Міністерство освіти і науки України
Вінницький національний технічний університет

Методичні вказівки до лабораторних робіт з дисципліни «Бази даних і знань» для студентів напряму підготовки «Управління інформаційною безпекою»

Вінниця
ВНТУ
2017

Затверджено Методичною радою Вінницького національного технічного університету Міністерства освіти і науки України як «Методичні вказівки. Електронне видання» та рекомендовано до використання в навчальному процесі(протокол № 7 від 29.03.2017 р.)

Рецензенти:
В. В. Карпінець, кандидат технічних наук, доцент
Д. І. Катєльніков, кандидат технічних наук, доцент

Методичні вказівки до лабораторних робіт з дисципліни «Бази даних і знань» для студентів напряму підготовки «Управління інформаційною безпекою» / Уклад. Ю.Є. Яремчук, Д.П. Присяжний, І. О. Дьогтєва, О. В. Салієва – Вінниця: ВНТУ, 2017. – 69 с.

У даних методичних вказівках до лабораторних робіт наводяться основні рекомендації до вивчення, підготовки та проведення лабораторних робіт з дисципліни «Бази даних і знань» та організації самостійної роботи студентів.

ЗМІСТ

ВСТУП

Лабораторні роботи є сполучною ланкою між лекційними заняттями і самостійною роботою студентів. В процесі виконання лабораторних робіт експериментально перевіряються ключові питання курсу «Бази данихі знань», набуваються практичні навички проектування і розробки баз даних засобами сучасних систем управління базами даних, перевіряється рівень засвоєння основних положень предмету.

Метою методичних вказівок є надання допомоги студентам в отриманні практичних навичок роботи з системами керування базами даних.

У серії лабораторних робіт використовуються Microsoft ACCESS, Microsoft SQL Server 2005, Microsoft Visual C # 2005 Express Edition. Важливою складовою частиною робіт є освоєння SQL стандарту.

Під час виконання лабораторних робіт студенти отримають навички щодо:

- ✓ підходів до проектування баз даних;
- ✓ створення таблиць;
- ✓ створення форм та організації взаємозв’язку між формами та таблицями;
- ✓ створення запитів на отримання даних з таблиць;
- ✓ створення звітів на основі таблиць і запитів.

Студент зобов'язаний до лабораторного заняття прочитати методичні вказівки до лабораторної роботи і спробувати виконати її самостійно. Під час лабораторного заняття студент показує викладачеві результати роботи, проводить консультації з питань, які виникли, та завершує роботу.

Захист роботи полягає в виконанні завдання до лабораторної роботи, відповіді на питання по темі лабораторної роботи і внесення деяких змін в базі даних, яка розроблялась,в присутності викладача.

Результати виконання робіт рекомендується зберігати в особистих папках, так як лабораторні роботи взаємопов'язані.

ЛАБОРАТОРНА РОБОТА № 1
СТВОРЕННЯ ФАЙЛУ БАЗИ ДАНИХ, ЙОГО РЕДАГУВАННЯ ТА КОРИГУВАННЯ

Мета:навчитись створювати та редагувати файл БД: вивчити структуру об'єкта "таблиця"; навчитися задавати полям різні типи даних, виконувати операції в таблицях; навчитися працювати з фільтрами.

1.1 Теоретичні відомості

1.1.1 Визначення та класифікація БД

База даних (БД) — впорядкований набір логічно взаємопов'язаних даних, що використовуються спільно та призначені для задоволення інформаційних потреб користувачів. У технічному розумінні включно й система керування БД.

Головне завдання БД — гарантоване збереження значних обсягів інформації (так звані записи даних) та надання доступу до БД користувачеві або ж прикладній програмі. Таким чином, БД складається з двох частин: збереженої інформації та системи керування.

БД можна класифікувати за різними ознаками (табл. 1.1).

Таблиця 1.1 —Класифікація БД

Класифікація БД:	*за моделлю даних*:	• ієрархічні; • мережеві; • реляційні; • об'єктні; • об'єктно-орієнтовані; • об'єктно-реляційні;
	за технологією фізичного зберігання:	• БД у вторинній пам'яті (традиційні); • БДвоперативнійпам'яті (in-memory databases); • БДутретиннійпам'яті (tertiary databases);
	за ступенем розподіленості:	• централізовані (зосереджені); • розподілені;
	за застосуванням мови програмування:	• відкриті (спираються на одну з універсальних мов); • замкнуті (використовується власна мова програмування);
	за функціями, які виконуються:	• інформаційні; • операційні;
	за сферою застосування:	• універсальні; • спеціалізовані;
	за «потужністю»:	• корпоративні; • настільні;
	за способом доступу:	• з локальним доступом; • з віддаленим (мережевим) доступом.

Окреме місце в теорії та практиці займають просторові (англ. spatial), тимчасові, або темпоральні (temporal) і просторово-часові (spatial-temporal) БД.

Системи централізованих БД з віддаленим (мережевим) доступом можуть допускати різні архітектури подібних систем:

- „файл-сервер",

- „клієнт-сервер”.

1.1.2 Коротка характеристика деяких СУБД

Створення БД, обробка та пошук необхідної інформації в ній здійснюється за допомогою **системи управління базами даних** (СУБД). СУБД — це набір певних програмних засобів, які надають можливість користувачеві швидко та ефективно взаємодіяти з БД.

MySQL— вільна СУБД. MySQL є власністю компанії Oracle Corporation, що отримала її разом з поглиненою Sun Microsystems, яка здійснює розробку і підтримку додатку. Розповсюджується під GNU General Public License і під власною комерційною ліцензією. Крім цього розробники створюють функціональність на замовлення ліцензійних користувачів, саме завдяки таким замовленням з'явився механізм реплікації.

MySQL із самого початку була дуже схожою на mSQL, проте з часом вона розширювалася і зараз MySQL — одна з найпоширеніших СУБД.

Використовується, у першу чергу, для створення динамічних веб-сторінок, оскільки має підтримку з боку різноманітних мов програмування.

MySQL є рішенням для малих і середніх додатків. Зазвичай MySQL використовується як сервер, до якого звертаються локальні або віддалені клієнти, проте до дистрибутиву входить бібліотека внутрішнього сервера, що дозволяє включати MySQL до автономних програм. Вихідні коди сервера компілюються на багатьох платформах.

Гнучкість СУБД MySQL забезпечується підтримкою великої кількості типів таблиць: користувачі можуть вибрати як таблиці типу MyISAM, що підтримують повнотекстовий пошук, так і таблиці InnoDB, що підтримують транзакції на рівні окремих записів. СУБД MySQL поставляється із спеціальним типом таблиць EXAMPLE, що демонструє принципи створення нових типів таблиць. Завдяки відкритій архітектурі, GPL-ліцензуванню, в СУБД MySQL постійно з'являються нові типи таблиць. MySQL характеризується великою швидкістю, стійкістю і простотою.

PostgreSQL— об'єктно-реляційна СУБД. Є альтернативою як комерційним СУБД (Oracle Database, Microsoft SQL Server, IBM DB2 та інші), так і СУБД з відкритим кодом (MySQL, Firebird, SQLite).

Порівняно з іншими проектами з відкритим кодом, такими як Apache, FreeBSD або MySQL, PostgreSQL дана СУБД не контролюється однією компанією, її розробка можлива завдяки співпраці багатьох людей та компаній, які бажають використовувати дану СУБД та впроваджувати в неї найновіші досягнення.

СУБД **Oracle**— це найпотужніший програмний комплекс, що дозволяє створювати додатки будь-якої складності. Ядром цього комплексу є БД, що зберігає інформацію, кількість якої за рахунок наданих засобів масштабування практично необмежена. З високою ефективністю працювати з відповідною інформацією одночасно може практично будь-яка кількість користувачів (за наявності достатніх апаратних ресурсів)без тенденції до зниження продуктивності системи при різкому збільшенні їх кількості.

Механізми масштабування в СУБД Oracle останньої версії дозволяють збільшувати потужність і швидкість роботи сервера Oracle, додатків, додаючи нові вузли кластеру. Це не вимагає зупинки працюючих додатків, переписування старих додатків, розроблених для звичайної одномашинної архітектури. Крім того, вихід з ладу окремих вузлів кластера також не призводить до зупинки програми.

Вбудовування до СУБД Oracle JavaVM повномасштабної підтримки серверних технологій (Java Server Pages, Java-сервлети, модулі Enterprise JavaBeans, інтерфейси прикладного програмування CORBA), призвели до того, що Oracle де-факто є стандартом СУБД для Internet.

СУБД Oracle поставляється практично для всіх існуючих операційних систем. Працюючи під Sun Solaris, Linux, Windows або на інший операційній системі з продуктами Oracle не буде виникати проблем у роботі. СУБД Oracle однаково добре працює на будь-якій платформі. Таким чином, компаніям, які розпочинають роботу з продуктами Oracle не доводиться змінювати мережеве оточення. Існує лише невелика кількість відмінностей при роботі з СУБД, обумовлених операційною системою.

Microsoft SQL Server— система управління реляційними БД, розроблена корпорацією Microsoft. Основна мова запитів— Transact-SQL, створена спільно Microsoft та Sybase. Transact-SQL є реалізацією стандарту ANSI/ISO щодо структурованої мови запитів (SQL) із розширеннями. Використовується для роботи з БД розміром від персональних до великих БД масштабу підприємства, конкурує з іншими СУБД у даному сегменті ринку.

При взаємодії з мережею Microsoft SQL Server і Sybase ASE використовують протокол рівня додатків під назвою Tabular Data Stream (TDS, протокол передачі табличних даних). Протокол TDS також був реалізований у проекті FreeTDS з метою забезпечити різні додатки можливістю взаємодії з БД Microsoft SQL Server і Sybase.

Для забезпечення доступу до даних Microsoft SQL Server підтримує Open Database Connectivity (ODBC) — інтерфейс взаємодії додатків з СУБД. SQL Server надає можливість підключення користувачів через веб-сервіси, що використовують протокол SOAP. Це дозволяє клієнтським програмам, не призначеним для Windows, кросплатформно з'єднуватися з SQL Server.

Microsoft Office Accessабо просто Microsoft Access - реляційна СУБД корпорації Microsoft. Основні компоненти MS Access:

- будівник таблиць;
- будівник екранних форм;
- будівник SQL-запитів (мова SQL в MS Access не відповідає стандарту ANSI);
- будівник звітів, що виводяться на друк.

Кожна нова версія Access все тісніше інтегрувалася з іншими програмними продуктами, що входять в Office. Важливим вдосконаленням стало те, що всі продукти Office і Visual Basic використовують спільну мову програмування Visual Basic for Applications (VBA).

Істотно розширює можливості MSAccess з написання додатків механізм зв'язку з різними зовнішніми СУБД: "зв'язані таблиці" (зв'язок з таблицею СУБД) і "запити до сервера" (запит на діалекті SQL, який "розуміє" СУБД). Також MS Access дозволяє будувати повноцінні клієнт-серверні додатки на СУБД MS SQL Server; є можливість поєднати інструменти для управління БД і засоби розробки.

Варіанти, які пропонуються для створення нової БД:

- ✓ створення БД на основі шаблону;
- ✓ створення БД без використання шаблону;
- ✓ копіювання даних з іншого джерела до таблиці Access;
- ✓ імпортування, додавання та зв’язування з даними з іншого джерела;
- ✓ відкриття наявної бази даних Access;
- ✓ створення настроюваного пустого шаблону.

1.1.3Структура, способи створення таблиць, типи полів у таблицях

Створення бази починається зі створення першої таблиці.

Таблиці БД–це об'єкти(двомірні таблиці), в яких безпосередньо зберігаються (містяться) дані.

Кожна таблиця складається із записів (рядків) та полів (стовпців).

Поле– стовпець таблиці, призначений для зберігання значень певної властивості (параметра) об'єкта. Значення поля часто називають фактом.

Запис– рядок таблиці. Один запис містить дані про окремий об´єкт, який описують у БД.

Поле, яке містить унікальні значення, що не повторюються в жодному із записів і не є порожнім, називається **ключовим**.

Існує декілька способів створення нової таблиці, що відрізняються рівнем автоматизації (табл.1.2):

Таблиця 1.2–Способи створення таблиці

Спосіб	Опис	Рівень атоматизації
Імпорт таблиць	імпорт даних з інших файлів	"найавтоматичніший" спосіб, полягає в імпорті таблиць з іншої бази (можливо створеної в іншій системі); в залежності від обставин з імпортованої таблиці може надійти структура полів, їх назви, властивості, вміст бази; якщо імпортується некоректно, внести виправлення.
Зв'язок з таблицями	встановлення зв'язків з даними, що зберігаються на іншому сервері	у випадках, коли мова йде про таблицю, що знаходиться на віддаленому сервері і яку не можна імпортувати повністю
Майстер таблиць	дозволяє вибрати поля, що включаються в таблицю, із широкого списку зразків полів різних типів	користуються досвідчені розроблювачі, оскільки варто володіти відповідною термінологією: "Мастер таблиц" задає ряд питань і, керуючись отриманими відповідями, створює структуру таблиці автоматично.
Режим таблиці	виводить бланк абстрактної таблиці, яка потім може приймати конкретні форму і зміст	відкриває заготовку, у якій всі поля мають формальні імена: Поле1, Поле2... і т.д. і один стандартний текстовий тип; дану таблицю можна відразу заповнювати інформацією.
Конструктор	можна безпосередньо вказати параметри елементів таблиці	найбільш універсальний, «ручний» метод, можна самостійно задати імена полів, вибрати тип і налаштувати властивості.

Тип даних визначається множиною значень, які може приймати дане поле в різних записах (табл. 1.3).

Таблиця 1.3–Типи даних

Тип	Опис
1	2
Текстовий	для введення тексту довжиною до 255 символів (за замовчуванням)
Поле МЕМО	для введення заміток або довгих описів
Числовий	для введення числових даних

Продовження таблиці 1.3

1	2
Дата/час	для введення часу і дати
Грошовий	використовується для роботи з грошовими величинами
Лічильник	для введення числа, яке автоматично збільшується на одиницю при додаванні в таблицю нового запису
Логічний	для збереження логічного значення Так або Ні
Об'єкти OLE	використовуються для збереження в таблиці OLE-об'єктів

	(наприклад, малюнків, звуків, документів Word та ін.)
Гіперпосилання	для запису в таблицю гіперпосилань

1.1.4 Прийоми роботи з таблицями БД, фільтри

На рис.1.1 подана типова таблиця БД, з якою можна працювати звичайними прийомами керування за допомогою миші.

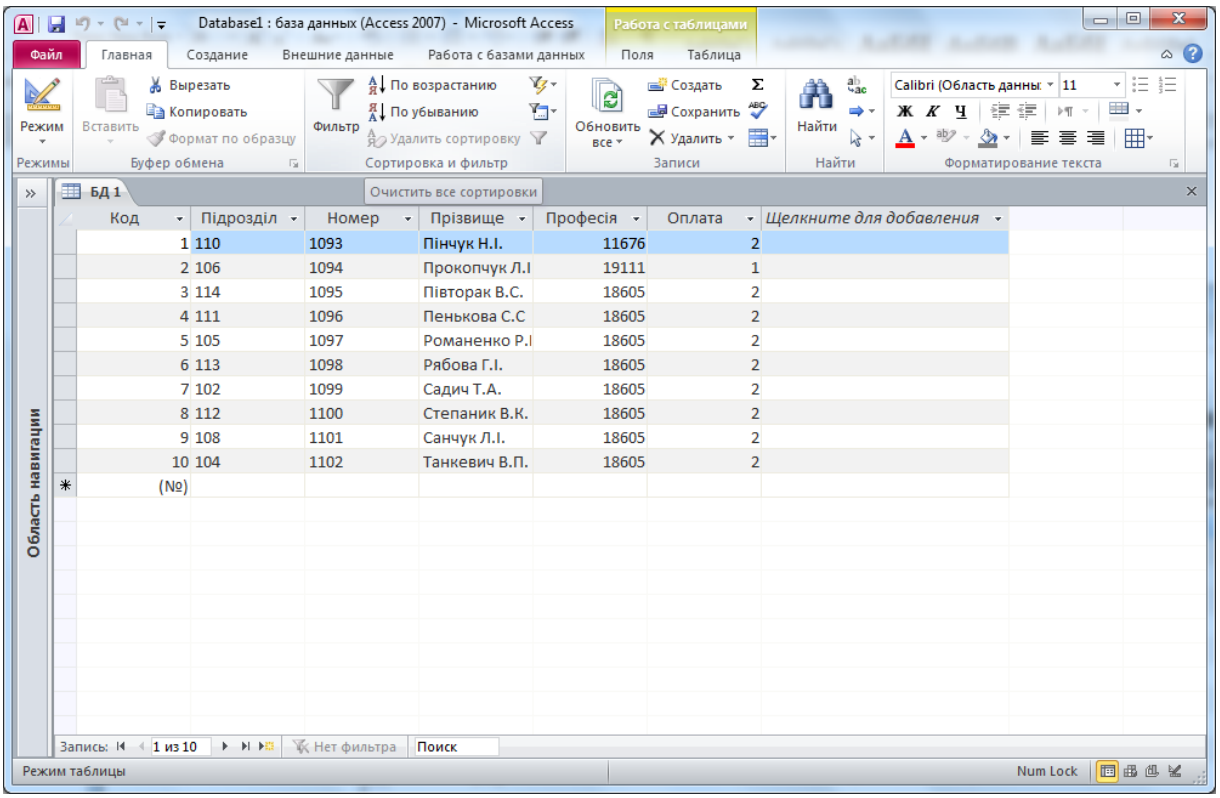

Рисунок 1.1

Зверніть увагу на рядок стану в нижній частині вікна. У Access 9x цей рядок називається полем номеру запису. Це поле містить кнопки переходу, за допомогою яких можна ефективно переміщувати по таблиці. А також в цій панелі знаходиться зручний пошук по БД.

Кожний запис має зліва кнопку *(маркер запису).* Натиснення лівої клавіші миші на цьому маркері виділяє весь запис і готує його до копіювання, переміщення, видалення.

Натиснення правою кнопкою на виділеному записі відкривається контекстне меню для операцій із записом.

Маркер, що знаходиться в лівому верхньому кутку таблиці– це маркер таблиці. Натиснення лівої кнопки миші виділяє всю таблицю, а правої – відкриває контекстне меню для операцій із таблицею в цілому.

Поля БД подані в таблиці стовпцями. Кожний стовпець має заголовок, у якому записане ім'я поля або те значення, яке задане у властивості «**Підпис**».

Якщо вміст поля не повністю розміщується в комірці таблиці, стовпець можна розширити. При наведенні покажчика миші на границю між стовпцями покажчик змінює форму. Тепер границю можна переміщувати методом перетягування, а подвійне натиснення лівої кнопки миші, виконане у цей момент, автоматично встановлює ширину стовпця рівній довжині найдовшого значення в даному полі.

Натиснення лівої кнопки миші на заголовку стовпця виділяє весь стовпець, а натиснення правою кнопкою на виділеному стовпці відкриває контекстне меню. У ньому є пункти, що дозволяють відсортувати записи по даному полю, вставити новий стовпець, сховати стовпець та інше.

Схований стовпець не зникає з бази, а тільки перестає відображатися на екрані. Щоб знову його відобразити, треба навести покажчик на границю між стовпцями в тому місці, де був схований стовпець, і виконати подвійне натиснення лівої кнопки миші. Схований стовпець знову стане видимим.

Особливості зберігання

Таблиці БД не є самостійними документами. Сама база - це документ. Їй відповідає файл на диску, і ми можемо зробити його копію. Структура таблиць - теж документ. У деяких системах вона має окремий файл, а в деяких (наприклад, в Access 9x) такого файла немає, але структура таблиць входить до складу загального файла БД поряд із запитами, формами, звітами й іншими об'єктами. При зміні структури таблиці система керування БД завжди видає запит на зберігання змін.

Вміст таблиць не можна зберегти примусовою командою або, навпаки, відмовитися від його зберігання. Всі зміни в таблицях зберігаються автоматично в режимі реального часу. Режим реального часу означає, що, поки ми працюємо з таблицею, відбувається її безупинне зберігання. Як тільки завершується введення даних в одне поле і відбувається перехід до наступного поля, дані негайно записуються на жорсткий диск. Тому, експериментуючи з таблицями, треба знати, що всі зміни, що вносяться в їх зміст, мають необоротний характер. Не можна щось змінити, видалити, а потім відмовитися від зберігання і повернутися до вихідного варіанту.

Відсутні та пропущені рядки

В процесі створення БД деякі поля можуть залишитись незаповненими внаслідок невідомого їх значення або за відсутності даних. Якщо залишити поле пустим, наприклад при відсутності даних, тоді йому буде присвоєне значення NULL. Відсутність значення у даного поля називається пустим рядком і позначається введенням двох парних лапок (” ”).

Спеціальний текст, який інформує про наявність поля, в якому міститься величина NULL або пустий рядок вводиться в поле властивостей ”**Формат** ” поля.

Робота з фільтрами

Фільтри призначені для вибору та відображення даних за певним критерієм. Для роботи в Microsoft Access передбачено кілька готових фільтрів. Вони доступні у вигляді команд меню в режимах таблиці, макета, в представленні форми та звіту. Доступні типи фільтрів:

- **Звичайні фільтри**: використовуються для фільтрації за значенням або діапазону значень.
- **Фільтрація по виділеному**: дозволяє відсортувати всі рядки в таблиці, що містять значення, яке збігається з виділеним значенням в рядку. Використовується в режимі таблиці.
- **Фільтр по формі**: використовується, якщо потрібно відфільтрувати кілька полів у формі або таблиці або знайти певний запис.
- **Розширений фільтр**: дозволяє задавати декілька умов для фільтрації та гнучко їх налаштувати.

1.2 Завдання до лабораторної роботи

1. Створіть файл БДdatabase.mdb, в якому містяться дані про кадровий склад підприємства.

Назви полів та їх типи наведені в табл. А.1 (Додаток А).

2. Введіть дані в БД відповідно до варіанту, користуючись табл. А.2 (Додаток А).
3. Спробуйте вилучити групу записів(пам’ятайте, що відмінити видалення неможливо, тому у діалоговому вікні, що підтверджує вилучення варто відхилити відповідну дію).
4. Додайте кілька записів в таблицю.
5. Сховайте якийсь із столпців таблиці, а потім зробіть його знову видимим.
6. Відсортуйте записи в таблиці по полям “Прізвище”, “Ставка”, “Дата народження” по зростанню.
7. Відмініть сортування.
8. Знайдіть за допомогою функції пошуку записи, у яких значення поля “Професія” дорівнює 18605.
9. Замініть значення поля “Професія”, що дорівнює 18605 на 17890, використовуючи функції заміни.
10. За допомогою фільтра отримайте список робітників віком від 20 до 40 років (попередньо введіть відповідні дані в таблицю).

1.3 Контрольні запитання

1. Що називається БД?
2. Поясніть термін СУБД та основні функції СУБД?
3. Які структури ACCESS ви знаєте?
4. Для чого служить структура "таблиця"?
5. Чим поле відрізняється від запису?
6. Опишіть процес створення нової та редагування структури існуючої таблиці.
7. Для чого даним задаються різні типи?
8. Для чого використовуються фільтри? Які типи фільтрів присутні в Microsoft Access?

ЛАБОРАТОРНА РОБОТА № 2
СТВОРЕННЯ ЗАПИТІВ

Мета роботи: навчитися створювати запити.

2.1 Теоретичні відомості

2.1.1 Необхідність запитів

Запит– це набір інструкцій, який можна використовувати для роботи з даними. Окрім повернення результатів, які можна сортувати, групувати або фільтрувати, за допомогою запиту також можна створювати, видаляти, копіювати або змінювати дані.

Наприклад, припустимо, що на великому підприємстві є БД “Кадри”, що містить найдокладніші відомості про кожного співробітника. Крім формальної інформації база може містити і конфіденційну. Інформація зберігається в базових таблицях. Працювати з БД “Кадри” можуть різні підрозділи підприємства, і усім їм потрібні різні дані. Не все те, що дозволено знати службі безпеки підприємства, повинно бути доступно головному лікарю, і навпаки. Тому доступ користувачів до базових таблиць закривають.Для доступу до даних є інший, гнучкий і зручний засіб - запити. Для однієї і тієї ж таблиці можна створити багато різних запитів, кожний із яких зможе добувати з таблиці лише частину інформації, яка необхідна у даний момент.

У результаті роботи запиту з загальної вихідної бази формується **результуюча таблиця**, що містить частину загальної інформації.

Важливою властивістю запитів є те, що при створенні результуючої таблиці можна не тільки вибирати інформацію з бази, але й обробляти її. При роботі запиту дані можуть упорядковуватися (сортуватися), фільтруватися (відсіюватися), об'єднуватися, розділятися, змінюватися, і при цьому ніяких змін у базових таблицях може не відбуватися. Результати обробки позначаються тільки на змісті результуючої таблиці, вона має тимчасовий характер, і іноді її навіть називають *моментальним знімком*.

Властивістю запитів є можливість виконувати підсумкові обчислення. Запит може не тільки видати результуючу таблицю, але і знайти, наприклад, середнє (найбільше, найменше, і т.п.) значення по відповідному полю.

2.1.2 Вибір базових таблиць для запиту

Створення запиту до бази починається з відкриття вкладки “**Створити**” і натиснення лівої клавіші миші на кнопці “ **Створити**”.

Створення запиту в режимі “**Конструктора**” починають із вибору тих таблиць бази, на яких буде заснований запит.

Вибір таблиць виконують у діалоговому вікні “**Додавання таблиці**”. У ньому відображаються всі таблиці, наявні в базі.Обрані таблиці заносять у верхню половину бланка “запиту за зразком”натисненням лівої клавіші миші на кнопці “**Додати**”.

У вікні “**Додавання таблиці**” зверніть увагу на наявність вкладок: “**Таблиці**”, “**Запити**”, “**Запити і таблиці**” (запит не обов'язково створювати тільки на основі таблиць).

2.1.3 Бланк запиту за зразком

Бланк запиту за зразком має дві панелі. На верхній панелі розташовані списки полів тих таблиць, на яких формується запит. Рядки нижньої панелі визначають структуру запиту, тобто структуру результуючої таблиці, у якій будуть міститися дані, отримані за результатами запиту.

Рядок “**Поле**” заповнюють перетягуванням назв полів із таблиць у верхній частині бланка. Кожному полю результуючої таблиці відповідає стовпець бланка запиту за зразком.

Рядок “**Ім’я таблиці**” заповнюється автоматично при перетягуванні поля.

Якщо натиснути на рядок “**Сортування**”, з'явиться кнопка списку, що розкривається, який містить види сортування. Якщо призначити сортування по якомусь полю, дані в результуючій таблиці будуть відсортовані по цьому полю.

Бувають випадки, коли поле повинне бути присутнім у бланку запиту за зразком, але не повинно відображатися в результуючій таблиці. У цьому випадку можна заборонити його виведення на екран, скинувши відповідний прапорець.

Найцікавіший рядок у бланку запиту за зразком називається “**Умови відбору**”, де записують ті критерії, по якому вибирають запис для включення в результуючу таблицю. По кожному полю можна створити свою умову відбору.

Запуск запиту виконують натисненням лівої клавіші миші на вкладці “**Конструктор**”**,** у групі “**Результати**” клацніть елемент “**Виконати**”, утвориться результуюча таблиця.

2.1.4 Обчислення в запитах

Перед тим, як створювати і використовувати поля, що обчислюються, варто звернути увагу на те, що поле, що обчислюється, існує тільки в результуючій таблиці. У вихідних таблицях таке поле не створюється, і при роботі звичайного запиту таблиці не змінюються (кожний, хто звертається до бази, може за допомогою запитів як завгодно маніпулювати даними й одержувати будь-які результати, але при цьому вихідні таблиці залишаються незмінно однаковими для всіх користувачів).

Для створення запиту, що робить обчислення, служить бланк запиту за зразком. Різниця тільки в тому, що в одному із стовпців замість імені поля записують формулу. У формулу входять поміщені в квадратні дужки назви полів, що беруть участь у розрахунку, а також знаки математичних операцій.

У вузький стовпець непросто записати довгу формулу, але якщо натиснути комбінацію клавіш SHIFT+F2, то відкривається допоміжне діалогове вікно, що називається “**Область введення**”, в якомуможна ввести формулу, а потім натисненням лівої клавіші миші на кнопці ОК перенести її в бланк запиту за зразком.

Якщо включити відображення поля, що обчислюється, результати розрахунків будуть видаватися в результуючій таблиці.Можна зробити поле,що обчислюється, полем сортування, щоб не тільки одержувати нові результати, але й аналізувати їх.

2.1.5 Запити на вибірку

Мета запиту на вибірку полягає в створенні результуючої таблиці, у якій відображаються тільки потрібні за умовою запиту дані з базових таблиць.

Запит на вибірку – запит, в якому добираються певні дані з таблиць і повертається результивний набір у вигляді об'єкта в режимі таблиці, при цьому дані не змінюються.

Запити на вибірку можна також використовувати для групування записів і обчислення сум, середніх значень, підрахунку записів і знаходження інших типів підсумкових значень.

2.1.6 Запити з параметром

Запити з параметрами – це запит, який при виконанні відображає у власному діалоговому вікні запрошення ввести дані, наприклад умова для повернення записів або значення, яке потрібно вставити в поле.

Можна розробити запит, що виводить запрошення на введення декількох одиниць даних, наприклад двох дат, потім Microsoft Access може повернути всі записи, що припадають на інтервал часу між цими датами. Запити з параметрами також зручно використовувати в якості основи для форм, звітів і сторінок доступу до даних.

Наприклад, припустимо, що в БД є таблиця, у якій містяться всі результати чемпіонатів світу з футболу. Наша задача: створити запит, за допомогою якого користувач може визначити, у якому році та або інша команда займала перше місце, причому вибір цієї команди - його особиста справа.

Для цієї мети служить спеціальна команда мови SQL, що виглядає так: LIKE [...]. У квадратних дужках можна записати будь-який текст, звернений до користувача.

Команду LIKE треба помістити в рядку "**Условие отбора**" і в те поле, по якому робиться вибір. У нашому випадку це стовпець збірних, що займали перші місця в чемпіонатах світу з футболу.

Після запуску запиту відкривається діалогове вікно, у якому користувачу пропонується ввести параметр.Якщо в якості параметра ввести слово "Бразилія", то видається результуюча таблиця, що містить запису по тим чемпіонатам, коли збірна Бразилії ставала чемпіоном.Якщо ввести слово "Італія", то результуюча таблиця буде іншою.

2.1.7 Запити на зміну

Вище сказано, що всі види запитів на вибірку створюють тимчасові результуючі таблиці, при цьому базові таблиці не змінюються. Проте, спеціально для розробників БД існує особлива група запитів - запити на зміну, які дозволяють автоматично створювати нові таблиці або змінювати вже наявні.

Запити на зміну – запит, який за одну операцію змінює або переміщує кілька записів.Існують запити:

На видалення запису – видаляє групу записів з однієї або декількох таблиць (можна видаляти тільки весь запис, а не окремі поля всередині нього).

На оновлення запису – вносить загальні зміни в групу записів однієї або декількох таблиць(дозволяє змінювати дані в існуючих таблицях).

На додавання записів – додає групу записів з однієї або декількох таблиць в кінець однієї або декількох таблиць.

На створення таблиці – створює нову таблицю на основі всіх або частини даних з однієї або декількох таблиць. Запит на створення таблиці корисний при створенні таблиці для експорту в інші бази даних **Microsoft Access** або при створенні архівної таблиці, яка містить старі записи.

На об'єднання– дозволяє об'єднати дані з двох таблиць з аналогічними структурами.

Логіка використання запитів на зміну така:

Створюється запит на вибірку, що відбирає дані з різних таблиць або створює нові дані шляхом обчислень.Після запуску запиту утвориться тимчасова результуюча таблиця.Дані з цієї тимчасової таблиці використовують для створення нових таблиць або зміни існуючих.

Наприклад, розглянемо запит на створення таблиці. Припустимо, що розробник таблиці "Підсумки по командах" захотів включити в неї поле "Результативність". Звичайно, він може розрахувати середню кількість м'ячів, забитих за гру кожній командою, але якщо ввести в таблицю таке поле, то доведеться заповнювати його вручну. Для таблиць, що містять багато записів, це рішення неприйнятне.Простіше створити запит на вибірку, у який увійдуть усі поля базової таблиці плюс нове, яке обчислюється, поле.

Натиснення лівої клавіші миші на кнопці "**Виконати**" дозволяє переконатися, що запит працює як треба і створює результуючу таблицю, більш повну ніж базова. Тепер можна дати команду на створення нової базової таблиці, рівній результуючій: вкладка"**Створити**" у пункті "**Запит**",що доступно тільки в режимі "**Конструктора**".

У тому ж меню присутні команда для створення запитів на відновлення даних, на додавання записів і на вилучення записів.

2.1.8 Перехресні запити

Перехресні запити – запити, які використовуються для розрахунків і представлення даних у структурі, що полегшує їх аналіз. Перехресний запит підраховує суму, середнє, кількість значень або виконує інші статистичні розрахунки, після чого результати групуються у вигляді таблиці за двома розділами даних, один з яких визначає заголовки стовпців, а інший – рядків.

Способи створення перехресного запиту

- Використання *майстра* перехресних запитів, дебільшість роботи він виконує самостійно, але деякі параметри в майстрі відсутні.
- Робота в режимі *конструктора*, який надає повніший контроль над структурою запиту та підтримує функції, які недоступні в майстрі.
- Створення запиту *в режимі SQL*.Проте в режимі SQL не можна вказувати типи даних параметрів. Якщо в перехресному запиті потрібно використовувати параметр, слід вказати тип даних параметра, змінивши запит у режимі конструктора.

Створення перехресних запитів за допомогою майстра

Для роботи з майстром перехресних запитів слід використовувати окрему таблицю або запит як джерело записів для перехресного запиту. Якщо окрема таблиця не містить усі дані, які потрібно включити до перехресного запиту, спочатку створіть вибірковий запит, який повертає потрібні дані.

1. На вкладці "**Створити**" у групі "**Макроси та код**" натисніть кнопку "**Майстер запитів**".
2. У діалоговому вікні "**Новий запит**" виберіть пункт "**Майстер перехресних запитів**" і натисніть кнопку ОК.
3. На першій сторінці майстра виберіть таблицю або запит, які потрібно використовувати для створення перехресного запиту.
4. На наступній сторінці виберіть поле, що містить значення, які потрібно використовувати як заголовки рядків.
5. На наступній сторінці виберіть поле, що містить значення, які потрібно використовувати як заголовки стовпців.
6. Якщо для заголовків стовпців вибрати поле «Дата/час», на наступній сторінці майстра буде запропоновано вказати інтервал для групування дат. Можна вказати Рік, Квартал,

Місяць, Дата або Дата/час. Якщо для заголовків стовпців не вибрано поле «Дата/час», майстер пропустить цю сторінку.

7. На наступній сторінці виберіть поле та функцію, які будуть використовуватися для обчислення зведених значень. Вибраний тип даних поля визначає доступні функції.

Тип даних поля- характеристика поля, яка визначає, які дані можуть зберігатися в ньому.

Наприклад, поле з текстовим типом даних може містити як текст, так і числа, а поле зчисловим типом даних — лише числові дані.

8. На тій самій сторінці встановіть або зніміть прапорець **Так**, включати суми рядків, щоб включити (запит матиме додатковий заголовок рядка, який використовує те саме поле та функцію, що й значення поля) або виключити суми рядків (буде вставлено додатковий стовпець, який підсумовує решту стовпців).

Наприклад, якщо перехресний запит обчислює середній вік за розташуванням і статтю (з заголовками стовпців статі), додатковий стовпець обчислює середній вік за розташуванням для всіх статей.

9. На наступній сторінці майстра введіть ім’я запиту, а потім вкажіть, чи потрібно переглянути результати або змінити структуру запиту.

Створення перехресного запиту в режимі конструктора

Для створення перехресного запиту в режимі конструктора можна використовувати будь-яку кількість джерел записів (таблиць і запитів). Проте структуру можна зробити простішою, створивши спочатку вибірковий запит, який повертає всі потрібні дані, а потім використовувати цей запит як єдине джерело записів для перехресного запиту.

1. На вкладці “**Створити**” у групі “**Макроси та код**” натисніть кнопку “**Конструктор запиту**”.

2. У діалоговому вікні “**Відображення таблиці**” двічі клацніть кожну таблицю або запит, які потрібно використовувати як джерело записів.

Якщо використовується кілька джерел записів, переконайтеся, що таблиці або запити об'єднаніза спільними для них полями. *Об'єднання* - зв'язок між полем однієї таблиці або запиту й полем іншої таблиці або запиту, які мають однаковий тип даних. Невідповідні записи можуть бути як включені, так і виключені, залежно від типу об'єднання.

3. Закрийте діалогове вікно Відображення таблиці.

4. На вкладці “**Конструктор**” у групі “**Тип запиту**” натисніть кнопку “**Перехресний**”.

5. У вікні конструктора запитів двічі клацніть кожне поле, яке потрібно використовувати як джерело заголовків рядків. Для заголовків рядків можна вибрати до трьох полів.

6. У бланку запиту в рядку “**Перехресний**” для кожного поля заголовка рядка виберіть пункт “**Заголовок рядка**”.

Щоб обмежити результати для цього поля, можна ввести вираз у рядку “**Критерії**”. Крім того, можна визначити порядок сортування для поля в рядку “**Сортування**”.

7. У вікні конструктора запитів двічі клацніть поле, яке потрібно використовувати як джерело заголовків стовпців. Для заголовків стовпців можна вибрати лише одне поле.

8. У бланку запиту в рядку “**Перехресний**” для поля заголовка стовпця виберіть пункт “**Заголовок стовпця**”.

Щоб обмежити результати для поля заголовка стовпця, можна ввести вираз у рядку “**Критерії**”. Проте використання виразу критерію з полем заголовка стовпця не обмежує кількість стовпців, повернутих перехресним запитом. Натомість воно визначає, які стовпці містять дані.

Наприклад, поле заголовка стовпця, яке має три можливі значення: червоний, зелений і синій. Якщо застосувати критерій ='синій' до поля заголовка стовпця, у перехресному запиті й надалі відображатимуться стовпець для червоного та стовпець для зеленого, але лише стовпець для синього міститиме дані.

*Примітка.*Якщо потрібно обмежити значення, що відображаються як заголовки стовпців, можна вказати список фіксованих значень за допомогою властивості запиту "**Заголовки стовпців**".

1. У вікні конструктора запитів двічі клацніть поле, яке потрібно використовувати для обчислення зведених значень. Для зведених значень можна вибрати лише одне поле.
2. У бланку запиту в рядку підсумків для поля зведених значень виберіть агрегатну функцію для обчислення значень.
3. У рядку "**Перехресний**" для поля зведених значень виберіть "**Значення**".
4. Вказувати критерії для поля зведених значень або виконувати сортування за цим полем не можна.
5. На вкладці "**Конструктор**" у групі "**Результати**" натисніть кнопку "**Запуск**".

Створення перехресного запиту в режимі SQL

Режим SQL не обмежує кількість таблиць або запитів, які можна використовувати як джерела записів для перехресного запиту. Проте структуру можна зробити простішою, створивши вибірковий запит, який повертає всі потрібні для перехресного запиту дані, а потім використавши цей вибірковий запит як джерело записів.

1. На вкладці "**Конструктор**" у групі "**Макроси та код**" натисніть кнопку "**Конструктор запиту**".
2. Закрийте діалогове вікно "**Відображення таблиці**".
3. На вкладці "**Конструктор**" у групі "**Результати**" натисніть кнопку SQL.
4. На вкладці "**Об'єкт SQL**" введіть або вставте такі SQL-оператори:

TRANSFORM
SELECT
FROM
GROUP BY
PIVOT

У першому рядку, після оператора TRANSFORM, введіть вираз, який буде використано для обчислення зведених значень, наприклад Сума([Обсяг]).

У другому рядку, після оператора SELECT, введіть список полів або виразів полів, які потрібно використати для заголовків стовпців. Розділяйте елементи списку за допомогою ком, наприклад [Бюджет].[Код_відділу], [Витрати].[Тип].

У третьому рядку, після оператора FROM, введіть список таблиць або запитів, які використовуються як джерела записів, наприклад Бюджет, Витрати.

У четвертому рядку, після оператора GROUP BY, введіть той самий список полів, який було використано в реченні SELECT у кроці 6.

У п'ятому рядку, після оператора PIVOT, введіть ім'я поля або вираз, який потрібно використовувати для заголовків стовпців, наприклад PIVOT [Бюджет].[Рік].

2.2 Завдання до лабораторної роботи

В лабораторній роботі необхідно створити п'ять таблиць запитів згідно з варіантом (табл. Б.2 Додаток Б).Матеріалом для створення запитів буде слугувати БД про робітників підприємства, що була створена в лабораторній роботі №1 (табл. А.2 Додаток А).

Перша таблиця запиту – запит на вибірку. Умова відбору для нього наведена в табл. Б.2 (Додаток Б) в стовпці "Запит типу 1".

Необхідно зробити вибірку по полю(будь-яке числове) яка б задовольняла умову > або < або = …

Наступна таблиця запиту – це таблиця, в якій подається відсортований зміст вихідної таблиці. Тобто необхідно взяти таблицю, яка вийшла у результаті першого запиту і відсортувати її.

В третьому запиті необхідно виконати обчислення. В графі “Запит типу 3” вказані номера пунктів з «відомостей до завдань для лабораторної роботи №2»(завдання наведені в наступному підпункті). Використовуючи ці номери необхідно створити запит, який буде виконувати обчислення згідно варіанту завдання, а потім виводити їх в таблицю.

В наступному завданні треба створити підсумковий запит (цей запит необхідно виконати для табл.А.2 Додаток А).

В останньому завданні необхідно створити запит на зміну. Для виконання цих завдань потрібно скористатись теоретичними відомостями до даної лабораторної роботи.

2.2.1 Відомості до завдань для лабораторної роботи № 2

1.Нарахування премії(табл. А.2 Додаток А)

Нарахування премії співробітникам підприємства виконується згідно коефіцієнта. Для визначення розміру премії необхідно виконати наступні операції:

а) визначити заробітну плату співробітника за місяць: якщо вид оплати в графі “Оплата” рівний 1, то зарплата визначається ставкою, що вказана в графі “Ставка”; якщо вид оплати рівний 2, то тарифна ставка, що вказана в графі “Ставка”, множиться на кількість годин (“Стаж”) і кількість робочих днів (“Відроблені”);

б) отриману зарплату помножити на коефіцієнт премії (“Премія”).

2.Нарахування за лікарняним листом(табл. А.2 Додаток А)

Для визначення розміру виплат за лікарняним листом необхідно виконати такі операції:

а) визначити денний заробіток працівника; якщо вид оплати (“Оплата”) рівний 1, то денний заробіток визначається діленням ставки, що вказана в графі “Ставка” на кількість робочих днів (22); якщо вид оплати рівний 2, то денний заробіток визначається множенням погодинної тарифної ставки (“Ставка”) на тривалість робочого дня (8 годин);

б) денний заробіток помножити на кількість лікарняних днів (графа “Лікарняні”);

в) отриманий результат помножити на коефіцієнт, що визначається стажем роботи (графа “Стаж”): якщо значення цієї графи менше 3, то коефіцієнт дорівнює 0.5; якщо значення графи більше або дорівнює 3, але менше 8, то коефіцієнт дорівнює 0.7; якщо значення графи більше або дорівнює 8, то коефіцієнт дорівнює 1.

3.Нарахування відпускних

Для нарахування відпускних потрібно виконати наступні операції:

а) визначити денний заробіток працівника (п. 2.а);

б) денний заробіток помножити на тривалість відпустки(24 дня).

4.Нарахування 13-ї зарплати(табл. А.2 Додаток А)

Для нарахування 13-ї зарплати необхідно виконати наступну послідовність дій:

а) визначити місячний заробіток працівника (п.1, а);

б) визначити розмір 13-ї зарплати шляхом множення місячного заробітку на коефіцієнт, що залежить від стажу роботи (графа “Стаж”): якщо значення графи “Стаж” менше 3, то коефіцієнт дорівнює 0.7; якщо значення більше або дорівнює 3, але менше 5, то коефіцієнт рівний 0.9; якщо значення графи більше або дорівнює 5, але менше 10, то коефіцієнт дорівнює 1.1; якщо стаж більше або дорівнює 10, то коефіцієнт дорівнює 1.5.

5.Нарахування вихідної допомоги(табл. А.2 Додаток А)

Для нарахування вихідної допомоги необхідно виконати такі операції:

а) визначити місячний заробіток працівника (п. 1, а)

б) визначити вихідну допомогу за формулою:

Вихідна допомога=місячний заробіток*”Допомога”, якщоМісячна зарплата < “Середня оплата”; Вихідна допомога=“Середня оплата”*”Допомога”, якщоМісячна зарплата > “Середня оплата”,де“Допомога” – коефіцієнт вихідної допомоги, “Середня оплата” – середня зарплата.

6.Нарахування авансу(табл. А.2 Додаток А)

Для нарахування авансу необхідно виконати такі дії:

а) визначити денний заробіток працівника (п. 2, а)

б) визначити аванс за формулою:

Аванс=”Денний заробіток” * (“Відроблені”+15)/2,де “Відроблені” – кількість днів, відроблених на 15-е число поточного місяця.

2.3 Контрольні запитання:

1. Що таке запит?
2. Назвіть призначення запитів. В чому відмінність запитів від таблиць БД.
3. Види запитів.
4. Назвіть види запитів на вибірку та їх особливості.
5. Коли використовуються і як створюються запити з параметрами.
6. Яке призначення та особливості створення перехресного запиту?
7. Способи створення запитів.
8. На основі яких об'єктів бази даних може формуватися запит?
9. Опишіть процес створення запитів за допомогою «Майстра запитів» програми MS Access.
10. Назвіть основні етапи при створенні запитів за зразком.
11. Що таке умова відбору? Як задається проста умова для відбору даних?
12. Як створити обчислюване поле в запиті? Порядок роботи з Будівником виразів MS Access.

ЛАБОРАТОРНА РОБОТА № 3
СТВОРЕННЯ ФОРМ

Мета роботи: навчитися створювати форми.

3.1 Теоретичні відомості

3.1.1 Необхідність форм

Форма – це об'єкт БД, який допомагає створювати інтерфейс користувача для застосуванняБД. «Зв'язана»– це форма, яка безпосередньо підключена до джерела даних, наприклад таблиці або запиту, і може використовуватися для введення, редагування або відображення даних із цього джерела даних. Ви також можете створити «вільну» форму, яка не зв'язується безпосередньо з джерелом даних, але містить кнопки, надписи або інші елементи керування, потрібні для роботи застосунку.

В Access можна створити форми нижченаведених*видів*:

- форма в стовпець або повноекранна форма;
- рядкова форма;
- таблична форма;
- форма головна/підлегла;
- зведена таблиця;
- форма-діаграма.

Дані в таблицю можна вносити і без допомоги форм, але існують принаймні чотири причини, що роблять форми незамінним засобом введення даних у базу.

По-перше, малокваліфікованому персоналу не можна давати доступ до таблиць.

По-друге, різні люди можуть мати різні права доступу до інформації, що зберігається в таблицях. *Наприклад*, один має право вводити тільки імена й адреси клієнтів, інший – тільки номера їхніх розрахункових рахунків, а третій – тільки грошові суми, що зберігаються на цих рахунках. Обмін інформацією між цими людьми потрібно унеможливити. Для введення даних їм дають різні форми, хоча дані з форм можуть надходити в одну таблицю.

По-третє, введення даних у таблицю – надзвичайно складне заняття. Вже після декількох годин роботи люди роблять помилки. Введення даних у форму простіше, аджебагато чого можна автоматизувати, до того ж елементи керування форм налаштовують таким чином, щоб при введенні даних виконувалася їх первинна перевірка.

І нарешті, по-четверте, треба згадати, звідки береться інформація для БД. Як правило, її беруть із паперових бланків (анкет, заяв, рахунків, і т. п.). Екранні форми можна зробити точною копією паперових бланків, з яких відбувається введення даних.

3.1.2 Створення форм

У програмі Access на вкладці **Створити** доступні кілька засобів швидкого створення форм, кожен із яких дає змогу створити форму одним клацанням миші. Проте, якщо ви бажаєте самостійно вибрати поля, які відображатимуться у формі, можна скористатися майстром форм. Майстер також дає змогу визначати спосіб

групування та сортування даних і використовувати поля з кількох таблиць або запитів (за умови, що зв'язки між таблицями та запитами вже визначено).

Створення форми за допомогою майстра форм

1. На вкладці “**Створити**” у групі “**Форми**” натисніть кнопку “**Майстер форм**”.

2. Дотримуйтеся вказівок на сторінках майстра форм.

*Примітка***.**Якщо потрібно додати до форми поля з кількох таблиць і запитів, не натискайте кнопку “**Далі**”або “**Готово**”після вибору полів із першої таблиці або запиту на першій сторінці майстра форм. Замість цього повторіть кроки для вибору таблиці або запиту, а потім клацніть будь-які додаткові поля, які потрібно додати до форми. Щоб продовжити, натисніть кнопку “**Далі**”або “**Готово**”.

3. На останній сторінці майстра натисніть кнопку “**Готово**”.

У майстрі форм можна отримати різноманітні результати, залежно від вибраних параметрів. Тому рекомендовано запускати майстер кілька разів, щоразу експериментуючи з різними параметрами, доки не отримаєте бажані результати.

Створення форми за допомогою засобу «Форма»

1. В області переходів виберіть таблицю або запит із даними, які слід відображати у формі.

2. На вкладці “**Створити**” у групі “**Форми**” натисніть кнопку “**Форма**”.

Access створює форму й відображає її в поданні розмічування. У цьому поданні можна змінювати структуру форми, а відображення в ній даних триватиме.

Створення розділеної форми за допомогою засобу «Розділена форма»

Розділена форма відрізняється від поєднання форми та підформи тим, що дані два подання зв’язані з одним джерелом даних і постійно синхронізуються одне з одним. Якщо вибрати поле в одній області форми, те ж поле буде обрано в іншій області форми. Ви можете додавати, редагувати або видаляти дані з будь-якої із двох областей (за умови, що джерело записів оновлюється та параметри форми не забороняють відповідні дії).

Робота з розділеними формами дає змогу використовувати переваги обох видів форм в одній. *Наприклад,* можна використати область табличного подання даних форми для швидкого пошуку запису, а потім використати область форми для перегляду й редагування цього запису.

1. В області переходів клацніть таблицю або запит із даними, які потрібно додати до форми, або відкрийте таблицю чи запит у вікні табличного подання даних.

2. На вкладці “**Створити**” у групі “**Форми**” натисніть кнопку “**Розділена форма**”.

Створення форми, у якій відображаються кілька записів, за допомогою засобу «Кілька елементів»

Якщо форму створено за допомогою засобу «Форма», то у створеній програмою Access формі відображається одночасно один запис. Якщо потрібно створити форму для одночасного відображення кількох записів, у якій доступно більше можливостей для настроювання, ніж у таблиці даних, можна використати засіб «Кілька елементів».

1. В області переходів виберіть таблицю або запит із даними, які слід відображати у формі.

2. На вкладці “**Створити**” у групі “**Форми**” натисніть кнопку “**Додаткові форми**” та виберіть пункт “**Кілька елементів**”.

Створена у програмі Access за допомогою засобу «Кілька елементів» форма нагадує дані в табличному поданні – дані впорядковано в рядках і стовпцях, одночасно можна переглядати кілька записів. Але у формі з кількома елементами доступно більше можливостей настроювання, ніж для даних у табличному поданні, *наприклад,* можна додати графічні елементи, кнопки та інші елементи керування.

Створення форми за допомогою засобу «Порожня форма»

Якщо потрібну форму не вдалося створити за допомогою майстра або засобу побудови форм, можна використати для побудови форми засіб «Пуста форма».

1. На вкладці “**Створити**” у групі “**Форми**” натисніть кнопку “**Порожня форма**”.

Пуста форма відкривається в поданні розмічування програми Access, також відображається область “**Список полів**”.

2. В області “**Список полів**” клацніть знак «плюс» (+) поруч із таблицею (таблицями) з полями, які слід відображати у формі.

3. Щоб додати поле до форми, клацніть його двічі або перетягніть до форми.

Примітки.

Після того, як перше поле додано, можна додати кілька полів одночасно. Для цього виберіть кілька полів, утримуючи натиснутою клавішу CTRL, і перетягніть їх до форми.

Порядок таблиць в області списку полів може змінюватися залежно від вибраної області форми. Якщо поле, яке потрібно додати до форми, не відображається, спробуйте вибрати іншу область форми та повторіть спробу додавання поля.

Засоби у групі “**Колонтитули**” на вкладці “**Проектування**” використовуються для додавання емблеми, назви, номерів сторінок або дати й часу до форми.

Використовуйте засоби у групі “**Елементи керування**” на вкладці “**Конструктор**”, щоб додати інші елементи керування до форми.

Якщо потрібно додати інші різноманітні елементи керування, перейдіть до режиму конструктора, клацнувши правою кнопкою миші форму та вибравши пункт “**Конструктор**”.

3.1.3 Структура форм

Створюючи форми автоматичними засобами, можна не замислюватися над їх структурою, але при розробці форми вручну зі структурою доводиться мати справу.

Структуру форми складають її розділи, а розділи містять елементи керування.

3.1.4 Елементи керування та налаштування форми

При створенні форми вручну елементи керування розміщують на ній так, як зручно проектувальнику.Крім елементів керування “Напис” і “Поле”, існує ще декілька корисних елементів керування.

Перемикачі. З ними можна зв'язати команди, *наприклад*, фільтрацію.

Прапорці. Діють аналогічно перемикачам, але допускають множинний вибір. Зручні для керування режимами сортування даних.

Список. Може містити фіксований набір значень або з заданого поля однієї з таблиць. Дозволяє не вводити дані, а вибирати із списку.

Поле зі списком. Застосовується, як і список, але займає менше місця у формі, оскільки список відкривається тільки після натиснення на кнопці.

Командні кнопки. З кожній із них можна зв'язати корисну команду, наприклад команду пошуку запису, переходу між записами й інші.

Вкладки. Дозволяють розмістити багато інформації на обмеженій площі. На вкладках розміщають інші елементи керування.

Поле об'єкта OLE. Служить для розміщення зовнішнього об'єкта, що відповідає прийнятої в Windows концепції зв'язування і впровадження об'єктів. Об'єктом, як правило, є ілюстрація, *наприклад,* фотографія, але це може бути і відеозапис, і музичний фрагмент, і голосове повідомлення.

Існують два типи полів для розміщення об'єктів OLE:

"**Вільна рамка об'єкта**" і "**Приєднана рамка об'єкту**". У першому випадку рамка не зв'язана ні з яким полем таблиць бази даних. Об'єкт, що знаходиться в ній, виконує роль ілюстрації і служить для оформлення форми. З "**Приєднана рамка об'єкту**" пов'язане одне з полів таблиці. У ній відображається вміст цього поля. Цей вміст може змінюватися при переході від одного запису до іншого.

Налаштування форми в поданні розмічування

Після створення форми можна легко налаштувати її макет за допомогою подання розмічування. Використовуючи дані дійсної форми як орієнтир, можна перевпорядкувати елементи керування й налаштовувати розміри.

1. Для переходу до подання розмічування клацніть правою кнопкою миші форму в області переходів і виберіть пункт "**Режим розмічування**".

Форма відображається в поданні розмічування програми Access.

2. Щоб змінити властивості форми, її елементи керування та розділи, можна скористатися вікном властивостей (натисніть клавішу F4).

3. Щоб додати поля з основної таблиці або запиту до макета звіту, можна використати область "**Список полів**"(на вкладці "**Конструктор**" у групі "**Знаряддя**" натисніть кнопку "**Додати наявні поля**"; або сполучення клавіш ALT+F8).

4. Потім можна перетягнути поля безпосередньо з області "**Список полів**" до форми.

- Для додавання одного поля клацніть його двічі або перетягніть з області "**Список полів**" до розділу форми, в якому слід його відображати.
- Для одночасного додавання кількох полів виберіть потрібні поля, утримуючи натиснутою клавішу CTRL. Потім перетягніть вибрані поля до форми.

Налаштування форми в режимі конструктора

Щоб перейти до режиму конструктора, в області переходів клацніть правою кнопкою миші ім'я форми та виберіть пункт "**Конструктор**".

Примітка.Режим конструктора недоступний під час роботи з веб-базою даних.

Програма Access відображає форму в режимі конструктора.

Далі користуватись пунктами 2–4 аналогічними у попередньому описі.

Створення написів

Редагування форм перебуває в створенні нових або зміні наявних елементів керування, а також у зміні їхнього взаємного розташування.

При розгляді прийомів створення нових елементів керування ми скористаємося тим фактом, що **Майстер**, що створив форму, не заповнив її розділ заголовка.

1. Перетягнувши вниз розділювальну межу між заголовком і областю даних, ми можемо звільнити вгорі досить місця для створення великого напису.

2. На панелі елементів існує спеціальний елемент керування для створення заголовків, що називається "**Напис**".

3. Натиснувши на ньому, а потім на формі, ми одержуємо текстову рамку, у якій можна вводити довільний текст. При введенні тексту не треба турбуватися про його форматування. Завершивши введення, треба натиснути клавішу ENTER, після чого можна оформлювати текст.

4. Для форматування елемента керування його треба спочатку виділити. Для цього служить інструмент "**Вибір об'єктів**".

При виділенні елемента керування навколо нього утворюється рамка з вісьмома маркерами (по кутках і по центрах сторін рамки). Рамку можна розтягувати або стискувати методом перетягування меж. При наведенні на маркер покажчик миші змінює форму, приймаючи зображення відкритої долоні. У цей момент рамку можна переміщувати.

5. Особливу роль грає лівий верхній маркер рамки. При наведенні на нього покажчик миші приймає форму вказівного пальця.

Коли об'єкт виділений, можна змінювати параметри шрифту, метод вирівнювання тексту й інші елементи форматування. Це виконують звичайними засобами форматування, доступними через відповідну панель інструментів Access.

Якщо натиснути на виділеному елементі правою кнопкою миші, відкриється його контекстне меню, у якому є додаткові можливості зміни оформлення.

3.2 Завдання до лабораторної роботи

1. Створити форму в режимі конструктора для заданого варіанта(лабораторна робота № 1), яка буде мати вигляд як на рис. В.1 (додаток В).
2. Форма повинна мати три розділи:

- Заголовок (повинен містити напис з назвою варіанту завдання, відповідно до варіанту лабораторної роботи № 2, *наприклад,* “Розрахунок премії”);
- область даних, що містить зв’язані поля таблиці “Кадри”;
- примітки (прізвище студента).

3. В області даних використати дві вкладки:

в першій вкладці – перша група полів таблиці “Основна інформація”;
в другій вкладці – друга група полів таблиці “Додаткова інформація”;

4. Створити форму в режимі “Майстер форм”.

3.3. Контрольні питання:

1. Що таке форма? Які дії вона дозволяє виконувати?
2. Які види форм можливо створити у Microsoft Access?
3. Чим відрізняється автоформа у стовпець, стрічкова, таблична? Для чого вони призначені.
4. Якими способами можна створювати форму у Microsoft Access?
5. Як створити на формі поле, що обчислюється?
6. Як викликати на екран вікно побудови виразів?
7. Які елементи керування можна вставити на форму?

ЛАБОРАТОРНА РОБОТА № 4
СТВОРЕННЯ ЗВІТІВ ЗАСОБАМИ MSACCESS

Мета: навчитись створювати звіти різних типів, використовуючи майстер та конструктор.

4.1 Теоретичні відомості

4.1.1 Визначення звіту та його складові

Звіт – це об'єкт БД, який використовується для відображення та підсумування даних. Звіти використовуються для розповсюдження або архівації знімків даних через друк, перетворення на файли формату PDF або XPS чи експортування в інші формати файлів.

Звіт може містити докладну інформацію про певний запис, зведені дані з багатьох записів або те та інше разом.

Звіт програми Access складається з кількох частин. Щоб побачити ці частини у клієнтській БД, звіт потрібно відкрити в режимі конструктора. У наведеному нижче списку описуються типи частин і їх призначення.

- **Верхній колонтитул звіту**(друкується лише один раз, на початку звіту). У ній використовуються дані, які зазвичай розміщуються на титульній сторінці, наприклад емблема, назва або дата. Якщо у верхньому колонтитулі звіту розмістити обчислюваний елемент керування, у якому використовується агрегатна функція Sum, сума обчислюватиметься для всього звіту. Спочатку друкується верхній колонтитул звіту, а потім – верхній колонтитул сторінки.
- **Верхній колонтитул сторінки**(друкується у верхній частині кожної сторінки). *Наприклад*, він використовується для повторення заголовка звіту на кожній сторінці.
- **Верхній колонтитул групи**(друкується на початку кожної нової групи записів). Він використовується для друку назви групи. *Наприклад*, у звіті, згрупованому за товарами, верхній колонтитул групи використовується для відображення назви товару. Якщо у верхньому колонтитулі групи розмістити обчислюваний елемент керування, у якому використовується агрегатна функція Sum, сума обчислюватиметься для поточної групи. Залежно від кількості рівнів групування звіт може містити кілька частин із верхнім колонтитулом групи.
- **Подробиці**(друкується один раз для кожного рядка у джерелі записів). Саме тут розміщуються елементи керування, які становлять тіло звіту.
- **Нижній колонтитул групи**(друкується в кінці кожної групи записів). Нижній колонтитул групи використовується для відображення зведених даних для групи. Залежно від кількості рівнів групування звіт може містити кілька частин із нижнім колонтитулом групи.

• **Нижній колонтитул сторінки**(друкується наприкінці кожної сторінки). Вона використовується для друку номерів сторінок або даних, які стосуються кожної сторінки.

• **Нижній колонтитул звіту**(друкується лише один раз наприкінці звіту). Він використовується для друку підсумків або інших зведених даних для всього звіту.

*Типи звітів.*Звіти, створені за допомогою цих засобів (табл. 4.1), не можна публікувати у службах Access Services. Якщо додати такий звіт до веб-БД, БД всеодно можна буде опублікувати, однак звіти клієнтської програми не відображатимуться у браузері. Дані звіти можна використовувати лише у програмі Access.

Таблиця 4.1 – Опис засобів для створення звітів

Зображення кнопки	Засіб	Опис
	Звіт	Створює простий табличний звіт, який містить усі поля вибраного в області переходів джерела записів.
	Конструктор звітів	Відкриває пустий звіт у режимі конструктора. До цього звіту можна додавати лише потрібні поля й елементи керування.
	Пустий звіт	Відкриває пустий звіт у режимі розмічування та відкриває область завдань "Список полів". Під час перетягування полів зі списку завдань до звіту створюється вбудований запит, який зберігається у властивості "Джерело записів" звіту.
	Майстер звітів	Запускає покроковий майстер, у якому можна вказати поля, рівні групування/сортування та параметри макета. На основі вказаних у майстрі параметрів створюється звіт.
	Етикетки	Запускає майстер, у якому можна вибирати стандартний або настроюваний розмір етикеток, а також указувати, які поля потрібно відображати та як їх слід сортувати. На основі вказаних у майстрі параметрів створюється етикетка.

4.1.2 Порядок створення звітів

Крок 1. Вибір джерела записів

У ролі джерела записів може виступати таблиця або іменований чи вбудований запит. Джерело записів має містити всі рядки та стовпці, дані з яких необхідно додати до звіту.

Крок 2. Вибір засобу для створення звітів

Засоби для створення звітів розміщені на вкладці “**Створити**” у групі “**Звіти**”**.**

Крок 3. Створення звіту

1. Натисніть кнопку потрібного звіту. Якщо запуститься майстер, виконайте його вказівки та на останній сторінці натисніть кнопку “**Готово**”**.** Відобразиться звіт у режимі розмічування.

2. Відформатуйте звіт так, як вам потрібно:

- Змініть розмір полів і етикеток, перетягуючи їхні краї.
- Перетягніть поле (і, за наявності, його підпис) в інше місце.
- Клацніть правою кнопкою миші поле та виберіть у контекстному меню відповідну команду, щоб розділити або об’єднати клітинки, видалити або вибрати поля чи виконати інше форматування.

4.1.3 Групування, сортування та підсумовування

Найшвидший спосіб додати до звіту групування, сортування або підсумовування – це клацнути правою кнопкою миші потрібне поле та вибрати в контекстному меню відповідну команду.

Групування, сортування та підсумовування також можна додавати за допомогою панелі "Групування, сортування й підсумок", коли звіт відкрито в режимі розмічування або конструктора:

1. Щоб відкрити цю панель, перейдіть на вкладку "**Конструктор**" і у групі "**Групування та підсумки**" натисніть кнопку "**Групування та сортування**".
2. Натисніть кнопку "**Додати групу**" або "**Додати сортування**", а потім укажіть поле, за яким потрібно виконати групування або сортування.
3. Щоб указати додаткові параметри та додати підсумки, у рядку групування або сортування натисніть кнопку "**Розгорнути**".

4.1.4 Створення професійного вигляду за допомогою тем

До БД Access тепер можна застосовувати теми Office 2010. Це дає змогу використовувати той самий стиль у всіх документах Office.

Застосування теми, шрифту або кольору Office впливає на всі форми та звіти в БД (а не лише на ті, з якими ви зараз працюєте).

1. Відкрийте звіт у режимі розмічування, клацнувши звіт правою кнопкою миші в області переходів і вибравши пункт "**Режим розмічування**".
2. На вкладці "**Конструктор**" у "**Теми**" виберіть тему, шрифт, колір (рис. 4.1):

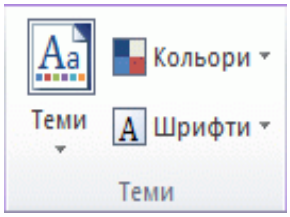

Рисунок 4.1 – Вибір теми, шрифту або кольору

- У колекції "**Теми**" виберіть один із наборів попередньо створених кольорів і шрифтів.
- У колекціях "**Кольори** та "**Шрифти**" кольори та шрифти можна вибирати окремо.

4.1.5 Додавання зображень

У програмі Access зображення завжди пов'язувалися з окремими елементами керування зображеннями у формах або звітах. Щоб змінити зображення, яке зазвичай використовувалося в кількох формах або звітах, потрібно було вручну змінювати кожен елемент керування. У програмі Access 2010 рисунок додається до БД один раз, а потім його можна використовувати в кількох об'єктах. Оновлення одного рисунка призведе до його оновлення в усіх об'єктах БД, у яких він використовується. Ця функція надзвичайно корисна, коли потрібно змінити щось на кшталт емблеми компанії або зображень тла, які використовуються в усій БД.

Додавання зображення:

1. В області переходів клацніть правою кнопкою миші звіт, до якого потрібно додати зображення, і виберіть пункт "**Режим розмічування**".
2. Клацніть місце у звіті, куди слід вставити зображення.

3. На вкладці “**Конструктор”** у групі **“Елементикерування”** натисніть кнопку **“Вставити зображення”**.

4. Виконайте одну з наведених нижче дій.

- “**Використайте наявне зображення**”**.** Якщо в колекції є потрібне зображення, додайте його до звіту.

- “**Завантажте нове зображення**”**.** У нижній частині колекції виберіть пункт “**Огляд**”. У діалоговому вікні “**Вставлення зображення**” вкажіть потрібний рисунок і натисніть кнопку “**Відкрити**”. Виділений рисунок буде доданий до звіту.

4.2 Завдання до лабораторної роботи

1. Створити звіт у звичайному режимі «Звіт».
2. Створити звіт в режимі Конструктора.
3. Додати у звіт графік та текст.
4. Виконати підсумкові дії у звіті (підсумовування, підрахунок кількості елементів, сортування тощо).
5. Додати поточну дату і час у звіт.
6. Створити елементи керування у звіті.
7. Створити звіт за допомогою Майстра звітів.

4.3 Контрольні запитання:

1. Що таке звіт?
2. Перелічити основні елементи звіту.
3. Що може бути джерелом записів для звіту?
4. Як вибрати джерело записів для звіту?
5. Як додати до звіту групування?
6. Як додати до звіту підсумовування?
7. Як додати до звіту сортування?
8. Які є способи створення звітів?
9. Які є стилі створення звітів?
10. Які макети використовуються при створенні звітів?
11. Назвати етапи створення звіту за допомогою Майстра звітів.

ЛАБОРАТОРНА РОБОТА №5
ПРОЕКТУВАННЯ РЕЛЯЦІЙНОЇ БАЗИ ДАНИХ

Мета: опис предметної сфери, створення концептуальної та логічної моделі БД.

1.1 Теоретичні відомості

5.1.1 Етапи проектування реляційної БД

Для проектування реляційної БД потрібно:

Визначити об’єкти, які містяться в БД.

Визначити зв’язки між об’єктами.

Визначити основні властивості об’єктів.

Визначити зв’язки між властивостями об’єктів.

Створити робочий словник даних для визначення таблиць, що входять до БД.

Визначити відношення між таблицями баз даних, базуючись на зв’язках між об’єктами даних, що містяться в таблиці, і включити цю інформацію до словника даних.

Продумати операції, що виконуються при створенні та зміні інформації таблиць, включаючи забезпечення цілісності даних.

Визначити, як використовувати індекси для прискорення виконання запитів, щоб уникнути сильного уповільнення роботи при додаванні даних до таблиці і надмірного збільшення об’єму дискового простору, що займається базою.

Визначити користувачів, яким дозволений доступ до даних, їх редагування, а також зміна при необхідності структури таблиць.

Описати структуру БД в цілому, завершити створення словників даних для своєї бази та для кожної таблиці, що міститься в ній, розробити процедури для операцій з БД, включаючи створення резервних копій і відновлення вихідних файлів.

5.1.2 Типи таблиць і ключів в реляційних БД

В реляційній базі даних існують базові та проміжні таблиці.

Базова таблиця–таблиця, яка включає один або декілька стовпців властивостей об’єкту і містить первинний ключ, що однозначно визначає цей об’єкт. Базова таблиця повинна містити первинний ключ. Базові таблиці часто називають первинними, оскільки мають первинний ключ.

Проміжна таблиця. Таблиця, що не є базовою, яка використовується для забезпечення зв’язків між іншими таблицями, називається таблицею відношень. Ключові поля в таблиці відношень повинні бути зовнішніми ключами, що зв’язані з первинними ключами базової таблиці.

Розглянемо типи ключів в реляційній БД:

Первинний ключ складається з набору значень, які однозначно визначають рядок (запис) базової таблиці. Будь-якому значенню первинного ключа повинна відповідати один і тільки один рядок (запис) таблиці. Первинний ключ включає одне поле тільки в тому випадку, якщо це поле не містить значень, що повторюються.

Ключі-кандидати. Будь-який стовпець або група стовпців, які задовольняють вимогам, що накладаються на значення первинного ключа, є кандидатами на те, щоб стати первинним ключем.

Складені ключі. Якщо для виконання умов, що накладаються на значення первинного ключа, заданий ключ включає декілька полів таблиці, то тоді він називається складеним.

Зовнішні ключі– це стовпець, значення якого відповідають значенням первинного ключа з іншої зв'язаної таблиці.

5.1.3 Схематичні моделі даних

Існує багато методів створення схем моделей даних. Одним з найбільш розповсюджених є метод, в якому використовується схема "Елемент - Відношення" (E-R) (рис. 5.1), яка була розроблена Пітером Ченом в 1976 році. **E-R схеми** призначені для наглядного представлення відношень між об'єктами і поведінки елементів.

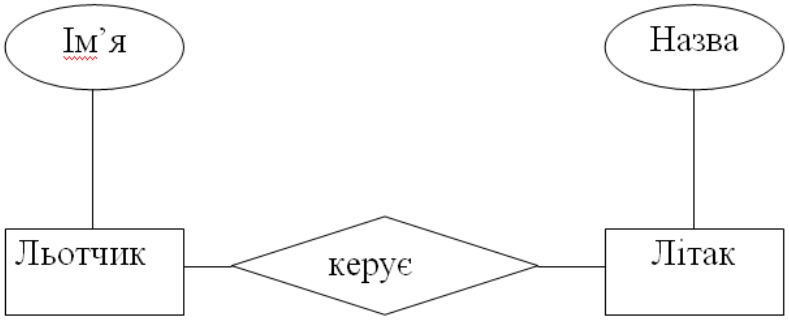

Рисунок 5.1 – Схема "Елемент - Відношення"

Елементи даних вказані у прямокутниках, атрибути даних – в овалах, а відношення між елементами – в ромбах. Відношення між об'єктами БД на концептуальному етапі можуть визначатися їх поведінкою. Таким чином E-R схеми включають принаймні одне дієслово, об'єкт якого знаходиться справа від символу відношення. Символи наносяться на схему по мірі конкретизації моделі. Однією з переваг E-R схем є те, що їх можна використовувати для представлення на порівняно невеликому просторі концептуальної моделі великих схем з багатьма БД.

Графічний опис структури таблиць у формі полосок, які містять імена полів і показують спрощені відношення між даними, використовується для того, щоб користувачам було легше зрозуміти розроблену модель даних. Діаграма, на якій показано логічне представлення даних, називається **схемою**.

5.1.4 Нормалізація даних в реляційній моделі

Нормалізацією називається формальна процедура, в ході якої атрибути даних групуються в таблиці, а таблиці групуються в БД.

Задачами нормалізації є:

- ✓ вилучення з таблиць інформації, що повторюється;
- ✓ створення структури, в якій передбачена можливість майбутніх змін;
- ✓ створення структури, в якій вплив структурних змін на додатки, що використовують дані цієї БД, зведено до мінімуму.

Для **першої нормальної форми** потрібно, щоб таблиця була двовимірною і не містила груп, що повторюються. У таких таблиць є тільки дві характеристики – довжина (кількість записів або рядків) та ширина (кількість полів або стовпців). Вона не повинна містити комірок, що включають кілька значень. Для того, щоб в одній комірці містилося кілька величин, необхідно ввести третій вимір – глибину, за допомогою якої можна зберігати в одній комірці одразу декілька значень.

Для **другої нормальної форми** потрібно, щоб дані у всіх не ключових стовпцях повністю залежали від первинного ключа і кожного елемента (стовпця) первинного ключа, якщо ключ є складеним. Під повною залежністю розуміються те, що значення в кожному не ключовому стовпці однозначно визначається значенням первинного ключа. Якщо одне з полів не залежить від величини первинного ключа, то необхідно включити в ключ доповнювальні таблиці. Перед перевіркою на відповідність другій нормальній формі таблиця повинна бути приведена до першої нормальної форми. Друга нормальна форма дозволяє видалити більшу частину даних, що повторюються, які часто залишаються після першого етапу.

Для **третьої нормальної форми** потрібно, щоб всі неключові стовпці таблиці не тільки залежали від первинного ключа таблиці, але були незалежними один від одного, тобто, щоб були відсутні транзитивні функціональні залежності між стовпцями. Для цього потрібно, щоб таблиці були попередньо приведені до першої, другої нормальної форми.

Типи відношень

Найпростішим відношенням між таблицями є **відношення “один-до-одного”**. В такому відношенні одному запису однієї таблиці відповідає тільки один запис у іншій. Таблиці, що зв’язані відношенням “один-до-одного” можна об’єднати в одну таблицю, яка складається з полів обох таблиць. *Наприклад*, це може бути потрібним для того, щоб скоротити час перегляду полів, що містять певний набір даних. В деяких випадках необхідно керувати доступом до частин таблиць, які містять важливі або конфіденційні дані. На рис. 5.2 показана E-R схема для таблиці “Інженер” та “Комп’ютер”. Одиниці з обох сторін ромба вказують на відношення “один-до одного”.

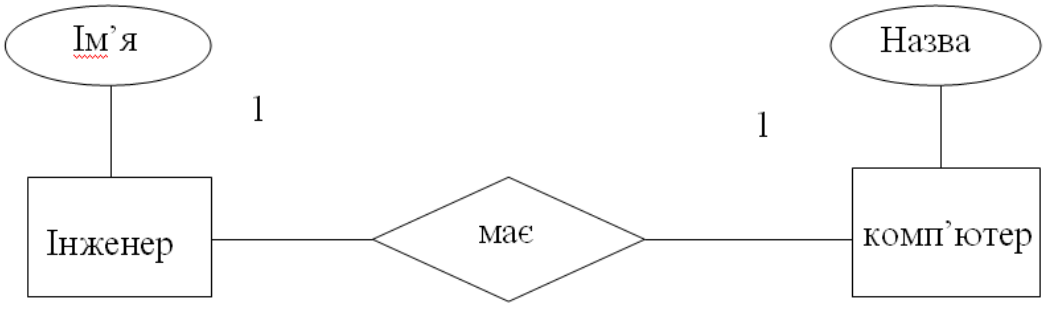

Рисунок 5.2 – E-R схема для таблиці “Інженер” та “Комп’ютер”

Таблиця з обов’язковим відношенням “один-до-одного” є базовою. Таблиця, зв’язана необов’язковим відношенням з базовою є зв’язаною таблицею. Для зберігання бази даних, що містить декілька таблиць, що зв’язані відношеннями “один-до-одного”, деякі з яких є необов’язковими, потрібно менше місця на диску.

Відношення “один-до-багатьох” зв’язує один запис першої таблиці з декількома записами другої за допомогою первинного ключа базової таблиці і відповідного йому зовнішнього ключа зв’язаної таблиці. Зовнішній ключ таблиці, що містить велику кількість відношень, може входити до складеного первинного ключа, але він є зовнішнім по відношенню до базової таблиці. Відношення “один-до-багатьох” використовується найбільш часто. На E-R схемі, що показана на рис.5.3 , це відношення позначено символом m.

Рисунок 5.3 – E-R схема для таблиці “Паром” та “Порти”

Відношення “багато-до-одного” протилежне відношенню “один-до-багатьох”. Якщо вибір відношення “багато-до-одного” або “один-до-багатьох” не має великої ролі, то відношення між таблицями називається рефлексивним. Відношення “багато-до-одного” є відображенням відношення “один-до-багатьох”. Всі відношення “багато-до-одного” в Access є рефлексивними. Рефлексивні відношення позначаються дієсловом у відповідній формі, яке розміщується зовні ромба, що визначає відношення.

5.1.5 Звязування таблиць між собою в Access

Зв’язок між таблицями можна створити за допомогою вікна “**Зв’язки**” або перетягуванням поля з області **Список полів**. Під час створення зв’язку між таблицями спільні поля можуть мати різні імена, хоча часто вони однакові. Проте такі поля мають містити дані одного типу. Якщо поле первинного ключа має тип “Лічильник”, зовнішній ключ може бути полем типу “Число”, якщо обидва поля мають однакові значення властивості **Розмір поля (FieldSize)**. *Наприклад,* поля типів “Лічильник” і “Число” можна зіставляти, якщо властивість **Розмір поля (FieldSize)** обох полів має значення Long Integer (довге ціле). Якщо обидва спільні поля мають тип “Число”, вони повинні мати однакове значення властивості **Розмір поля (FieldSize)**.

Створення зв’язку між таблицями за допомогою вікна “Зв’язки”.

1. У вкладці **Файл** виберіть команду **Відкрити**.
2. У діалоговому вікні **Відкрити** виберіть і відкрийте БД.
3. У вкладці **Знаряддя бази даних** у групі **Зв’язки** натисніть кнопку **Зв’язки**.
4. Якщо зв’язки ще не визначено, автоматично відображається діалогове вікно **Відображення таблиці**.
5. Якщо воно не відображається, у вкладці **Конструктор** у групі **Зв’язок** натисніть кнопку **Відобразити таблицю**.
6. У діалоговому вікні **Відображення таблиці** відображаються всі таблиці й запити в базі даних. Щоб переглянути лише таблиці, виберіть вкладку **Таблиці**. Щоб переглянути лише запити, виберіть вкладку **Запити**. Для перегляду як таблиць, так і запитів виберіть вкладку **Разом**.
7. Виберіть одну або кілька таблиць чи запитів і натисніть кнопку **Додати**. Закінчивши додавати таблиці й запити до вікна “Зв’язки”, натисніть кнопку **Закрити**.

8. Перетягніть поле з однієї таблиці (зазвичай це первинний ключ) до спільного поля (зовнішнього ключа) в іншій таблиці. Щоб перетягнути кілька полів, натисніть клавішу CTRL, клацніть потрібні поля, а потім виконайте перетягування. Відобразиться діалогове вікно **Редагування зв'язків**.

9. Переконайтеся, що відображувані імена полів відповідають спільним полям зв'язку. Якщо ім'я поля неправильне, клацніть його й виберіть зі списку нове поле.

10. Щоб забезпечити для цього зв'язку обмеження цілісності, установіть прапорець **Забезпечення цілісності даних**.

11. Натисніть кнопку **Створити**. Між двома таблицями буде зображено лінію зв'язку. Якщо встановлено прапорець **Забезпечення цілісності даних**, лінія на кінцях виглядає товстішою. Крім того, знову ж таки, якщо встановити прапорець **Забезпечення цілісності даних**, над товстішим відрізком лінії з одного боку відображатиметься число **1**, а над товстішим відрізком лінії з іншого боку – знак нескінченності (∞).

Примітка:

- **Створення зв'язку "один-до-одного"**. Обидва спільні поля (зазвичай, поля первинного та зовнішнього ключів) повинні мати унікальний індекс. Це означає, що для властивостей **Індексовано** цих полів потрібно встановити значення **Так (без повторень)**. Якщо обидва поля мають унікальні індекси, у програмі Access створюється зв'язок "один-до-одного".
- **Створення зв'язку "один-до-багатьох"**. Поле на стороні зв'язку "один" (зазвичай, первинний ключ) повинно мати унікальний індекс. Це означає, що для властивості **Індексовано** цього поля потрібно встановити значення **Так (без повторень)**. Поле на стороні "багато" не повинно мати унікальний індекс. Воно може мати індекс, але він має підтримувати повторення. Це означає, що властивість **Індексовано** повинна мати значення або **Ні (No)**, або **Так (повторення дозволені)**. Коли одне поле має унікальний індекс, а друге поле такого індексу не має, у програмі Access створюється зв'язок "один-до-багатьох".
- **Створення зв'язку таблиці за допомогою області "Список полів".**Ви можете додати поле до наявної таблиці, відкритої в поданні таблиці, перетягнувши його з області Список полів. В Список полів відображаються поля, доступні у зв'язаних таблицях, а також поля, доступні в інших таблицях. Якщо перетягнути поле з "іншої" (незв'язаної) таблиці, а потім запустити майстер підстановок і виконати його вказівки, між таблицею в області Список полів і таблицею, до якої перетягується поле, буде автоматично встановлено зв'язок "один-до-багатьох". До цього, створеного програмою Access, зв'язку посилальні обмеження цілісності за промовчанням не застосовуються. Для застосування посилальних обмежень цілісності зв'язок слід відредагувати.

Відкриття таблиці у вікні табличного подання даних

1. На вкладці **Файл** виберіть команду **Відкрити**.
2. У діалоговому вікні **Відкрити** виберіть і відкрийте БД.
3. В області переходів клацніть правою кнопкою миші таблицю, до якої потрібно додати поле та з якою потрібно створити зв'язок, а потім натисніть кнопку **Відкрити**.

Відкриття області "Список полів"

Натисніть сполучення клавіш ALT+F8. Відобразиться область **Список полів**. В області **Список полів** відображаються всі інші таблиці бази даних, згруповані за категоріями. Під час роботи з таблицею у вікні табличного подання даних у програмі Access відображаються поля в одній із двох категорій області **Список полів**: **Поля, наявні в пов'язаних таблицях** і **Поля, наявні в інших таблицях**. У першій категорії перелічуються всі таблиці, які мають зв'язок із таблицею, що зараз використовується, а у другій категорії – усі таблиці, з якими поточна таблиця не має зв'язків.

Якщо в області **Список полів** клацнути знак плюс (+), розташований поруч з іменем таблиці, буде відображено список усіх полів, доступних у цій таблиці. Щоб додати поле до таблиці, перетягніть його з області **Список полів** до таблиці у вікні табличного подання даних.

Додавання поля та створення зв'язку з області "Список полів"

1. В області **Список полів** у розділі **Поля, наявні в інших таблицях** клацніть знак плюс (+), розташований поруч з іменем таблиці, щоб відобразити список полів.
2. Перетягніть потрібне поле з області **Список полів** до таблиці, відкритої у вікні табличного подання даних.
3. Після появи лінії вставлення вставте поле. Запуститься **майстер підстановок**.
4. Дотримуйтесь інструкцій **майстра підстановок**.

Відобразиться таблиця у вікні табличного подання даних.

Якщо перетягнути поле з "іншої" (незв'язаної) таблиці, а потім запустити майстер підстановок і виконати вказівки, між таблицею в області **Список полів** і таблицею, до якої перетягується поле, буде автоматичного встановлено зв'язок "один-до-багатьох". Для застосування посилальних обмежень цілісності зв'язок слід відредагувати.

5.2 Завдання до лабораторної роботи

1. Розробити ER-модель і первинні таблиці БД.
2. Провести нормалізацію первинних таблиць.
3. Ввести нормалізовані таблиці в СУБД.
4. Призначити первинні ключі всім таблицям:

а) Увійти у режим конструктора;

б) Встановити курсор навпроти потрібного ключового поля і натиснути кнопку «**Ключове поле**»;

в) При необхідності призначення ключем декількох полів натиснути клавішу Ctrl і, не відпускаючи її, виділити кольором потрібні поля і натиснути кнопку «**Ключове поле**».

5. Створити зв'язки між ключовими полями всіх таблиць. Задати властивості нового зв'язку:"забезпечення цілісності"; "каскадне оновлення полів".
6. Створити запит, який містить в собі поля з різних таблиць.

5.3 Контрольні запитання:

1. Назвіть основні етапи проектування реляційної БД.
2. З якою метою виконується нормалізація?
3. Що називають нормалізацією відношень?
4. З яких етапів складається життєвий цикл БД?
5. Що відбувається на етапі концептуального проектування БД?
6. Що відбувається на етапі логічного проектування БД?
7. Які ви знаєте рівні проектування інформаційних систем?
8. Які особливості проектування реляційних БД?
9. Перша нормальна форма та методи її досягнення.
10. Друга нормальна форма та її характеристики.
11. Третя нормальна форма та її характеристики.
12. Зв'язки між сутностями та їх утворення.

ЛАБОРАТОРНА РОБОТА №6
СТВОРЕННЯ БАЗ ДАНИХ В MICROSOFT SQL SERVER

Мета: за допомогою операторів мови Transact SQL навчитися створювати БД, спроектовувати її базові таблиці і визначати відношення між ними.

6.1 Пояснення до виконання роботи

6.1.1 Мова SQL

Мова SQL є найбільш поширеною мовою для роботи з БД. На даний час існують такі міжнародні стандарти на мову SQL: SQL1, SQL2, SQL3.

Мова SQL не володіє функціями повноцінної мови розробки і орієнтована на доступ до БД. Використання мови SQL може бути самостійним і вона може включатися в склад засобів розробки програм. В цьому випадку її називають вбудованою SQL. Розрізняють два головних методи використання вбудованої SQL: статичний і динамічний.

Статичне використання передбачає застосування в програмі функцій викликів мови SQL, які включаються в програмний модуль і виконуються після компіляції програми.

Динамічне використання передбачає динамічну побудову викликів функцій мови SQL та інтерпретацію цих викликів у ході виконання програми. Динамічний метод застосовується тоді, коли вид SQL запиту заздалегідь невідомий і будується у діалозі з користувачем.

Будь-яке SQL-застосування реляційної БД складається з трьох частин: інтерфейса користувача, набору таблиць в БД і SQL-машини.

Microsoft SQL сервер підтримує доступ до БД в багатокористувацькому режимі за допомогою локальної мережі, та мережі Інтернет. Доступ до бази реалізованої на Microsoft SQL сервері, може здійснюватись за допомогою Web додатків, мов програмування Java, та C#. А також підтримується робота з мовою розмітки XML.

Microsoft SQL Server – це реляційна СУБД. У реляційних БД дані зберігаються в таблицях. Взаємопов'язані дані можуть групуватися в таблиці, крім того, можуть бути встановлені також і відношення між таблицями. Користувачі отримують доступ до даних на сервері через програми, а адміністратори, виконуючи завдання конфігурування, адміністрування та підтримки БД, виробляють безпосередній доступ до сервера. SQL Server є масштабованою БД, це означає, що вона може зберігати значні обсяги даних і підтримувати роботу багатьох користувачів, які здійснюють одночасний доступ до БД.

Під час створення БД необхідно визначити її ім'я, розмір, а також файли і групи файлів, у яких вона буде зберігатися.

SQL Server створює нову БД у два етапи:

1. Використовуючи копію бази Model, SQL Server ініціалізує нову та її метадані.

2. Після цього SQL Server заповнює частину, що залишилася, БД (крім сторінок із внутрішніми даними, що відбивають використання дискового простору, зайнятого БД) порожніми сторінками.

6.1.2 Приклад створення БД

Як приклад БД, яка буде створена програмно за допомогою операторів мови Transact SQL, виберемо БД «Книжкова справа» (рис. 6.1).

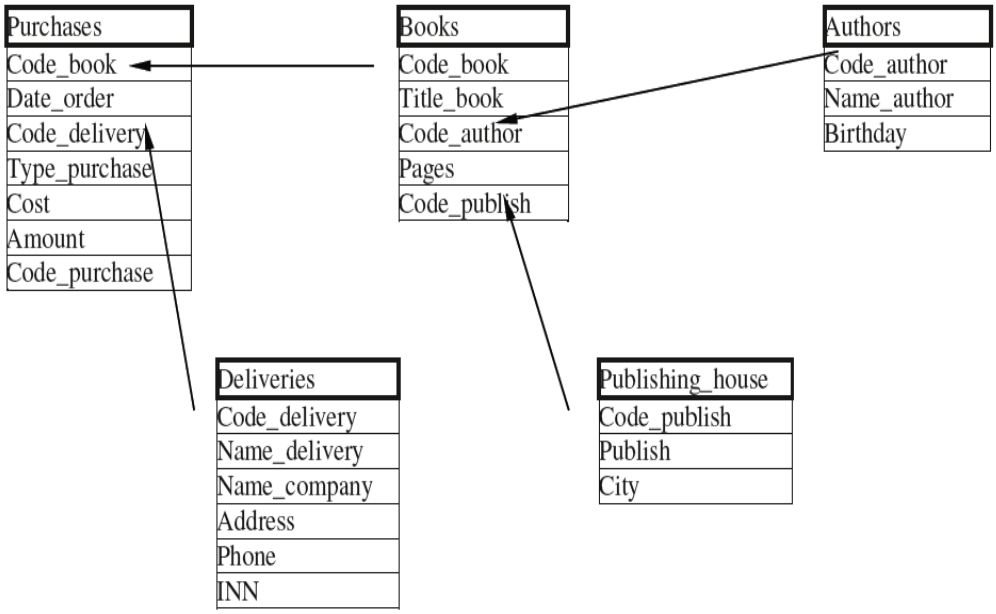

Рисунок 6.1 – Фрагмент БД «Книжкова справа»

Структура таблиць даної БД наведена в табл. 6.1–6.5.

Таблиця 6.1 – Покупки (назва таблиці Purchases)

Назва поля	Тип поля	Опис поля
Code_book	Int	Код книги, яку купують
Date_order	DateTime	Дата замовлення книги
Code_delivery	Int	Код постачальника
Type_purchase	Bit	Тип закупівлі (опт/роздріб)
Cost	Money	Вартість одиниці товару
Amount	Int	Кількість екземплярів
Code_purchase	Int	Код покупки

Таблиця 6.2 – Довідник книг (Назва таблиці Books)

Назва поля	Тип поля	Опис поля
Code_book	Int	Код книги
Title_book	Char	Назва книги
Code_author	Int	Код автора
Pages	Int	Кількість сторінок
Code_publish	Int	Код видавництва

Таблиця 6.3 – Довідник авторів (назва таблиці Authors)

Назва поля	Тип поля	Опис поля
Code_author	Int	Код автора
Name_author	Char	ПІБ автора
Birthday	DateTime	Дата народження

Таблиця 6.4 – Довідник постачальників (назва таблиці Deliveries)

Назва поля	Тип поля	Опис поля
Code_delivery	Int	Код постачальника
Name_delivery	Char	ПІБ відповідальної особи
Name_company	Char	Назва компанії-постачальника
Address	Char	Юридична адреса
Phone	Numeric	Контактний телефон
INN	Char	ІПН

Таблиця 6.5 – Довідник видавництв (назва таблиціPublishing_house)

Назва поля	Тип поля	Опис поля
Code_publish	Int	Код видавництва
Publish	Char	Видавництво
City	Char	Місто

Запустити SQL Server Management Studio, перевірити включення сервера. Для запуску MS SQL Server 2005 оберіть утиліту SQL Server Management Studio та запустіть її. Для написання програмного коду в SQL Server Management Studio потрібно натиснути кнопку «Створити запит» («New query») на панелі інструментів «Стандартна» («Standart»).

Створити нову БД з назвою DB_Books за допомогою команди: CREATE DATABASE DB_BOOKS.

Для виконання команди натиснути F5.

Відкрити утиліту SQL Server Management Studio. Перевірити наявність БД DB_Books, якщо її не бачите в розділі DataBases, то натисніть F5 для оновлення. Створена БД матиме наступний вигляд (рис.6.2):

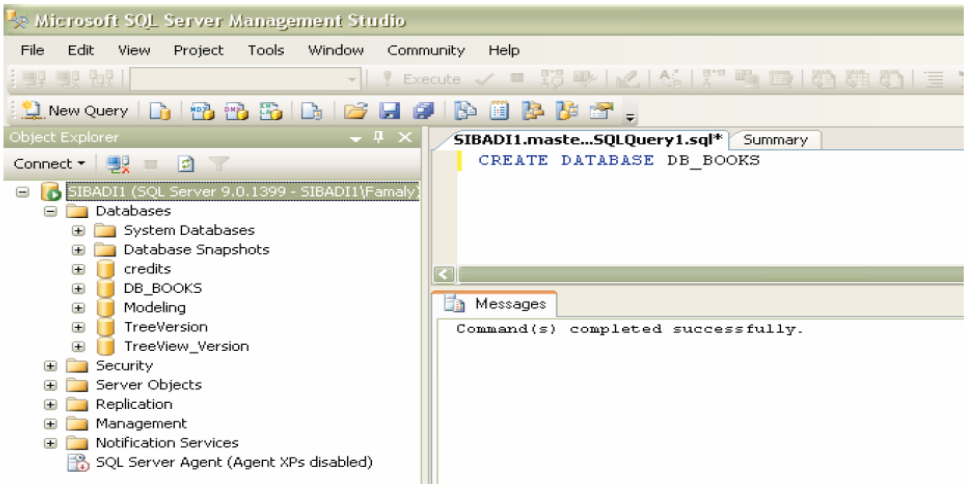

Рисунок 6.2 – Результат створення БД

Створити в ній перераховані таблиці за допомогою нижченаведнених команд (для створення нової сторінки для коду в SQL Server Management Studio натиснути кнопку «Створити запит»):

use DB_BOOKS

CREATE TABLE Authors(Code_author INT PRIMARY KEY, name_author CHAR(30), Birthday DATETIME)

CREATE TABLE Publishing_house(Code_publish INT PRIMARY KEY, Publish CHAR(30), City CHAR(20))

CREATE TABLE Books(Code_book INT PRIMARY KEY, Title_bookCHAR(40), Code_author INT FOREIGN KEY REFERENCES Authors(Code_author), Pages INT, Code_publish INT FOREIGN KEYREFERENCES Publishing_house(Code_publish))

CREATE TABLE Deliveries(Code_delivery INT PRIMARY KEY, Name_delivery CHAR(30), Name_company CHAR(20), AddressVARCHAR(100), Phone BIGINT, INN CHAR(13))

CREATE TABLE Purchases(Code_purchase INT PRIMARY KEY, Code_book INT FOREIGN KEY REFERENCES Books(Code_book), Date_order SMALLDATETIME, Code_delivery INT FOREIGN KEYREFERENCES Deliveries(Code_delivery), Type_purchase BIT, Cost FLOAT, Amount INT)

Запустіть команду клавішею F5.

В утиліті SQL Server Management Studio перевірити наявність БД DB_Books і таблиць в ній.

У розділі діаграм створити нову діаграму, в яку додати зі списку п'ять наших таблиць, перевірити зв'язки між таблицями (рис. 6.3).

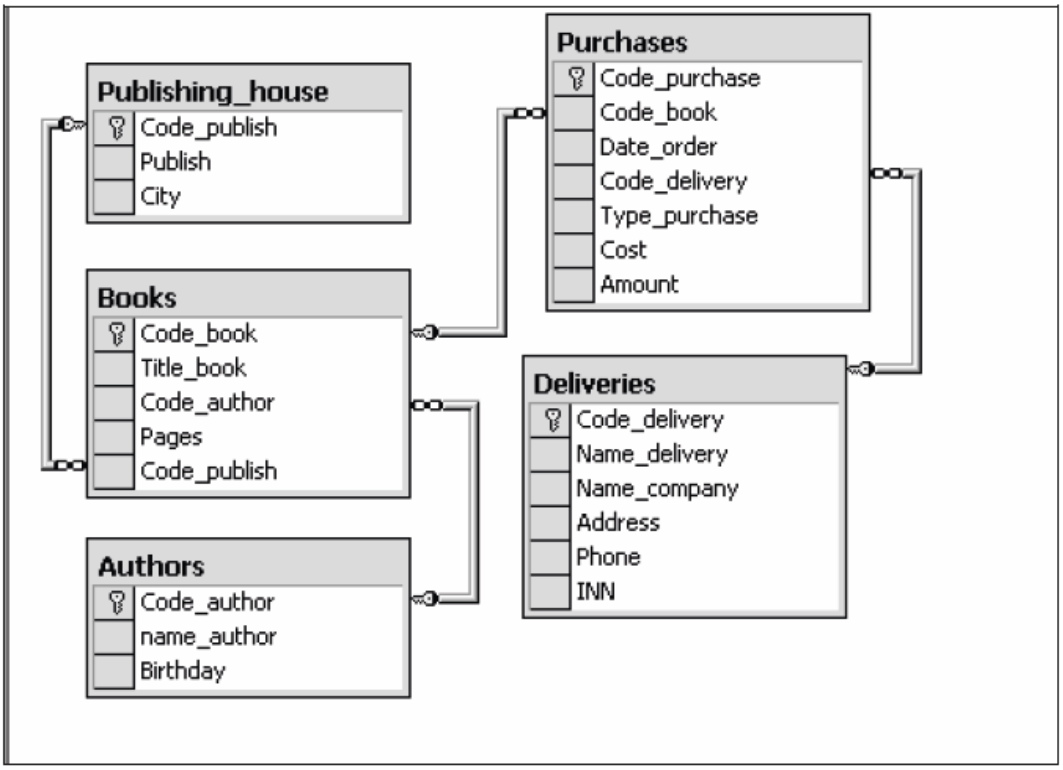

Рисунок 6.3 – Результат створення діаграми

Використані оператори:

PRIMARY KEY – ознака створення ключового поля.

FOREIGN KEY ... REFERENCES ... – ознака створення поля зв'язку з іншою таблицею.

CREATE TABLE – команда створення таблиці в поточній БД.

USE - зробити активною конкретну БД.

CREATE DATABASE – команда створення нової БД.

6.2 Завданнядолабораторної роботи

В утиліті SQL Server Management Studio створити нову БД за допомогою оператора Create Database, назву БД визначити, виходячи з предметної області. Закоментувати оператор (-- – однорядковий коментар, /* */– багаторядковий коментар). Програмно зробити активною створену БД за допомогою оператора Use. Створити перераховані таблиці за допомогою операторів Create table, причому самостійно визначити типи таблиць (батьківськачи підпорядкована), типи полів і їх розміри, знайти поля типу Primary key і Foreign key. Зберегти файл програми з назвою ПрізвищеСтудента_Лаб_1_№варіанта. У SQL Server Management Studio в розділі діаграм створеної БД згенерувати нову діаграму, перевірити зв'язки між таблицями.

6.3 Контрольні запитання:

1. Яке призначення і склад оператора SELECT.
2. Назвіть вимоги до порядку розміщення стовпців в операторі SELECT.
3. Яка особливість використання символу (*) в операторі SELECT.
4. Охарактеризуйте призначення пропозиції оператора SELECT – FROM.
5. Яке призначення пропозиції оператора SELECT – WHERE.
6. Яка суть пошуку за шаблоном.
7. Які особливості використання ключових слів AND і OR.
8. Яке призначення пропозиції оператора SELECT – ORDER BY.
9. Яке призначення і склад оператора INSERT.
10. Яке призначення і структура оператора UPDATE.

ЛАБОРАТОРНАЯ РАБОТА № 7
ВИКОРИСТАННЯ ОПЕРАТОРІВ МАНІПУЛЮВАННЯ ДАНИМИ В MICROSOFT SQL SERVER

Мета: навчитися використовувати оператори маніпулювання даними Select, Insert, Update, Delete.

7.1 Пояснення до лабораторної роботи

7.1.1 Оператори маніпулювання даними

SQL займає центральне місце у проектуванні та використанні реляційних БД. Будь-який програмний додаток, що взаємодіє з реляційною базою даних, незалежно від його користувальницького інтерфейсу, посилає серверу баз даних оператори SQL.

Оператор SQL складається з набору команд, що виконують певні дії над об'єктами бази даних або даними, що зберігаються в ній. Реляційні БД підтримують три типи операторів SQL: мову визначення даних (Data Definition Language, DDL), мову маніпулювання даними (Data Manipulation Language, DML) і мову управління даними (Data Control Language, DCL).

Мова маніпулювання даними використовується для отримання, занесення, редагування та видалення даних, що містяться в об'єктах, визначених за допомогою DDL. Як основні оператори маніпулювання даними використовуються оператори: SELECT, INSERT, UPDATE і DELETE.

Оператор SELECT здійснює вибірку інформації, що зберігається в БД і дозволяє вибрати один або декілька кортежів з однієї або декількох таблиць. Оператор INSERT додає в таблицю новий кортеж, оператор UPDATE служить для редагування даних, оператор DELETE видаляє кортежі з таблиці.

Оператор **SELECT** є фактично найважливішим для користувача й самим складним оператором SQL. Він призначений для вибірки даних із таблиць, тобто він і реалізує одне з основних призначень БД – надавати інформацію користувачеві.

Оператор SELECT завжди виконується над деякими таблицями, що входять у БД. Насправді в БД можуть бути не тільки постійно збережені таблиці, а також тимчасові таблиці й так звані подання. Подання – це SELECT – вираз, що зберігається в БД. З погляду користувачів подання – це таблиця, яка не зберігається постійно в БД, а "виникає" у момент звертання до неї. З погляду оператора SELECT і постійно збережені таблиці, і тимчасові таблиці, й подання виглядають зовсім однаково. Звичайно при реальному виконанні оператора SELECT системою враховуються розбіжності між збереженими таблицями й поданнями, але ці розбіжності приховані від користувача. Результатом виконання оператора SELECT завжди є таблиця.

Оператор **INSERT** додає в таблицю нову стрічку. Якщо користувач буде вводити всі значення в нову стрічку в порядку, який був визначений при створенні таблиці, то можна вказувати лише ім'я таблиці та список значень, що вводяться.

Якщо список вводу неповний, або порядок значень відрізняється від того, що закладався при створенні таблиці, після назви таблиці вказуєься потрібний список атрибутів. Атрибутам, що не вказані в списку присвоюється значення по замовчуванню, якщо воно вказувалося при створенні таблиці або ж значення NULL.

За допомогою оператора INSERT можна переміщати значення від однієї таблиці до іншої, якщо структура згаданих таблиць ідентична.

Оператор **UPDATE** дозволяє змінювати значення деяких або всіх полів таблиці.

Наприклад, нехай необхідно замінити назву предмета навчання “Математика” (subj_id = 43) на назву “Вища математика”, при цьому ідентифікаційний номер необхідно зберегти

UPDATE subject1

SET subj _name = ‘Вищаматематика’ , HOUR = 36, SEMESTER= 1

WHEREsubj_id = 43;

Усунення стрічок із таблиці здійснюється за допомогою команди **DELETE**. Можна усувати всі стрічки таблиці. В результаті таблиця стає пустою, після чого вона може бути усунена командою DROP TABLE.

Для усунення декількох стрічок застосовується умова WHERE:

DELETEFROMSTUDENT1

WHERECITY = ‘Київ’ ;

В умові WHERE команди DELETE можна застосовувати підзапити. В атрибутах умови FROM підзапиту не можна посилатися на таблицю, з якої здійснюється усунення. Однак можна посилатися на поточну стрічку, що є кандидатом на усунення тобто на стрічку, яка на даний час перевіряється в умові основного запиту.

7.2 Завдання до лабораторної роботи:

Створити нову БД з назвою DB_Books за допомогою оператора Create Database, створити в ній перераховані таблиці с допомогою операторів Create table за прикладом лабораторної роботи № 6. Зберегти файл програми з назвою ПрізвищеСтудента_Лаб_6_DB_Books. В утиліті SQL Server Management Studio за допомогою кнопки «Створити запит» створити окремі програми по кожному запиту, які зберігаються на диску з назвою: ПрізвищеСтудента_Лаб_7_№_завдання. Можна зберігати всі виконані запити в одному файлі. Для перевірки роботи операторів SELECT попередньо створіть програму, яка з допомогою операторів INSERT заповнить всі таблиці БД DB_Books декількома записами, збережіть програми з назвою ПрізвищеСтудента_Лаб_7_Insert. Перелік завдань, які потрібно виконати, наведено в табл. Д.1 (Додаток Д).

Сортування

1. Вибрати всі відомості про книги з таблиці Books і впорядкувати результат за кодом книги (поле Code_book).

2. Вибрати з таблиці Books коди книг, назви і кількість сторінок (поля Code_book, Title_book і Pages), впорядкувати результат за назвами книг (поле Title_book по зростанню) і по полю Pages (по спаданню).

3. Вибрати з таблиці Deliveries список постачальників (поля Name_delivery, Phone і INN), впорядкувати результат по полю INN (по спаданню).

Зміна порядку полів

4. Вибрати всі поля з таблиці Deliveries таким чином, щоб в результаті порядок стовпців був наступним: Name_delivery, INN, Phone, Address, Code_delivery.

5. Вибрати всі поля з таблиці Publishing_house таким чином, щоб в результаті порядок стовпців був наступним: Publish, City, Code_publish.

Вибір деяких полів з двох таблиць

6. Вибрати з таблиці Books назви книг і кількість сторінок (поля Title_book і Pages), а з таблиці Authors вибрати ім’я відповідного автора книги (поле Name_ author).

7. Вибрати з таблиці Books назви книг і кількість сторінок (поля Title_book і Pages), а з таблиці Deliveries вибрати ім’я відповідного постачальника книги (поле Name_delivery).

8. Вибрати з таблиці Books назви книг і кількість сторінок (Поля Title_book і Pages), а з таблиці Publishing_house вибрати назву відповідного видавництва і місця видання (поля Publish і City).

Умова неточного збігу

9. Вибрати з довідника постачальників (таблиця Deliveries) назви компаній, телефони та ІПН (поля Name_company, Phone і INN), у яких назва компанії (поле Name_company) починається з “ВАТ”.

10. Вибрати з таблиці Books назви книг і кількість сторінок (поля Title_book і Pages), а з таблиці Authors вибрати ім’я відповідного автора книги (поле Name_ author), у яких назва книги починається зі слова “Мемуари”.

11. Вибрати з таблиці Authors прізвища, імена, по батькові авторів (поле Name_ author), значення яких починаються з “Іванов”.

Відсутність точного співпадання значень одного з полів.

12. Вивести список назв видавництв (поле Publish) з таблиці Publishing_house, які не перебувають в місті “Харків” (умова по полю City).

13. Вивести список назв книг (поле Title_book) з таблиці Books, які випущені будь-якими видавництвами, крім видавництва “ПітерСофт” (поле Publish з таблиці Publishing_house).

Вибір записів відповідно до діапазону значень (Between)

14. Вивести прізвища, імена, по батькові авторів (поле Name_author) з таблиці Authors, у яких дата народження (поле Birthday) знаходиться в діапазоні 01.01.1840 - 01.06.1860.

15. Вивести список назв книг (поле Title_book з таблиці Books) і кількість примірників (поле Amount з таблиці Purchases), які були закуплені в період з 12.03.2016 по 15.06.2016 (умова по полю Date_order з таблиці Purchases).

16. Вивести список назв книг (поле Title_book) і кількість сторінок (поле Pages) з таблиці Books, у яких обсяг в сторінках належить діапазону 200–300 (умова по полю Pages).

17. Вивести список прізвищ, імен, по батькові авторів (поле Name_author) з таблиці Authors, у яких прізвище починається на одну з букв діапазону “В”–“Г” (умова по полю Name_author).

Вибір записів відповідно до діапазону значень (In).

18. Вивести список назв книг (поле Title_book з таблиці Books) і кількість (поле Amount з таблиці Purchases), які були надані постачальниками з кодами 3, 7, 9, 11 (умова по полю Code_delivery з таблиці Purchases).

19. Вивести список назв книг (поле Title_book) з таблиці Books, які випущені такими видавництвами: “Київ-Софт”, “Альфа”, “Наука” (умова по полю Publish з таблиці Publishing_house).

20. Вивести список назв книг (поле Title_book) з таблиці Books, які написані наступними авторами: “Леся Українка”, “Іван Франко”, “Тарас Шевченко” (умова по полю Name_author з таблиці Authors).

Вибір записів з використанням Like

21. Вивести список авторів (поле Name_author) з таблиці Authors, які починаються на букву “К”.

22. Вивести назви видавництв (поле Publish) з таблиці Publishing_house, які містять в назві поєднання “софт”.

23. Вибрати назви компаній (поле Name_company) з таблиці Deliveries, у яких значення закінчується на “ський”.

Вибір записів відповідно до декількох умов

24. Вибрати коди постачальників (поле Code_delivery), дати замовлень (поле Date_order) і назви книг (поле Title_book), якщо кількість книг (поле Amount) в замовленні більше 100 або ціна (поле Cost) за книгу знаходиться в діапазоні від 200 до 500.

25. Вибрати коди авторів (поле Code_author), імена авторів (поле Name_author), назви відповідних книг (поле Title_book), якщо код видавництва (поле Code_Publish) знаходиться в діапазоні від 10 до 25 і кількість сторінок (поле Pages) в книзі більше 120.

26. Вивести список видавництв (поле Publish) з таблиці Publish-ing_house, в яких випущені книги, назви яких (поле Title_book) починаються зі слова “Праці” і місто видання (поле City) – “Харків”.

Багатотабличні запити (вибірка з двох таблиць, вибірка з трьох таблиць з використанням JOIN)

27. Вивести список назв компаній-постачальників (поле Name_company) і назви книг (поле Title_book), які вони постачали в період з 01.01.2016 по 31.12.2016 (умова по полю Date_order).

28. Вивести список авторів (поле Name_author), книги яких були випущені у видавництві “Мир” (умова по полю Publish).

29. Вивести список постачальників (поле Name_company), які постачають книги видавництва “Махаон” (умова по полю Publish).

30. Вивести список авторів (поле Name_author) і назви книг (поле Title_book), які були поставлені постачальником “ВАТ Книготорг” (умова по полю Name_company).

Обчислення

31. Вивести сумарну вартість партії однойменних книг (використовуючи поля Amount і Cost) і назву книги (поле Title_book) в кожній поставці.

32. Вивести вартість однієї друкованої сторінки кожної книги (використовувати поля Cost і Pages) і назви відповідних книг (поле Title_book).

33. Вивести кількість років з моменту народження авторів (використовувати поле Birthday) і імена відповідних авторів (поле Name_author).

Обчислення підсумкових значень з використанням агрегатних функцій

34. Вивести загальну суму поставок книг (використовувати поле Cost), виконаних “ЗАТ Опторг” (умова по полю Name_company).

35. Вивести загальну кількість всіх поставок (використовувати будь-яке поле з таблиці Purchases), виконаних в період з 01.01.2016 по 01.02.2016 (умова по полю Date_order).

36. Вивести середню вартість (використати поле Cost) і середню кількість екземплярів книг (використати поле Amount) в одній поставці, де автором книги є “Олесь Гончар” (поле Name_author).

37. Вивести всі відомості про постачання (всі поля таблиці Purchases), а також назву книги (поле Title_book) з мінімальною загальною вартістю (використовувати поля Cost і Amount).

38. Вивести всі відомості про постачання (всі поля таблиці Purchases), а також назву книги (поле Title_book) з максимальною загальною вартістю (використовувати поля Cost і Amount).

Зміна найменувань полів

39. Вивести назву книги (поле Title_book), сумарну вартість партії однойменних книг (використовувати поля Amount і Cost), помістивши результат в поле з назвою Itogo, поставки за період з 01.01.2016 по 01.06.2016 (умова по полю Date_order).

40. Вивести вартість однієї друкованої сторінки кожної книги (використати поля Cost і Pages), помістивши результат в поле з назвою One_page, і назви відповідних книг (поле Title_book).

41. Вивести загальну суму поставок книг (використовувати поле Cost) і помістити результат в поле з назвою Sum_cost, виконаних "ВАТ Луч" (умова по полю Name_company).

Використання змінних в умові

42. Вивести список угод (всі поля з таблиці Purchases) за останній місяць (умова з використанням поля Date_order).

43. Вивести список авторів (поле Name_author), вік яких менше заданого користувачем (умова з використанням поля Birthday).

44. Вивести список книг (поле Title_book), яких закуплено менше, ніж зазначено в запиті користувача (умова з використанням поля Amount).

Використання змінних замість назв таблиць

45. Вивести список назв компаній-постачальників (поле Name_company) і назви книг (поле Title_book), які вони поставили.

46. Вивести список авторів (поле Name_author), книги яких були випущені у видавництвах "СВІТ", "НАУКА" (умова по полю Publish).

47. Вивести список видавництв (поле Name_company), книги, що були продані за ціною 350 грн. (Поле Cost).

Вибір результату в курсор

48. Вивести список назв книг (поле Title_book) і кількості сторінок (поле Pages) в кожній книзі і помістити результат в курсор з назвою Temp1.

49. Вивести список назв компаній-постачальників (поле Name_company) і помістити результат в курсор з назвою Temp2.

50. Вивести список авторів (поле Name_author) і помістити результат в курсор з назвою Temp3.

Використання функцій спільно з підзапитом

51. Вивести список книг (поле Title_book), у яких кількість сторінок (поле Pages) більше середньої кількості сторінок усіх книг в таблиці.

52. Вивести список авторів (поле Name_author), вік яких менше середнього віку всіх авторів в таблиці (умова по полю Birthday).

53. Вивести список книг (поле Title_book), у яких кількість сторінок (поле Pages) дорівнює мінімальній кількості сторінок книг, представлених в таблиці.

Використання квантора існування в запитах

54. Вивести список видавництв (поле Publish), книги яких були придбані оптом ("опт" з поля Type_Purchase).

55. Вивести список авторів (поле Name_author), книг яких немає в таблиці Books.

56. Вивести список книг (поле Title_book), які були поставлені постачальником "ЗАТ Квантор" (умова по полю Name_company).

Оператор обробки даних Update

57. Змінити в таблиці Books вміст поля Pages на 300, якщо код автора рівний 56 (поле Code_author) і назва книги (поле Title_book) – "Мемуари".

58. Змінити в таблиці Deliveries вміст поля Address на "немає відомостей", якщо значення поля є порожнім.

59. Збільшити в таблиці Purchases ціну (поле Cost) на 20 відсотків, якщо замовлення було оформлено протягом останнього місяця (умова по полю Date_order).

Оператор обробки даних Insert

60. Додати в таблицю Purchases новий запис, причому так, щоб код покупки (поле Code_purchase) було автоматично збільшено на одиницю, а в тип закупівлі (поле Type_purchase) внести значення “опт”.

61. Додати в таблицю Books новий запис, причому замість ключового поля поставити код (поле Code_book), автоматично збільшений на одиницю від максимального коду в таблиці, замість назви книги (поле Title_book) написати “Наука. Техніка. Інновації”.

62. Додати в таблицю Publish_house новий запис, причому замість ключового поля поставити код (поле Code_publish), автоматично збільшений на одиницю від максимального коду в таблиці, замість назви міста – “Київ” (поле City), замість видавництва – “Наука” (поле Publish).

Оператор обробки даних Delete

63. Видалити з таблиці Purchases всі записи, у яких кількість книг в замовленні (поле Amount) = 0.

64. Видалити з таблиці Authors всі записи, у яких немає імені автора в полі Name_Author.

65. Видалити з таблиці Deliveries всі записи, у яких не зазначено ІПН (поле INN порожнє).

7.3 Контрольні запитання:

1. Яке основне призначення оператора SELECT?
2. Які головні властивості результуючого набору описує більшість операторів SELECT?
3. Напишіть загальну структуру конструкцій оператора SELECT.
4. Які конструкції оператора SELECT є обов’язковими та з якою метою вони використовуються?
5. Що таке підзапити і для чого вони використовуються?
6. Дайте визначення курсору.
7. Підтримку яких функцій забезпечують курсори?
8. Назвіть методи додавання інформації в БД.
9. Опишіть призначення, структуру та порядок застосування оператора INSERT.
10. Опишіть призначення, структуру та порядок застосування оператора UPDATE.
11. Опишіть призначення, структуру та порядок застосування операторів, що видаляють дані з базових таблиць.

ЛАБОРАТОРНА РОБОТА № 8
СТВОРЕННЯ КЛІЄНТСЬКОЇ ЧАСТИНИ ДОДАТКУ ДЛЯ ПЕРЕГЛЯДУ, РЕДАГУВАННЯ ДАНИХ БД. ВИКЛИК ПРОЦЕДУР З КЛІЄНТСЬКОЇ ЧАСТИНИ

Мета роботи: навчитися створювати клієнтську програму для роботи з БД із застосуванням вбудованих інструментів на Visual C # 2005.

8.1 Пояснення до виконання роботи

Для створення клієнтського додатка на Visual C # 2005 використовуємо приклад БД с назвою **DB_Books**, яка була створена в лабораторній роботі №6. При виконанні прикладів і завдань звертайте увагу на відповідність назв БД, таблиць та інших об'єктів проекту.

На **Visual C # 2005**:

1. У проекті вибираємо меню Tools => Connect to DataBase.
2. У вікні, яке відкрилося, в полі Data Source ставимо Microsoft SQL Server, в поле Server Name – SQLEXPRESS, далі в поле Select or enter DB name обираємо ім'я БД, до якої будемо підключатися, і натискаємо OK.
3. Тепер, відкривши вікно Server explorer, можна побачити підключену БД. Натиснувши на неї, у вікні властивостей копіюємо Connection String, ще буде в нагоді.
4. На форму додати 5 компонентів типу DataGridView (перейменувати компоненти на Purchases, Books, Authors, Deliveries, Publish).
5. У вкладці Data виберемо Add New Data Source. У вікні, яке з'явилось, виберемо DataBase і натиснемо Next. Вибираємо нашу БД, тиснемо Next. У вікні, яке з'явилось, поставимо галочку на пункті Table (вибираємо всі створені таблиці). Тиснемо Finish.
6. У кожній таблиці DataGridView змінимо властивість DataSource на відповідно назві цієї таблиці (рис.8.1):

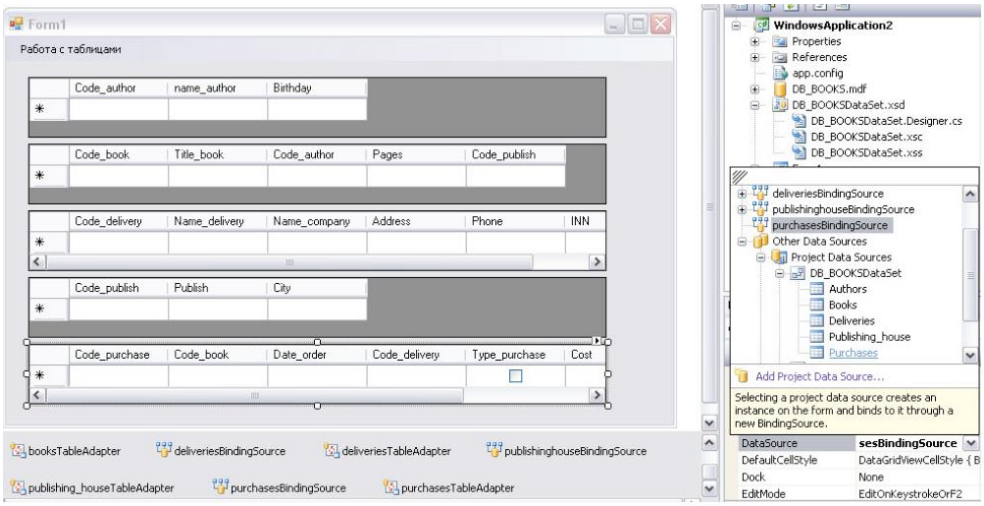

Рисунок 8.1

7. На основній формі (Form1) додати компонент. У редакторі меню зробити перший пункт «Робота з таблицями» і в підменю пункти: «Автори»,«Книги»,«Видавництва»,«Постачальники»,«Поставки».

8. Створити п'ять форм, кожну з яких назвати: FormAuthors, FormPurchases, FormBooks, FormDeliveries, FormPublish.

9. На основній формі в підпунктах меню в відповідних методах Click викликати відповідні форми за допомогою коду:

дляFormAuthors:

```
FormAuthors myForm2 = new FormAuthors();
myForm2.Show();
```

дляFormPurchases:

```
FormPurchases myForm3 = new FormPurchases();
myForm3.Show();
```

дляFormBooks:

```
FormBooks myForm4 = new FormBooks();
myForm4.Show();
```

дляFormDeliveries:

```
FormDeliveries myForm5 = new FormDeliveries();
myForm5.Show();
```

дляFormPublish:

```
FormPublish myForm6 = new FormPublish();
myForm6.Show();
```

10. На форми FormAuthors, FormPurchases, FormBooks, FormDeliveries, FormPublish додати по парі компонент типу DataGridView і Binding-Navigator. Налаштувати у DataGridView властивість DataSource для зв'язку з відповідним джерелом даних. Потім необхідно налаштувати у BindingNavigator властивість BindingSource для зв'язку зі створеною таблицею (значення має збігатися із значенням властивості елемента DataGridView).

11. Перевірити роботу додатка.

12. На форму FormBooks додати 3 компонента типу TextBox і 2 компонента ComboBox.

У 1-го компонента TextBox змінити властивості:

(DataBinding)

Text booksBindingSource - Code_book

У 2-го компонента TextBox змінити властивості: (DataBinding)

Text booksBindingSource - Title_book

У 1-го компонента ComboBox змінити властивості: (DataBinding)

SelectedValue booksBindingSource - Code_author

DataSource authorsBindingSource

DisplayMember name_author

ValueMember Code_author

У 3-го компонента TextBox змінити властивості: (DataBinding)

Text booksBindingSource - Pages

У 2-го компонента ComboBox змінити властивості: (DataBinding)

SelectedValue booksBindingSource - Code_publish

DataSource publishinghouseBindingSource

DisplayMember Publish

ValueMember Code_publish

13. У компонента DataGridView прибрати всі галочки з властивостей редагування і додавання.

14. На форму FormBooks додати компонент типу Button (кнопка оновлення даних), властивість Text змінити на «Оновити» та прописати подію Click:

```
this.Validate();
this.booksBindingSource.EndEdit();
```

this.booksTableAdapter.Update(this.dB_BOOKSDataSet.Books);

15. Аналогічно для інших форм додати елементи типу TextBox

16. Перевірити роботу додатка.

17. На форму FormBooks додати 5 компонентів типу Button.

У 1-го компонента Button змінити властивості і метод:

Text Фільтр по поточному видавництву;

У методі Click кнопки написати код:

```
int bb = dataGridView1.CurrentCell.RowIndex;
booksBindingSource.Filter = "Code_Publish = " +
dataGridView1[4,bb].Value;.
```

У 2-го компонента Button змінити властивості і метод:

Text Фільтр за поточною назвою книги.

У методі Click кнопки написати код:

```
int bb = dataGridView1.CurrentCell.RowIndex;
booksBindingSource.Filter = "Title_book = " +
dataGridView1[1, bb].Value;.
```

У 3-го компонента Button змінити властивості і метод:

Text Фільтр по поточному автору.

У методі Click кнопки написати код:

```
int bb = dataGridView1.CurrentCell.RowIndex;
booksBindingSource.Filter = "Code_Author = " +
dataGridView1[0, bb].Value;.
```

У 4-го компонента Button змінити властивості і метод:

Text Фільтр за кількістю книг.

У методі Click кнопки написати код:

```
int bb = dataGridView1.CurrentCell.RowIndex;
booksBindingSource.Filter = "Pages = " +
dataGridView1[3, bb].Value;.
```

У 5-го компонента Button змінити властивості і метод:

Text Зняти фільтр.

У методі Click кнопки написати код:

```
booksBindingSource.Filter = "";.
```

18. Аналогічно для інших форм додати елементи типу Button, які будуть запускати фільтри по відповідним значенням полів поточного запису в об'єкті Grid.

19. Перевірити роботу додатка.

20. Створити форму під назвою FormProcedure.

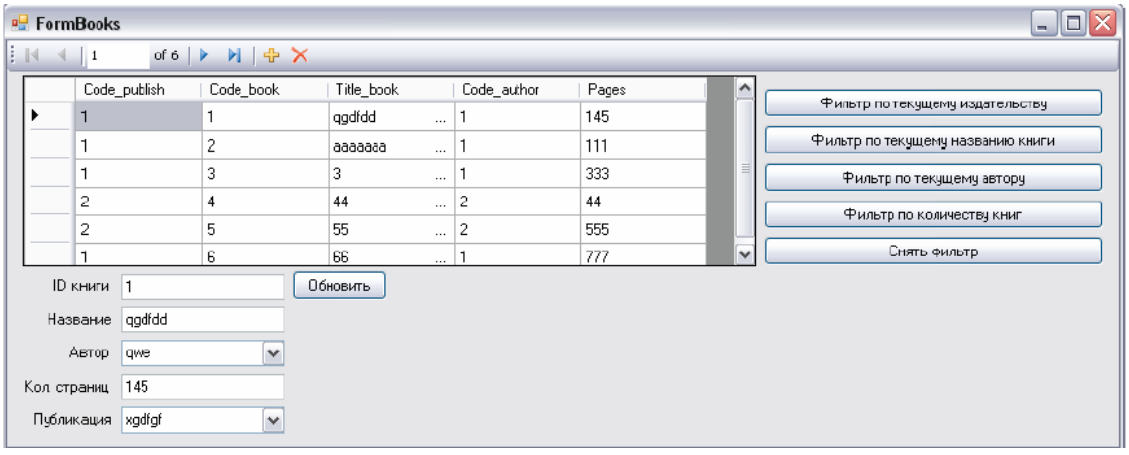

Рисунок 8.2 —Приклад розташування компонентів на формі FromBooks

21. Додати на головній формі в меню пункт з назвою Робота з процедурами. У методі Click пункту меню написати код для запуску форми FormProcedure (див. приклад коду в пункті 9 даної лабораторної роботи).

22. Зайти Tool -> Choose Toolbox Items. Поставити галочки на елементах SqlCommand і SqlConnection, застосувати зміни.

23. Додати на форму компонент SqlConnection і у властивості ConnectionString вибрати DB_DOOK.mdf

24. Тепер можна підключити збережену процедуру Count_purchases, яку слід виконати таким чином:

Приклад створення процедури з вхідним і вихідним параметрами.

Створити процедуру для визначення кількості замовлень, зроблених за вказаний період:

```
CREATE PROC count_purchases
@ D1 SMALLDATETIME, @ d2 SMALLDATETIME,
@c INT OUTPUT
AS
SELECT @ c = count (Code_purchase) from Purchases WHERE Date_order
BETWEEN @ d1 AND @ d2
SET @c = ISNULL (@ c, 0)
```

Завдання. Створіть дану процедуру в розділі Stored Procedures БД DB_Books через утиліту SQL server Management Studio.

Запустити її за допомогою команд:

```
DECLARE @ c2 INT
EXEC count_purchases '01 -jun-2006 ', '01 -jul-2006', @ c2 OUTPUT
SELECT @ c2
```

На форму FormProcedure додати компонент SqlCommand. Змінити такі його властивості:

Connection на SqlConnection1;

CommandType на StoredProcedure;

CommantText на Count_purchases.

25. У компонента SqlCommand1 вибрати властивість Parameters і у властивостях кожного вхідного параметра виправити властивість SqlDbType – на DateTime, а для вихідного параметра властивість Value – Int. Також, якщо параметр зі значенням ReturnValue (параметр Direction) не створений, то необхідно створити його (він повинен зверху) і задати йому ім'я @ReturnValue з властивістю SqlDbType - Int.

26. На форму FormProcedure додати 3 компонента типу TextBox (Імена відповідно TextBox1, TextBox2, TextBox3) і 1 компонент типу Button. Поруч з кожним компонентом TextBox поставити Label і виправити їх властивості Text відповідно на «Кількість покупок за вказаний період»,«Введіть дату початку періоду»,«Введіть дату кінця періоду».

27. На кнопці змінити назву на «Виконати запит». У методі Click кнопки написати наступний код:

```
int count_save;
sqlCommand1.Parameters["@d1"].Value =
Convert.ToDateTime(textBox1.Text);
sqlCommand1.Parameters["@d2"].Value =
Convert.ToDateTime(textBox2.Text);
sqlConnection1.Open();
sqlCommand1.ExecuteNonQuery();
sqlConnection1.Close();
count_save = (int)sqlCommand1.Parameters["@ReturnValue"].Value;
```

```
textBox3.Text = Convert.ToString(count_save);
```

28. Перевірити роботу додатка.

8.2 Завдання до лабораторної роботи

На Visual C # 2005 створити новий проект, далі для індивідуальноїБД, створеної в лабораторній роботі №6, створити інтерфейс, що включає всі функції і процедури, які описані в даній лабораторній роботі.

8.3 Контрольні запитання

Запитання при перевірці виконаного завдання, що стосуються використаних функцій і процедур, їх призначення та використання.

ЛАБОРАТОРНА РОБОТА № 9
СТВОРЕННЯ ЗВІТНИХ ФОРМ В КЛІЄНТСЬКОМУ ДОДАТКУ

Мета роботи: навчитися створювати форми звітних документів за даними БД.

9.1 Пояснення до виконання роботи

Для виконання трьох перших завдань використовуємо приклад БДз назвою **DB_Books**, яка була створена в лабораторній роботі №6.

При виконанні прикладів і завдань звертайте увагу на відповідність назв БД, таблиць та інших об'єктів проекту.

Звіти багато в чому схожі на форми і теж дозволяють отримати результати роботи запитів в наочній формі, але тільки не на екрані, а ввигляді роздруківки на принтері. Таким чином, в результаті роботи звітустворюється паперовий документ.

На Visual C # 2005 є кілька способів створення звітів. Один із способів створення звітів це використання генератора звіту FastReport. Після встановлення генератора необхідно перезапустити Visual C # 2005. Потім необхідно додати компоненти FastReport як в лабораторній роботі № 8 в пункті 22.

Для створення звіту необхідно помістити компонент Report на головну форму. Після цього подвійним клацанням натиснути на компонент і вибрати дані, які нам потрібні для складання звіту. Після цього відкриється сам редактор звітів. Для збереження змін потрібно просто зберегти файл звіту в будь-якому місці.

Структура звіту

Звіти складаються з розділів або секцій (Bands), а розділи можуть містити елементи управління. Для налаштування розділів треба натиснути на робочій області на кнопку «**Налаштувати бенди**».

1. Структура звіту складається з наступних розділів: *заголовок звіту, підвал звіту, заголовок сторінки, підвал сторінки, область даних, заголовок колонки, підвал колонки, фоновий.*

2. Розділ *заголовок звіту* служить для друку загального заголовка звіту.

3. Розділ *заголовок сторінки* можна використовувати для друку підзаголовків, якщо звіт має складну структуру і займає багато сторінок.Тут можна також розміщувати і номери сторінок, якщо це не зроблено вколонтитулі.

4. В *області даних* розміщують елементи управління, пов'язані з вмістом полів таблиць бази. У ці елементи керування видаютьсядані з таблиць для друку на принтері. Ці розділи будуть друкуватися стільки разів, скільки записів є в зв’язному запиті або таблиці.

5. Розділ *підвал сторінки* використовують для тих же цілей, що і розділ*заголовок сторінки*. Можна використовувати для підстановки полів дляпідписів посадових осіб, якщо є необхідність підписувати звітна кожній сторінці.

6. Розділи колонок використовують для розміщення додаткової інформації або підсумкової інформації за всіма даними звіту. Друкується зверху чи знизу *області даних*.

7. Для попереднього перегляду звіту в тому вигляді, як він буде розташований на папері, необхідно викликати метод Show компонента Report (На головній формі в меню додати розділ і в методі Click написати цей метод, наприклад Report1.Show ()). Приклад звіту в режимі «конструктор» наведено на рис. 9.1, а в режимі попереднього перегляду – на рис. 9.2.

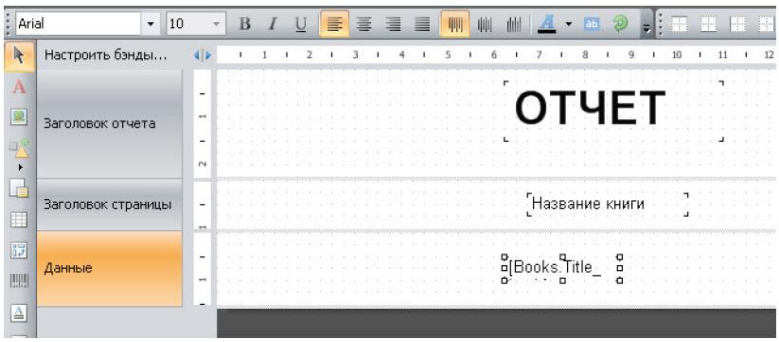

Рисунок 9.1 –Приклад звіту в режимі «конструктор»

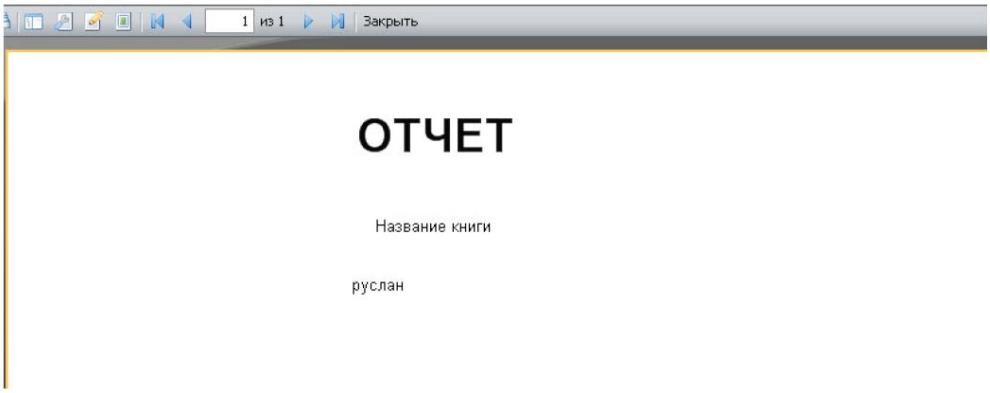

Рисунок 9.2 –Приклад звіту в попередньому перегляді

Завдання 1. Створення звіту в табличній формі, який вибирає з таблиці Books всі поля, крім кодів, з таблиці Publish_house – назви видавництв і місце видавництва, з таблиці Authors – ім'я автора.

1. У проекті на головній формі в меню додати пункт меню Звіти, а також підпункти:

Звіт в табличній формі;

Звіт у вільній формі;

Звіт по двох таблицях.

2. У проекті на головну форму додати 3 компонента Report

3. У першого компонента Report змінити DataSet (стрілка на компоненті Task -> Select DataSet) на відповідні дані, які необхідні для звіту. Відкрити вікно дизайну звіту (подвійним клацанням по компоненті).

4. У властивостях включити такі розділи (налаштувати бенди), як *заголовок звіту, заголовок даних, дані*.

5. У розділі *заголовок звіту* розмістити мітку (компонент Текст). У властивостях змінити його зовнішній вигляд і підпис «Приклад табличного звіту».

6. У розділі *заголовок даних* встановити компонент Таблиця (для імітації обрамлення шапки таблиці) і написати в ній - Назва книги, Автор, Видавництво.

7. На розділ *дані* перетягнути об'єкти з панелі *дані* наступним чином:

Джерела даних -> Books -> title_book

Джерела даних -> Books -> Autors -> name_autor

Джерела даних -> Books -> Publishing_house -> publish

Розташувати компоненти симетрично під написами в таблиці.

8. У головній формі програми в підпункті Звіт в табличній формі в методі Click написати команду: Report1.Show ().

9. Запустити програму, перевірити роботу.

Завдання 2. Створення звіту у вільній формі з даними з першого завдання. Створимо картку книги для бібліотечної картотеки.

Особливість звіту у вільній формі в тому, що він створює шаблон на кожний окремий запис таблиці, іншими словами, він створюється задокументам, у яких немає шапки і приміток. Прикладом таких документів може служити прибутковий або видатковий касовий ордер, етикетка для товару або цінник в магазині, лист-запрошення і т.д.

1. У другому компоненті Report встановити властивості DataSet на необхідні. У властивостях (налаштувати бенди) включити розділ *дані*.

2. На розділ *дані* перетягнути об'єкти з панелі *дані* наступним чином:

Джерела даних -> Books -> title_book

Джерела даних -> Autors -> name_autor

Джерела даних -> Publishing_house -> publish

3. У головній формі програми в підпункті Звіт у вільній формі в методі Click написати команду: Report2.Show ().

4. Запустити програму, перевірити роботу.

Завдання 3. Створення звіту за двома таблицями. Створимо звіт, в якому спочатку будуть виводитися дані автора книги зтаблиці Authors, а потім список книг, які написав цей автор.

1. У третьому компоненті Report встановити властивості DataSet на необхідні. У властивостях (налаштувати бенди) включити розділи: *заголовок звіту, дані.*

2. У розділі *заголовок звіту* розмістити мітку (компонент Текст). У властивостях змінити її зовнішній вигляд і підпис «Звіт по авторам і написаним книгам».

3. Викликати майстер угрупування. Панель Звіт -> Майстер угрупування. В якості умови угруповання вказати поле, по якому буде здійснюватися угрупування даних: Autors -> name_autor. Натиснути Додати.

4. В результаті отримаємо «бенди»: Тема групи (містить ім'я автора),Дані, Підвал групи. На розділ *дані* перетягнути об'єкти з панелідані наступним чином:

Джерела даних -> Autors -> Books ->title_book.

Приклад наведено на рис. 9.3.

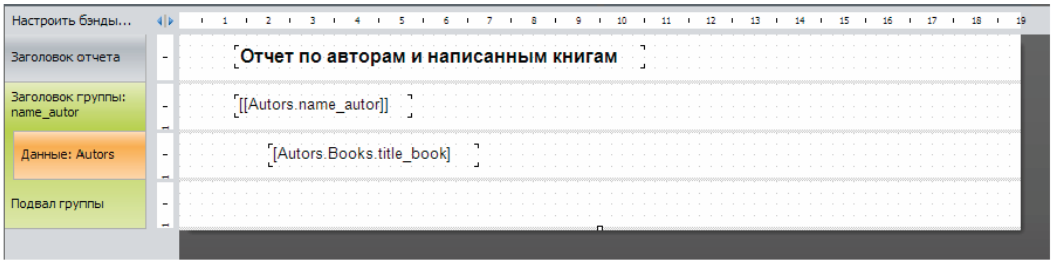

Рисунок 9.3 –Приклад звіту

5. У головній формі програми в підпункті Звіт подвох таблицях в методі Click написати команду: Report3.Show ().

6. Запустити програму, перевірити роботу.

9.3 Завдання до лабораторної роботи

За індивідуальною БД, яка видана за варіантами (з лабораторної роботи №6) зробити в клієнтському додатку чотири звіти, які будуть запускатися через меню головної форми:

- Звіт в табличній формі по одному з довідників, причому в розділі «Примітка» вивести підсумкову кількість записів в звіті;
- Звіт у вільній формі. Виберіть одну з таблиць, по якій можна зробити або бейдж, або цінник, або запрошення. При створенні звіту використовуйте рисунокяк підкладку;
- Звіт за запитом. З'єднайте дані всіх трьох таблиць, кодові поля в запит не пишіть. Створіть звіт в табличній формі з підсумковим полем в розділі «Примітка» (це може бути сума або кількість і т.п.,в залежності від вмісту запитів). Кожен рядок в звіті повинен мати номер по порядку. *Наприклад*:

1) Крупа 10 кг
2) Борошно 20 кг
і т.д.;

- Звіт по декількох таблицях. Виберіть одну пару пов'язаних таблиць, визначте головну і залежну таблиці і зробітьзвіт в табличній формі, в якому дані з головної таблиці розшифровуються (доповнюються) даними з залежної таблиці.

9.4 Контрольні запитання

1. Які способи створення звітів Вам відомі?
2. Які генератори звітів у MicrosoftVisualC# 2005 при створенні звітів Ви використовували?
3. Особливості генератора звітів FastReport.
4. Елементи звіту у FastReport. Перелічіть типи бендів.
5. Назвіть об’єкти, які можна використовувати у звіті генератора звітів FastReport.

ПЕРЕЛІК РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ

1. Delaney K. InsideMicrosoft® SQLServer™ 2005: QueryTuningandOptimization, MicrosoftPress, 2007. –448 p.
2. Microsoft SQL Server 2005. Реализация и обслуживание. Учебный курс Microsoft. / Пер. с англ. – М.: «Русская редакция», СПБ.: «Питер», 2007. – 768 с.
3. Бегг К., Коннолли Т. Базы данных. Проектирование, реализация и сопровождение. Теория и практика. 3-е издание.– М.: Издательский дом «Вильямс». – 2003. – 1440 с.
4. Бейли Л. Изучаем SQL. – СПб.: Питер, 2012. – 592 с.: ил.
5. Бекаревич Ю.Б., Пушкина Н.В. Самоучитель Microsoft Access 2009. – СПб.: БХВ - Петербург, 2009. – 720 с.
6. Бергер А.Б., Горбач И.В., Меломед Э.Л., Степаненко В.П., Щербинин В.А. MicrosoftSQLServer 2005. AnalysisServices. OLAP и многомерный анализ данных.–СПб.: БХВ-Петербург, 2007. –928 с.
7. Богданов М. Р. Visual Basic 2005 на примерах. – СПб.: БХВ-Петербург, 2007. – 592 с.
8. Бондарь Александр Microsoft SQL Server 2012; БХВ-Петербург - Москва, 2013. –608 с.
9. Браст Эндрю Дж., Форте Стивен Разработка приложений на основе MS SQL Server 2005. Мастер-класс. / Пер. с англ. – М.: Издательство «Русская редакция», 2007. – 880 с.
10. Гурвиц Г.А. Microsoft Access 2010. Разработка приложений на реальном примере / Г. Гурвиц. – СПб: БХВ-Петербург. –2012. –498с.
11. Дейт К. Дж. Введение в системы баз данных– К.:Диалектика, 1998. –784 с.
12. Дейт К. Дж. Введение в системы баз данных, 6-е издание. – К.; М.; СПб.: Издательский дом "Вильямс", 2008. – 848 с.
13. Душан Петкович. Microsoft SQL Server 2008. Руководство для начинающих MicrosoftSQLServer 2008: ABeginner'sGuide. – СПб.: БХВ – Петербург, 2009. – 752 с.
14. Ицик Бен-Ган. Microsoft SQL Server 2008. Основы T-SQL Microsoft SQL Server 2008: T-SQL Fundamentals, БХВ - Петербург, Русская Редакция, 2009. – 430 с.
15. Коннолли Т. Базы данных. Проектирование, реализация и сопровождение. Теория и практика. / Т.Коннолли, К. Бегг.–(5-е издание). – СПБ: Вильямс, 2013. – 1440 с.
16. Маклаков С.В. BPWin, ERWin. CASE – средства разработки информационных систем. – М.: Диалог-МИФИ, 2000. –254 с.
17. Мамаев Е.В. MicrosoftSQLServer 2000.− СПб.: БХВ-Петербург, 2005.− 1280 с.

18. Мейер Д. Теория реляционных баз данных. – М.: Мир, 1987. – 608 с.

19. Михеева В.Д., Харитонова И.А. Microsoft Access 2002. – СПб.: БХВ - Петербург, 2007. – 1040 с.

20. Моисеенко С.И., Соболь Б.В. Разработка приложений в Microsoft Office Access. Краткое руководство. Вильямс, 2006. –272 с.

21. Одиночкина С.В. Разработка баз данных в Microsoft Access 2010 / С.В.Одиночкина. – СПб.: НИУ ИТМО. –2012. –180 с.

22. Остринская Л.И., Семенова И.И., Дороболюк Т.Б. Теория и практика работы с современными базами и банками данных: Учебное пособие. − Омск: Изд-во СибАДИ, 2005.− 250 с.

23. Проектирование и реализация баз данных Microsoft SQL Server 2000. Учебный курс MCAD/MCSE, MCDBA/Пер. с англ. – 2-е изд., испр. – М.: Издательско-торговый дом «Русская Редакция», 2003. – 512 с.

24. С. Байдачный, Д. Маленко, Ю. Лозинский SQL Server 2005: Новые возможности для разработчиков– М: СОЛОН-Пресс, 2006. –208 с.

25. Семенова И.И. SQL стандарт в СУБД MSSQLSERVER, ORACLE, VFP И ACCESS: манипулирование данными. − Омск: Изд-во СибАДИ, 2008.− 57 с.

26. Семенова И.И. Разработка клиент-серверных приложений в MicrosoftSQLServer 2005 и MicrosoftVisualC# 2005 ExpressEdition: Учебно-методическое пособие.– Омск: Изд-во СибАДИ, 2010.– 65 с.

27. Семеренко В. П. Візуальне програмування : навчальний посібник / В. П. Семеренко. – Вінниця : ВНТУ, 2010. – 113 с.

28. Уинкуп, С. MicrosoftSQLServer 6.5; СПб: БХВ - Москва, 2011.–896 с.

29. Фуфаев Э.В. Базы данных / Э.В.Фуфаев, Д.Э. Фуфаев. –(7-е изд.). – СПб: Академия. –2012. –320 с.

30. Хальворсон М. Microsoft Visual Basic 2005. – М.: ЭКОМ Паблишерз, 2007. – 640 с.

31. Хомоненко А.Д. Базы данных. Учебник для ВУЗов / А.Д. Хомоненко, В.М. Цыганков, М.Г. Мальцев. –Харьков: Корона-Принт. – 2012. – 260 с.

32. Шевякова Д.А., Степанов А.М., Карпов Р.Г. Самоучитель Visual Basic 2005. – СПб.: БХВ-Петербург, 2007. – 576 с.

33. Шкрыль А.А. Разработка клиент-серверных приложений в Delphi.− СПб.: БХВ-Петербург, 2006.− 480 с.

Додаток А
Пояснення та варіанти завдань до лабораторної роботи № 1

Таблиця А.1

№ атрибуту	Найменування атрибуту	Ім’я поля	Тип поля
1	Код підрозділу	Підрозділ	Числовий
2	Табличний номер	Номер	Числовий
3	Прізвище, ім’я, по батькові	Прізвище	Символьний
4	Код професії	Професія	Числовий
5	Вид оплати: Оклад, Годинна ставка Ставка	Оплата	Символьний
6	Ставка	Ставка	Числовий
7	Аванс	Аванс	Числовий
8	Стаж	Стаж	Числовий
9	Коефіцієнт премії	Премія	Числовий
10	Кількість лікарняних днів	Лікарняні	Числовий
11	Коефіцієнт вихідної допомоги	Допомога	Числовий
12	Кількість відроблених днів на 15-те число поточного місяця	Відроблені	Числовий
13	Дата народження	Дата	Дата
14	Примітка(домашня адреса)	Адреса	Символьний

Таблиця А.2

Варіант	№	Підрозділ	Номер	Прізвище	Професія	Оплата
1	2	3	4	5	6	7
1	1	101	1063	Кондратюк Ю. Б.	24195	1
	2	104	1064	Калишер А. Ф.	19111	1
	3	102	1065	Криворука Р. О.	15613	2
	4	106	1066	Кучер Л. Н.	18605	2
	5	108	1067	Ковальчук Т. Ю.	18605	2
	6	107	1068	Кваша Т. В.	18605	2
	7	105	1069	Кондратюк В. С.	19111	1
2	8	109	1070	Крапивняк С. Н.	18605	2
	9	101	1071	Захаров Д. В.	21035	1
	10	106	1072	Кузьмина Е. С.	18605	2
	11	104	1073	Комаха А. В.	18605	2
	12	102	1074	Кальченко А. А.	15613	2
	13	110	1075	Лапенко В. Н.	18605	2
	14	108	1076	Лунева Н. В.	18605	2
3	15	105	1077	Лепетан Н. А.	18605	2
	16	111	1078	Липова Е. В.	18605	2
	17	107	1079	Липинська Т. Л.	18605	2
	18	109	1080	Мельникова Г.П.	18605	2
	19	103	1081	МеткосА. Н.	16108	2
	20	101	1082	Макидон Л. Н.	22493	1
	21	112	1083	Магурова Т. П.	18605	2
4	22	101	1084	Майданюк З. К.	21999	1
	23	111	1085	Мусійчук Н. І.	18605	2
	24	109	1086	Макогорчук Г. В.	18605	2
	25	103	1087	Момот Є. І.	16187	2
	26	103	1088	Муляр Л. І.	16187	2
	27	113	1089	Остапчук Н. Г.	18605	2
	28	106	1090	Остапчук Л. В.	18605	2
5	29	110	1091	Омельчук А. В.	18605	2
	30	104	1092	Остапенко Т. М.	18605	2
	31	110	1093	Пінчук Н. І.	11676	2
	32	106	1094	Прокопчук Л. І.	19111	1
	33	114	1095	Півторак В. С.	18605	2
	34	111	1096	Пенькова С. С.	18605	2
	35	105	1097	Романенко Р.Ю.	18605	2
6	36	113	1098	Рябова Г. І.	18605	2
	37	102	1099	Садич Т. А.	15613	2
	38	112	1100	Степаник В. К.	18605	2
	39	108	1101	Санчук Л. І.	18605	2
	40	104	1102	Танкевич В. П.	18605	2
	41	102	1103	Татарчук Г. П.	20148	1
	42	110	1104	Тарасюк С. А.	18605	2

Продовження таблиці А.2

1	2	3	4	5	6	7
7	43	103	1105	Ткаченко В. Н.	16187	2
	44	114	1106	Федчишина В.Ф.	18605	2
	45	107	1107	Хажинська Н. І.	18605	2
	46	113	1108	Черватюк Є. Н.	18605	2
	47	103	1109	Частник Є. Р.	20148	1
	48	112	1110	Чернега Л. С.	18605	2
	49	107	1111	Шовга Т. К.	19111	1
8	50	107	1112	Шевчук А. І.	18605	2
	51	111	1113	Шавлюк Л. В.	11676	2
	52	106	1114	Щербатюк Є. Н.	18605	2
	53	114	1115	Юшко К. Т.	18605	2
	54	102	1116	Алексенко А. А.	15013	2
	55	105	1117	Більська Л. Г.	19605	2
	56	109	1118	Білокурська Т.В.	18605	2
9	57	111	1119	Бойко Н. Н.	18605	2
	58	108	1120	Борисова Л. Н.	19111	1
	59	112	1121	Войтюк В. Н.	11676	2
	60	112	1122	Василинюк І. В.	18605	2
	61	108	1123	Глодська Н. В.	18605	2
	62	109	1124	Гринюк Г. Г.	19111	1
	63	113	1125	Гудована Т. А.	18605	2
10	64	113	1126	Дурбак І. С.	11676	2
	65	114	1127	Дорошук А. С.	18605	2
	66	101	1128	Загороднюк Т.А.	20855	1
	67	105	1129	Хмеляр Т. В.	18605	2
	68	114	1130	Пристанчук Г. І.	11676	2
	69	110	1131	Луценко В. А.	18605	2
	70	104	1132	Янчук Р. О.	18605	2

Продовження таблиці А.2

Варіант	№	Ставка	Аванс	Стаж	Премія	Лікарняні	Допомога	Відроблені
1	2	8	9	10	11	12	13	14
1	1	240.00	100,00	10,1	0,50	5	0,4	13
	2	160.00	80,00	10,1	0,50	5	0,7	7
	3	0,75	50,00	10,1	0,50	5	0,4	4
	4	0,80	60,00	10,1	0,50	5	0,5	10
	5	0,80	65,00	10,1	0,50	5	0,8	13
	6	0,83	65,00	10,1	0,50	5	0,6	12
	7	165	70,00	10,1	0,50	5	0,5	13
2	8	0,85	60,00	8,3	0,40	10	0,6	11
	9	130	65,00	8,3	0,40	10	0,4	13
	10	0,80	55,00	8,3	0,40	10	0,8	13
	11	0,83	60,00	8,3	0,40	10	0,6	12
	12	0,75	55,00	8,3	0,40	10	0,6	13
	13	0,83	55,00	8,3	0,40	10	0,7	1
	14	0,85	65,00	8,3	0,40	10	0,8	12
3	15	0,80	50,00	6,2	0,30	15	0,5	12
	16	0,80	50,00	6,2	0,30	15	0,5	10
	17	0,80	50,00	6,2	0,30	15	0,4	3
	18	0,83	55,00	6,2	0,30	15	0,9	12
	19	0,85	60,00	6,2	0,30	15	0,7	11
	20	170	90,00	6,2	0,30	15	0,8	9
	21	0,83	50,00	6,2	0,30	15	0,6	5
4	22	165	80,00	9,0	0,50	5	0,7	12
	23	0,80	50,00	9,0	0,50	5	0,7	11
	24	1,80	80,00	9,0	0,50	5	0,7	12
	25	0,76	60,00	9,0	0,50	5	0,6	12
	26	1,00	80,00	9,0	0,50	5	0,4	4
	27	0,80	50,00	9,0	0,50	5	0,5	4
	28	0,83	50,00	9,0	0,50	5	0,6	12
5	29	0,83	50,00	7,5	0,40	10	0,6	12
	30	0,85	60,00	7,5	0,40	10	0,6	11
	31	0,70	45,00	7,5	0,40	10	0,4	5
	32	165	80,00	7,5	0,40	10	0,5	12
	33	0,85	65,00	7,5	0,40	10	0,6	11
	34	0,80	55,00	7,5	0,40	10	0,7	12
	35	0,83	55,00	7,5	0,40	10	0,8	11
6	36	0,80	50,00	8,5	0,30	15	0,9	1
	37	0,76	45,00	8,5	0,30	15	0,8	12
	38	0,85	60,00	8,5	0,30	15	0,7	12
	39	0,83	0	8,5	0,30	15	0,6	12
	40	0,83	50,00	8,5	0,30	15	0,5	11
	41	160,00	75,00	8,5	0,30	15	0,6	12
	42	0,83	0	8,5	0,30	15	0,6	7

Продовження таблиці А.2

1	2	8	9	10	11	12	13	14
7	43	0,75	50,00	4,3	0,50	5	0,7	8
	44	0,95	65,00	4,3	0,50	5	0,5	12
	45	0,80	50,00	4,3	0,50	5	0,4	12
	46	0,85	0	4,3	0,50	5	0,8	12
	47	160,00	75,00	4,3	0,50	5	0,5	11
	48	0,83	55,00	4,3	0,50	5	0,7	7
	49	170,00	80,00	4,3	0,50	5	0,8	7
8	50	0,80	60,00	5,7	0,40	10	0,5	12
	51	0,75	55,00	5,7	0,40	10	0,4	11
	52	0,83	50,00	5,7	0,40	10	0,6	12
	53	0,80	50,00	5,7	0,40	10	0,5	9
	54	0,75	45,00	5,7	0,40	10	0,4	11
	55	0,85	60,00	5,7	0,40	10	0,5	12
	56	0,80	65,00	5,7	0,40	10	0,6	13
9	57	0,85	85,00	7,0	0,30	15	0,6	12
	58	170,00	80,00	7,0	0,30	15	0,5	11
	59	0,75	50,00	7,0	0,30	15	0,4	10
	60	0,83	0	7,0	0,30	15	0,7	9
	61	0,80	0	7,0	0,30	15	0,6	9
	62	165,00	80,00	7,0	0,30	15	0,6	8
	63	0,85	60,00	7,0	0,30	15	0,7	7
10	64	0,80	55,00	9,5	0,50	5	0,7	11
	65	0,83	60,00	9,5	0,50	5	0,4	12
	66	180,00	90,00	9,5	0,50	5	0,5	12
	67	0,80	0	9,5	0,50	5	0,5	7
	68	0,75	50,00	9,5	0,50	5	0,7	7
	69	0,85	65,00	9,5	0,50	5	0,5	7
	70	0,85	55,00	9,5	0,50	5	0,7	12

Додаток Б
Пояснення та варіантизавдань до лабораторноїроботи № 2

Таблиця Б.1

Професія	Назва професії	Середня оплата
24195	Начальник цеху	240,00
22493	Інженер-технолог	170,00
21035	Бухгалтер	130,00
21999	Електромеханік	165,00
20148	Бригадир	160,00
20855	Старший майстер	180,00
19111	Майстер	160,00
15613	Контролер	140,00
16187	Слюсар-ремонтник	155,00
16180	Слюсар КВПа	150,00
18605	Швачка	145,00
11676	Гладильщик	135,00
0	Військпред	200,00

Таблиця Б.2

Варіант	Запит типу 1	Запит типу 2	Запит типу 3	Запит типу 4	Запит типу 5
	Умова відбору	Вид сортування	Формула для обчислень	Підсумковий запит	Запит на зміну
1	<x	Зростання	1	Min	Обновлення
2	= x	Спадання	1	Max	Вилучення
3	>x	Зростання	2	Avg	Обновлення
4	= x	Спадання	2	Min	Додавання
5	>x	Зростання	3	Count	Вилучення
6	<x	Спадання	3	Avg	Додавання
7	= x	Зростання	4	Max	Вилучення
8	<x	Спадання	4	Avg	Обновлення
9	= x	Зростання	5	Min	Додавання
10	>x	Спадання	5	Count	Обновлення
11	<x	Зростання	6	Max	Додавання
12	>x	Спадання	6	Count	Вилучення

Додаток В
Приклад до лабораторноїроботи № 3

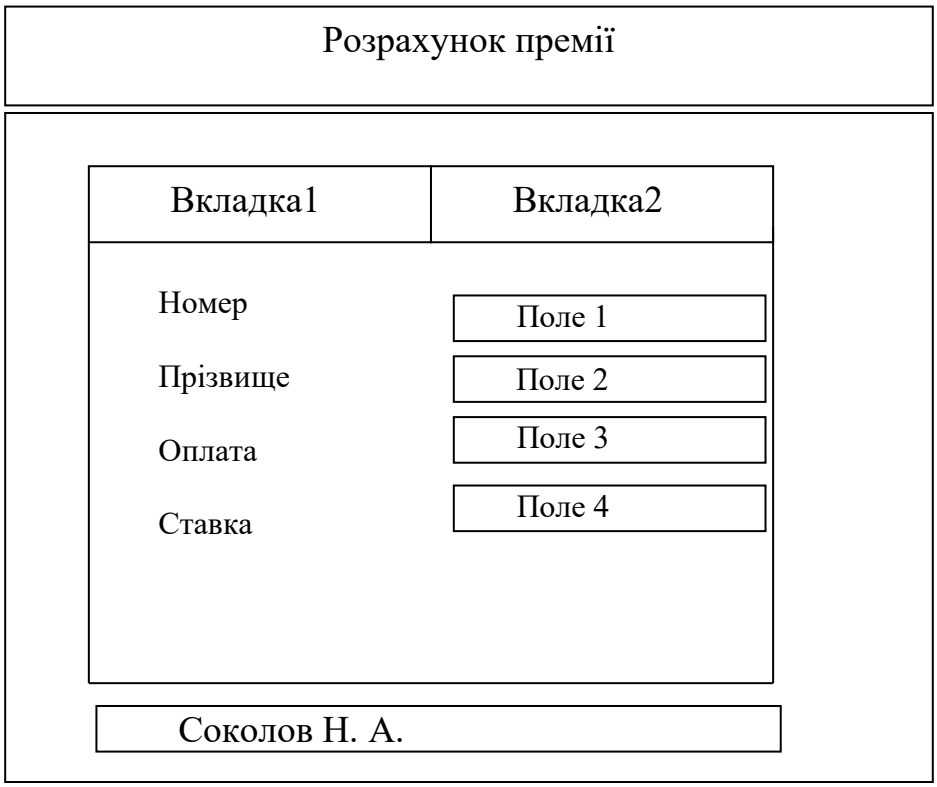

Рисунок В.1

Додаток Г
Варіантизавдань до лабораторноїроботи № 6

Варіант 1. БД«Облік виданих подарунків неповнолітнім дітям співробітників»

Код співробітника	Код співробітника	Код дитини
Прізвище	Ім’я дитини	Вартість подарунка
Ім’я	Дата народження	Дата видачі подарунка
Побатькові	Код дитини	Код видачі
Посада		
Підрозділ		
Дата прийняття на роботу		

Варіант 2. БД«Облік виконаних ремонтних робіт»

Код приладу в ремонті	Код приладу	Код майстра
Назва приладу	Код майстра	Прізвище майстра
Тип приладу	ПІБ власника приладу	Ім’я майстра
Дата виробництва	Дата прийому в ремонт	Побатькові майстра
	Вид поломки	Розряд майстра
	Вартість ремонту	Дата прийняття на роботу
	Код ремонту	

Варіант 3. БД «Продаж квітів»

Код квітів	Код квітів	Код продавця
Назва квітів	Дата продажу	Прізвище
Сорт квітів	Ціна	Ім’я
Середня висота	Код продавця	Побатькові
Тип листка	Код продажу	Розряд
Додаткові відомості		Оклад
		Дата прийняття на роботу

Варіант 4. БД «Надходження лікарських засобів»

Код ліків	Код ліків	Код постачальника
Назва ліків	Код постачальника	Скорочена назва
Показання до застосування	Дата постачання	Повна назва
Одиниця виміру	Ціна за одиницю	Юридична адреса
Кількість в упаковці	Кількість	Телефон
Назва виробника	Код надходження	ПІБ керівника

Варіант 5. БД «Списання обладнання»

Код обладнання	Код обладнання	Код співробітника
Назва обладнання	Причина списання	Прізвище
Тип обладнання	Дата списання	Ім’я
Дата надходження	Код співробітника	Побатькові
ПІБ відповідального	Код списання	Посада
Місце встановлення		Підрозділ
		Дата прийняття на роботу

Варіант 6. БД «Куховарська книга»

Код блюда	Код блюда	Код продукту
Тип блюда	Код продукту	Назва продукту
Маса блюда	Об’єм продукту	Одиниця виміру
Порядок приготування		
Кількість калорій		
Кількість вуглеводів		

Варіант 7. БД «Реєстрація вхідної документації»

Код реєстратора	Код документа	Код організації-відправника
Прізвище	Номер документа	Скорочена назва
Ім’я	Дата реєстрації	Повна назва
Побатькові	Короткий зміст документа	Юридична адреса
Посада	Тип документа	Телефон
Дата прийняття на роботу	Код організації-відправника	ПІБ керівника
	Код реєстратора	

Варіант 8. БД «Звільнення співробітника»

Код співробітника	Код документа	Код статті звільнення
Прізвище	Номер документа	Назва статті звільнення
Ім’я	Дата реєстрації	Причина звільнення
Побатькові	Дата звільнення	Номер статті звільнення
Посада	Код статті звільнення	Номер пункту / підпункту звільнення
Підрозділ	Код співробітника	
Дата прийняття на роботу	Грошова компенсація	

Варіант 9. БД «Звільнення співробітника»

Код співробітника	Код документа	Код відпустки
Прізвище	Номер документа	Тип відпустки
Ім’я	Дата реєстрації	Оплата відпустки
Побатькові	Дата початку відпустки	Пільги по відпустці
Посада	Дата кінця відпустки	
Підрозділ	Код співробітника	
Дата прийняття на роботу	Код відпустки	

Варіант 10. БД «Реєстрація вихідної документації»

Код відправника	Код документа	Код організації-одержувача
Прізвище	Номер документа	Скорочена назва
Ім’я	Дата реєстрації	Повна назва
Побатькові	Короткий зміст документа	Юридична адреса
Посада	Тип документа	Телефон
Дата прийняття на роботу	Код організації-одержувача	ПІБ керівника
	Код відправника	

Варіант 11. БД «Призначення на посаду»

Код співробітника	Код документа	Код посади
Прізвище	Номер документа	Назва посади
Ім’я	Дата реєстрації	Пільги за посадою
Побатькові	Дата призначення	Вимоги до кваліфікації
Дата прийняття на роботу	Код співробітника	
Дата народження	Код посади	
Стать		

Варіант 12. БД «Видача обладнання на прокат»

Код клієнта	Код видачі	Код обладнання
Прізвище	Номер документа	Назва обладнання
Ім’я	Дата початку прокату	Тип обладнання
Побатькові	Дата закінчення прокату	Дата надходження в прокат
Адреса	Код обладнання	
Телефон	Код клієнта	
Серія і номер паспорта	Вартість	

Варіант 13. БД «Списання обладнання з прокату»

Код обладнання	Код обладнання	Код співробітника
Назва обладнання	Причина списання	Прізвище
Тип обладнання	Дата списання	Ім’я
Дата надходження в прокат	Код співробітника	Побатькові
	Номер документа	Посада
	Дата реєстрації	Дата прийняття на роботу
	Код списання	

Варіант 14. БД «Прийом квітів в магазин»

Код квітів	Код квітів	Код постачальника
Назва квітів	Дата надходження	Скорочена назва
Сорт квітів	Ціна за одиницю	Повна назва
Середня висота	Код постачальника	Юридична адреса
Тип листка	Код надходження	Телефон
Додаткові відомості	Кількість	ПІБ керівника

Варіант 15. БД «Реєстрація клієнтів готелю»

Код номера	Код реєстрації	Код клієнта
Тип номера	Код номера	Прізвище
Перелік зручностей	Дата заїзду	Ім’я
Ціна за добу	Дата виїзду	Побатькові
	Вартість	Адреса
	Код клієнта	Телефон
		Серія і номер паспорта

Варіант 16. БД «Повернення обладнання в службу прокату»

Код клієнта	Код повернення	Код обладнання
Прізвище	Номер документа	Назва обладнання
Ім’я	Дата повернення	Тип обладнання
Побаткові	Стан обладнання	Дата надходження в прокат
Адреса	Код обладнання	
Телефон	Код клієнта	
Серія і номер паспорта	Штраф	

Варіант 17. БД «Облік матеріальних цінностей на підприємстві»

Код цінності	Код встановлення на облік	Код матеріально-відповідального
Назва цінності	Код цінності	Прізвище
Тип цінності	Код матеріально-відповідального	Ім’я
Закупівельна вартість	Дата встановлення на облік	Побаткові
Термін гарантії	Місце знаходження цінності	Посада
Дата початку гарантії		Дата прийняття на роботу
		Підрозділ

Варіант 18. БД «Склад ремонтних робіт»

Код ремонтної роботи	Код ремонтної роботи	Код майстра
Код етапу роботи	Код майстра	Прізвище
Назва етапу роботи	Вартість ремонту	Ім’я
Вартість етапу	Кількість днів ремонту	Побаткові
	Назва ремонтної роботи	Розряд
		Дата прийняття на роботу

Варіант 19. БД «Продаж ліків»

Код ліків	Номер чека	Номер чека
Назва ліків	Ціна за одиницю	Дата продажу
Показання до застосування	Кількість	Сума
Одиниця виміру	Код ліків	ПІБ касира
Кількість в упаковці	Код запису в чеках	
Назва виробника		

Варіант 20. БД «Облік виконання по вхідній документації»

Код виконавця	Код документа	Код документа
Прізвище	Дата призначення на виконання	Номер документа
Ім’я	Строк виконання в днях	Дата реєстрації
Побаткові	Тип результату	Короткий зміст документу
Посада	Код виконавця	Тип документу
Підрозділ	Факт виконання	Організація-відправник
Дата прийняття на роботу		Код виконавця

Додаток Д
Варіантизавдань до лабораторноїроботи № 7

Таблиця Д.1

Варіант	Перелік завдань												
1	1	6	11	16	21	26	31	36	41	46	51	56	61
2	2	7	12	17	22	27	32	37	42	47	52	57	62
3	3	8	13	18	23	28	33	38	43	48	53	58	63
4	4	9	14	19	24	29	34	39	44	49	54	59	64
5	5	10	15	20	25	30	35	40	45	50	55	60	65
6	6	11	16	21	26	31	36	41	46	51	56	61	1
7	7	12	17	22	27	32	37	42	47	52	57	62	2
8	8	13	18	23	28	33	38	43	48	53	58	63	3
9	9	14	19	24	29	34	39	44	49	54	59	64	4
10	10	15	20	25	30	35	40	45	50	55	60	65	5
11	2	6	12	16	22	26	32	36	42	46	52	56	62
12	1	5	11	15	21	25	31	35	41	45	51	55	61
13	3	7	13	17	23	27	33	37	43	47	53	57	63
14	4	8	14	18	24	28	34	38	44	48	54	58	64
15	5	9	15	19	25	29	35	39	45	49	55	59	65
16	6	13	26	33	36	43	46	53	56	63	1	11	17
17	7	18	27	28	37	38	47	48	57	58	2	4	10
18	8	9	18	19	28	29	38	39	48	49	58	59	60
19	9	14	29	24	39	34	49	44	59	54	61	65	3
20	10	12	20	22	30	32	40	42	50	52	60	62	8