожною категорією та за всіма товарами загалом.

3. Створити базу даних обліку інформації про співробітників фірми та клієнтське застосування для доступу до бази даних. У базі даних має бути наведена 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 засоби введення інформації до бази даних (форми), в яких обчислити:

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

4. Створити базу даних для контролю параметрів процесу та клієнтське застосування для доступу до бази даних. У базі даних має бути наведена така інформація: шифр параметра, найменування, розмірність, мінімальне значення, максимальне значення, поточне значення, шифр апарата, найменування апарата, лінійні розміри тощо.

5. Створити базу даних для контролю успішності студентів у різних групах та клієнтське застосування для доступу до бази даних. У базі даних має бути наведена 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".

6. Створити базу даних для розрахунку стипендії студентів у різних групах та клієнтське застосування для доступу до бази даних. У

базі даних має бути наведена 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 студентський, за групами та загальний.

7. Створити базу даних для обліку роботи автотранспортного підприємства та клієнтське застосування для доступу до бази даних. У базі даних має бути наведена така інформація: табельний номер водія, прізвище, ім'я, побатькові, 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 за водіями, за класами водіїв, за типами автомобілів та загальний.

8. Створити базу даних для обліку роботи комп'ютерної фірми та клієнтське застосування для доступу до бази даних. У базі даних має бути наведена така інформація: марка комп'ютера, тип процесора, тактова частота, розмір оперативної пам'яті та жорсткого диску, ціна, початкова кількість, дата продажу, прізвище покупця, кількість проданих комп'ютерів тощо. Забезпечити виконання таких обчислень:

- кількість проданих комп'ютерів кожної марки,
- обсяг проданих комп'ютерів марки, яка визначена в параметрі,
- обсяг замовлень кожним покупцем.

9. Створити базу даних для обліку роботи міської АТС та клієнтське застосування для доступу до бази даних. У базі даних має бути наведена 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 за місяцями,

- обсяг сплати за кожним видом розмов,
- рахунок за місяць.

10. Створити базу даних для обліку роботи каси аеропорту та клієнтське застосування для доступу до бази даних. У базі даних має бути наведена 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 компонента *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++ Builder*.
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*?

СПИСОК ЛІТЕРАТУРИ

Основна

1. *Боровиков В. В.* Access 2002. — М.: Солон-Р., — 560с.
2. *Боуман Д. и др.* Практическое руководство по SQL / Д. Боуман, С. Эмерсон, М. Дарновски — К.: Диалектика, 1997. — 336 с.
3. *Методичні* вказівки до виконання лабораторних робіт "Система управління базами даних Microsoft Access: лабораторний практикум (Частина 1)", / О. В. Вітюк, А. В. Кузьмін, Н. М. Москалькова, В. В. Попов, М. Є. Сініцький, Ю. А. Тарнавський — К.: МАУП, 2003. — 166 с.
4. *Методичні* вказівки до виконання лабораторних робіт "Система управління базами даних Microsoft Access: лабораторний практикум (Частина 2)" / О. В. Вітюк, А. В. Кузьмін, Н. М. Москалькова, В. В. Попов, М. Є. Сініцький, Ю. А Тарнавський. — К.: МАУП, 2004. — 168 с.
5. *Грабер М.* Введение в SQL. — М.: Лори, 1996. — 379 с.
6. *Коннолли Т. и др.* Базы данных. Проектирование, реализация и сопровождение. Теория и практика. — М: Изд-во "Вильямс", 2000. — 1120 с.
7. *Попов В. В. и др.* Практикум і контрольні роботи з MS Access: Методичні вказівки до виконання контрольних і самостійних робіт / В. В. Попов, Л. О. Левченко, Н. М. Москалькова — К.: МАУП, 2006. — 136 с.
8. *Послед Б. С.* Access 2002. Приложения баз данных. Лекции и упражнения. — М.: DiaSoft UP. — 656с.
9. *Хомоненко А. Д. и др.* Базы данных: Учебник для высших учебных заведений/ Под ред. проф. А. Д. Хомоненко. — СПб.: КОРОНА принт, 2000. — 416 с.

10. *Роберт Сигнор, Михаэль О. Стегман.* Использование ODBC для доступа к базам данных. — М.: БИНОМ, 1995. — 384 с.

Додаткова

11. *Бекаревич Ю., Пушкина Н.* Самоучитель MicrosoftT Access 2000. — М., 1999. — 480 с.
12. *Бойко В. В., Савинков В. М.* Проектирование баз данных информационных систем. — М.: Финансы и статистика, 1989. — 351 с.
13. *Дейт К.* Введение в системы баз данных. — 7-е изд. — М.: Издат. дом "Вильямс", 2001. — 1072 с.
14. *Диго С. М.* Проектирование и использование баз данных. — М.: Финансы и статистика, 1995. — 208 с.
15. *Дрибас В. П.* Реляционные модели данных. — М.: Мир, 1992 — 192 с.
16. *Каратыгин А.* Access 2000. Руководство пользователя с примерами. — Изд-во ЛБЗ-ЮМС. — 376 с.
17. *Коупстейк С.* Access 97 шаг за шагом. — М.: Бином, 1998. — 208 с.
18. *Мартин Дж.* Организация баз данных в вычислительных системах. — М.: Мир, 1980. — 662 с.
19. *Мейер М.* Теория реляционных баз данных. — М.: Мир, 1987. — 608 с.
20. *Нагао М. и др.* Структуры и базы данных / М. Нагао, Т. Катаяма, С. Уэмура — М.: Мир, 1986. — 197 с.
21. *Парг К. и др.* Секреты Access. — К.: Диалектика, 1998. — 210 с.
22. *Ульман Д.* Основы систем баз данных. — М.: Финансы и статистика, 1983. — 334 с.
23. *Глушаков С. В., Зорянский В. Н., Хоменко С. Н.* Программирование в среде Borland C++ Builder 6. — М.:Фолио, 2003. — 508 с.
24. *Фаронов В. В.* Программирование баз данных в Delphi 7. — М., 2003. — 464 с.

ЗМІСТ

Пояснювальна записка..... 3
Тематика самостійної роботи з дисципліни “Практичні аспекти побудови баз даних”..... 7
Список літератури..... 48

Відповідальний за випуск *А. Д. Вегеренко*
Редактор *С. Г. Рогузько*
Комп’ютерне верстання *А. М. Голянда*

Зам. № ВКЦ-3801

Формат 60×84/16 . Папір офсетний.
Друк ротаційний трафаретний. Наклад 30 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