Лабораторна робота №1
Тема: Створення бази даних.

Теоретичні відомості:

В даних методичних вказівках розглядаються ключові моменти використання структурованої мови запитів **SQL** (*Structured Query Language*), яка надає засоби створення і обробки даних в реляційних базах даних і є основною базовою мовою в різних СУБД.

Команди мови **SQL** можна поділити на три категорії:

DDL (*Data Definition Language*)– мова визначення даних – складається з команд, які створюють об'єкти (таблиці, індекси, представлення і так далі) у базі даних.

DML (*Data Manipulation Language*) – мова маніпулювання даними – це набір команд, що визначають, які значення представлені в об'єктах бази даних у будь-який момент часу.

DCL (*Data Control Language*) – Мова Керування Даними – складається із засобів, які визначають чи дозволити користувачеві виконувати певні чи дії ні.

При наведенні правопису команд прийняті такі узгодження:

- службові слова виділені жирним шрифтом і написані великими літерами (**CREATE)**;
- слова, виділені курсивом і написані малими літерами (*пароль*), є ідентифікаторами, заданими користувачем;
- параметри в квадратних дужках ([ ])можуть не задаватись;
- вертикальна лінія (|) визначає варіанти використання;
- фігурні дужки визначають обов'язкове включення однієї з конструкцій ({ })

Інсталяція програми *Firebird 1.5*

Firebird 1.5 є клоном *Interbase*, програми, яка призначена для роботи з базами даних. Інсталяція програми *Firebird 1.5* по замовчуванню відбувається у папку **C:\Program Files\Firebird\Firebird_1_5**. Під час інсталяції необхідно вибрати архітектуру **Super Server**, вказати необхідні налаштування: запуск захисника (*Guardian*), спосіб роботи сервера як додатку (*Application*) чи як сервісної служби (*Service*); автоматичний старт при завантаженні комп'ютера, інсталяцію менеджера сервера (*Install Server Control Applet*) на панель управління, копіювання необхідних бібліотек. Менеджер сервера використовується для зміни названих вище опцій і для запуску сервера. В іншому випадку для запуску сервера необхідно завантажити файл **fbserver.exe** з папки **BIN**, яка розташована у папці *Firebird_1_5*.

Як клієнтська система, використовується програма *IBExpert*, ярлик якої

знаходиться на робочому столі IBExpert .

Створення бази даних

Для створення бази даних з використанням утиліти *isql* (*isql* означає *interactive SQL* – інтерактивний SQL) в інтерактивному режимі, в командному

рядку необхідно перейти в директорію C:\Program Files\ Firebird\ Firebird_1_5\ BIN і набрати ***ISQL***.

В *isql* кожна команда закінчується крапкою з комою (;). Команду можна розбивати на декілька рядків, натискуючи клавішу <*Enter*>.

Сукупність дій відносно бази даних, в результаті яких дотримується цілісність бази даних, називається **транзакцією**.

Для підтвердження транзакції використовується команда **COMMIT;**. Якщо транзакція завершується успішно, то усі зміни фіксуються в базі даних.

Якщо сукупність дій, направлених на базу даних, порушують її цілісність, то відповідно їх потрібно відмінити за допомогою команди **ROLLBACK;**.

Для завершення роботи необхідно в командному рядку набрати команду **QUIT**;

1. Створення бази даних

Розглянемо спрощену конструкцію оператора **CREATE DATABASE**:

CREATE {**DATABASE** | **SCHEMA**} *'ім'я_БД'*
[**USER** *'ім'я_користувача'* [**PASSWORD** *'пароль'*]]
[**PAGE_SIZE** [=] *число*]
[**DEFAULT CHARACTER SET** *код*];

Слово **SHEMA** є синонімом слова **DATABASE**.

'ім'я_БД' – задає шлях і ім'я бази даних. Якщо в назві присутні пропуски, то використовуються подвійні лапки.

USER *'ім'я_користувача'* – задає ім'я користувача. Типовим ім'ям є **SYSDBA**.

PASSWORD *'пароль'* – задає пароль. Типовим є пароль **masterkey**.

В команді **CREATE DATABASE** лапки навколо шляху до файлу, імені користувача і пароля є обов'язковими.

PAGE_SIZE [=] *число* – задає розмір сторінки бази даних. Це може бути 1024, 2048, 4096, 8192, 16384. По замовчуванню використовується 1024. Бажано, щоб розмір сторінки співпадав з розміром кластера жорсткого диска.

DEFAULT CHARACTER SET *код* – встановлення кодування, яке використовується по замовчуванню. Для Східної Європи необхідно задавати **WIN1251**. Якщо даний параметр не задано, то кодування встановлюється в **NONE**.

Приклад:

CREATE DATABASE 'D:\Students\univer.gdb'
PAGE_SIZE 8192
USER 'SYSDBA' **PASSWORD** 'masterkey'
DEFAULT CHARACTER SET WIN1251;

2. Знищення активної бази даних

DROP DATABASE;

3. Приєднання до бази даних

CONNECT *'ім'я_БД'* **[USER** *'ім'я_користувача'***] [PASSWORD** *'пароль'***];**

Приклад:

CONNECT 'D:\Students\univer.fdb'

USER ’SYSDBA’ **PASSWORD** ’masterkey’;

4. Від’єдання від бази даних

DISCONNECT {*ім’я_БД***|ALL | DEFAULT}** ;

ALL або **DEFAULT** від’єднує усі відкриті бази даних**.**

5. Створення таблиць

Розглянемо спрощений синтаксис команди

CREATE TABLE *ім’я_таблиці*
(*ім’я_поля тип_даних* **[(***розмір_поля* **[**, *точність***])] [***обмеження***],**
ім’я_поля тип_даних **[(***розмір_поля* **[**, *точність***])] [***обмеження***],…,**
[PRIMARY KEY (*ім’я_поля***)],**
[UNIQUE (*ім’я_поля***)],**
[FOREIGN KEY (*ім’я_поля***) REFERENCES** *ім’я_таблиці1*
(*ім’я_поля_первинного_ключа_таблиці1***)**
[ON DELETE {NO ACTION|CASCADE|SET DEFAULT|SET NULL}]
[ON UPDATE {NO ACTION|CASCADE|SET DEFAULT|SET NULL}]],
[CHECK (*умова_обмежень***)]**
);

При створенні таблиці задається перелік полів (стовпців).

тип_даних – задається зарезервований тип даних або назва домену або розрахункове поле (**COMPUTED BY** *вираз*).

обмеження:

- задання за замовчуванням певного значення або **NULL**-значення або для текстових полів імені користувача бази даних
  DEFAULT *значення* **| NULL | USER**
- умову нерівності нулю (**NOT NULL**);
- обмеження **CHECK**
  [CONSTRAINT *ім’я_обмеження***] CHECK** *умова***,**
- визначення первинного ключа
  [CONSTRAINT *ім’я_обмеження***] PRIMARY KEY,**
- визначення унікального ключа
  [CONSTRAINT *ім’я_обмеження***] UNIQUE,**
- визначення зовнішнього ключа і зв’язок з іншою таблицею за визначеним полем. При зв’язку з іншою таблицею задаються умови цілісності для знищення записів **ON DELETE** або зміни ключового поля **ON UPDATE.** За замовчуванням підтримується опція **NO ACTION.**

  [CONSTRAINT *ім’я_обмеження***] FOREIGN KEY (***ім’я_поля)* **REFERENCES** *ім’я_таблиці1* **(***ім’я_поля_первин._ключа_таблиці1***)**
  [ON DELETE {NO ACTION|CASCADE|SET DEFAULT|SET NULL}]
  [ON UPDATE {NO ACTION|CASCADE|SET DEFAULT|SET NULL}]];

  Варіанти дій:

  NO ACTION (за замовчуванням) – неможливо змінити або видалити значення в головній таблиці (первинний ключ), поки існують відповідні значення в підпорядкованій (зовнішній ключ);

CASCADE – при зміні значень ключового поля (первинного ключа) в головній таблиці змінюються відповідно значення в зв'язаному стовпці підпорядкованої таблиці (зовнішній ключ). При видаленні записів з головній таблиці видаляються відповідні записи із підпорядкованої таблиці;

SET NULL – при зміні значень первинного ключа головної таблиці чи видаленні записів з головної таблиці значення стовпця, який є зовнішнім ключем у підпорядкованій таблиці, встановлюються в NULL;

SET DEFAULT – значення стовпця, який є зовнішнім ключем у підпорядкованій таблиці, встановлюються в значення, яке в списку значень первинного ключа головної таблиці записано за замовчуванням;

Приклад.

Створюється таблиця T1 з первинним ключем P1. Створюється таблиця T2 з зовнішнім ключем F2, яка зв'язується з таблицею T1 з первинним ключем P1. Зміни значень первинного ключа P1 таблиці T1 автоматично приводять до зміни ключа F2 таблиці T2. Видалення записів таблиці T1 приводить до встановлення NULL-значень зовнішнього ключа відповідних записів таблиці T2.

```
CREATE TABLE T1 (P1 INTEGER NOT NULL PRIMARY KEY);

CREATE TABLE T2
(F2 INTEGER REFERENCES T1(P1)
ON UPDATE CASCADE
ON DELETE SET NULL);
```

При визначенні первинного або унікального ключів таблиці обов'язково потрібно задавати обмеження **NOT NULL**. Ім'я обмежень бажано задавати, оскільки вони використовуються при пошуку помилок і знищенні обмежень. Якщо ім'я обмежень не задано, то вони генеруються системою.

Визначення ключів або обмеження **CHECK**, якщо вони включають декілька полів, вказуються після визначення усіх полів таблиці. Перед обмеженнями варто задавати параметр **[CONSTRAINT** *ім'я_обмеження***]** для визначення імені обмежень.

Типи даних :

Числові:

SMALLINT – цілі числа від -128 до +127

INTEGER – цілі числа від -32768 до +32767

BIGINT – цілі числа від -2 147 483 648 до +2 147 483 647

FLOAT – дійсні числа від $3.4*10^{-38}$ до $3.4*10^{38}$ з 7 значущими цифрами п

DOUBLE PRECISION – дійсні числа від $1.7*10^{-308}$ до $1.7*10^{308}$ з 15 значущими цифрами

NUMERIC[(*розмірність* [, *точність*])] – фіксований формат.

розмірність – загальне число знаків (максимальне 18 знаків);

точність – число знаків після коми.

DECIMAL[(*розмір* [, *точність*])] – фіксований формат.

Дата, час:

DATE – поле дати, задається у форматі dd.mm.yyyy

TIME – час, задається у форматі hh:mm:ss

TIMESTAMP – дата і час, задається у форматі dd.mm.yyyy hh:mm

Текстові:

CHAR(*розмір***) [CHARACTER SET** *код***] [(COLLATE** *код1***)]** – рядок символів фіксованої довжини, що містить будь-які друковані символи розмірністю від 0 до 32767.

VARCHAR(*розмір*) **[CHARACTER SET** *код***] [(COLLATE** *код1***)]** – рядок символів змінної довжини, що містить будь-які друковані символи розмірністю від 0 до 32767.

BLOB [SUB_TYPE *число***] [SEGMENT SIZE [***розмір***]]** – поле, що містить дані великого об'єму такі як графіка, текст, цифровий звук, у двійковому вигляді. Якщо *число* дорівнює **1**, то це текст. Замість числа **1** можна вказувати константу **TEXT**.

Опис типу даних для стовпця типу **CHAR, VARCHAR** або **BLOB-text** може включати пропозицію **CHARACTER SET** визначаючи специфічне кодування для вибраного стовпця. Інакше стовпець використовує визначену за замовчуванням для бази даних кодування. Якщо кодування бази даних змінене, всі стовпці згодом визначені мають нове кодування, але існуючі стовпці не змінюються.

За допомогою опції **COLLATE** вказується вибраний порядок сортування. Для кодування **WIN1251** допустимі порядки сортування **WIN1251, WIN1251_UA, PXW_CYRL. PXW_CYRL** встановлює порядок сортування для баз даних PARADOX.

Приклади обмежень для числових полів

CHECK (*ім'я_поля* **[NOT] BETWEEN 0 AND 6)**

CHECK (*ім'я_поля* > 10000 **AND** *ім'я_поля* <= 2000000**)**

Приклади обмежень для текстових полів

CHECK (*ім'я_поля* **[NOT] IN ('software', 'hardware', 'other', 'N/A'))**

CHECK (*ім'я_поля* **[NOT] LIKE '*ware*')**

CHECK (*ім'я_поля* **=10 OR (***ім'я_поля* **> 20 AND** *ім'я_поля* **<= 100) OR** *ім'я_поля* **IS [NOT] NULL)**

CHECK (*ім'я_поля* **[NOT] CONTAINING** *значення***)**

CHECK (*ім'я_поля* **[NOT] STARTING WITH 'V')**

CHECK (*ім'я_поля* **= UPPER (***ім'я_поля***))**

Розрахункове поле

COMPUTED BY (STIP*1,2)

Приклад:

Створення таблиці PREDMET і зв'язок її з таблицею VYKLAD по полю VNOM.

CREATE TABLE PREDMET
(PNOM **INTEGER NOT NULL PRIMARY KEY**,
PNAME **VARCHAR**(15),
VNOM **INTEGER** ,
CONSTRAINT FK_PREDMET_1 **FOREIGN KEY** (VNOM)
REFERENCES VYKLAD (VNOM),
GOD **INTEGER CHECK**(GOD>0)
);

6. Знищення таблиці

DROP TABLE *ім'я_таблиці***;**

Приклад:

Знищення таблиці VYKLAD.

DROP TABLE VYKLAD;

7. Створення домену

Домен – це набір усіх допустимих значень, які може приймати певне поле таблиці. Після створення домен може використовуватись при визначенні стовпців таблиць як додатковий тип даних.

CREATE DOMAIN *назва_домену* **[AS]** *тип_даних*
[DEFAULT { значення | NULL | USER}]
[NOT NULL] [CHECK (<*умова обмежень*>**)]**
[CHARACTER SET *код* **[COLLATE** *сортування***]];**

Для типу даних команда створення домену може включати:

- значення по замовчуванню **DEFAULT**;
- умову нерівності нулю **NOT NULL**;
- обмеження **CHECK.** При заданні умов замість імені поля вказується службове слово **VALUE;**
- кодування **CHARACTER SET** і порядок сортування **COLLATE** для типу **CHAR, VARCHAR** або **BLOB-текст**.

Приклад:

Створення домену GOD, який може приймати цілі значення , більші від 10, і по замовчуванню дорівнює 50.

CREATE DOMAIN GOD
AS INTEGER
DEFAULT 50
CHECK (**VALUE** > 10);

Створення домену DESCRIPT з типом BLOB-текст і кодуванням

CREATE DOMAIN DESCRIPT
AS BLOB SUB_TYPE TEXT **SEGMENT SIZE** 80
CHARACTER SET WIN1251;

8. Знищення домену

DROP DOMAIN *назва_домену*;

Домен можна знищити в тому випадку, якщо він не використовується в таблицях.

Завдання до виконання:

1) В командному рядку перейдіть у папку **C:\Program Files\Firebird\Firebird_1_5\bin** і завантажте утиліту **isql.**
2) Створіть на диску **D:** у папці **Students** базу даних **Univer.gdb** для користувача **SYSDBA** з паролем **masterkey**. Розмір сторінки задайте 4096, кодування WIN1251.
3) Під'єднайтесь до бази даних.
4) Створити домен **NAM** для текстових даних, що містять не більше 20 символів, задавши кодування і порядок сортування **WIN1251.**
5) Створіть таблицю **VYKLAD** з полями

Таблиця **VUKLAD**

VNOM	VFAM	VIMA	VOTCH	KAF	POSADA	OKLAD
10001	Бачишина	Лариса	Дмитрівна	ПМ	Ст. викладач	710,40

VNOM – табельний номер викладача, є цілим, не допускає нульових значень, вибирається як первинний ключ таблиці. **VFAM** – прізвище, **VIMA** – ім'я, **VOTCH** – по-батькові викладача. Для прізвища, імені, по-батькові задайте тип даних у вигляді домену **NAM. KAF** – місце роботи. **POSADA** – посада викладача, **OKLAD** – оклад викладача. Для окладу викладача виберіть тип даних **NUMERIC** або **DECIMAL**, вказавши 2 знаки після коми.

6) Створіть таблицю **PREDMET** з полями

Таблиця **PREDMET**

PNOM	PNAME	VNOM	GOD	SEMESTR
301	Бази даних	101	100	5

PNOM – номер предмета в навчальному плані є цілим і є первинним ключем**. PNAME** – назва предмета. **GOD** – кількість годин по даному предмету. **SEMESTR** – семестр, в якому читається предмет. При створенні таблиці **PREDMET** задайте зв'язок з таблицею **VYKLAD** по полю **VNOM**. Задайте умови цілісності: при знищенні – **SET NULL**, при зміні – **UPDATE.**

7) Від'єднайтесь від бази даних і вийдіть з режиму *isql***.**
8) За допомогою ярлика на робочому столі завантажте клієнт-додаток *IBExpert*.
9) За допомогою команди **Database ⇒ Register Database** зареєструйте створену базу даних.
10) За допомогою команди **Database ⇒ Connect to Database** відкрийте базу даних **UNIVER**.
11) За допомогою команди **Database ⇒ New Table** створіть таблицю **STUDENTS** з полями

Таблиця **STUDENTS**

SNOM	SFAM	SIMA	SOTCH	STIP	GRUP	FORM	FOTO
200101	Іванов	Сергій	Петрович	75,00	ПМ-32	платна	

SNOM – номер залікової книжки студента є первинним ключем. Поля **SFAM** – прізвище, **SIMA** – ім'я, **SOTCH** – по-батькові студента, Для прізвища, імені, по-батькові задайте тип даних у вигляді домену **NAM. GRUP** – група, в якій навчається студент. Для поля **GRUP** задайте значенням по замовчуванню назву своєї групи Поле **STIP** – стипендія – має тип даних **NUMERIC** або **DECIMAL** з 2 знаками після коми. Для поля **FOTO** – фото студента – вибирається тип даних **BLOB**, **SUBTYPE**=BYNARY, **SIZE**=2048. Для поля **FORM** – форма навчання – задайте обмеження, що допускають ввід тільки двох значень 'платна' або 'бюджет'.

12) За допомогою команди **Database ⇒ New Table** створіть таблицю **USPISH** з полями

Таблиця **USPISH**

NOM	DATA	SNOM	PNOM	OCINKA
1	13.06.2005	200101	301	5

NOM є первинним ключем, для поля **DATA** – дата здачі іспиту – тип даних **DATE**. **OCINKA** – оцінка за іспит є цілим. Для поля **OCINKA** допускається ввід значень від 1 до 5 включно або NULL. По замовчуванню встановлюється значення **NULL** (не визначено).

13) На вкладці **Constraints** вікна **Table** задайте первинний і зовнішній ключі. Встановіть зв'язок таблиці **USPISH** з таблицею **STUDENTS** по полю **SNOM.** Задайте умови цілісності, що задовольняють умовам каскадування.

14) В зошит запишіть структури таблиць і схему даних.

15) Завершіть роботу.

Контрольні запитання:

1) Яка програма є сервером баз даних?
2) Яка програма є клієнтом баз даних?
3) Що таке транзакція?
4) Яка мова використовується для роботи в СУБД?
5) Як створити базу даних?
6) Де зберігаються дані?
7) Які типи даних ви знаєте?
8) Що означає NULL-значення?
9) Яким чином задаються обмеження для даних?
10) Як встановлюються зв'язки між таблицями?
11) Як реалізована цілісність даних в БД?