МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

МЕТОДИЧНІ ВКАЗІВКИ

до виконання лабораторної роботи

«ВИКОРИСТАННЯ ОПЕРАТОРА SELECT В СУБД MySQL»

з курсів «Бази даних та знань», «Бази даних та інформаційні системи»
для студентів спеціальностей «Інформатика», «Прикладна математика»

Затверджено
редакційно-видавничою
радою університету
протокол № 2 від 06.12.2012 р.

Харків
НТУ«ХПІ»
2013

Методичні вказівки до виконання лабораторної роботи «Використання оператора SELECT в СУБД MySQL» з курсів «Бази даних та знань», «Бази даних та інформаційні системи» для студентів спеціальностей «Інформатика», «Прикладна математика» / уклад. І. О. Багмут, О. С. Галас. – Х. : НТУ «ХПІ», 2013. – 28 с.

Укладачі: І. О. Багмут,
О. С. Галас

Рецензент Ю. А. Плаксій

Кафедра «Системи та процеси управління»

ВСТУП

Мета роботи – навчитися використовувати оператор SELECT мови SQL для конструювання однотабличних запитів до СУБД MySQL. Спочатку комп'ютер призначався для вирішення обчислювальних задач. З розвитком зовнішніх запам'ятовуючих пристроїв стало можливим зберігати й обробляти за допомогою комп'ютерів великі обсяги інформації. У наш час комп'ютер застосовується буквально в усіх сферах людської діяльності, де треба зберігати та оброблювати великі масиви даних: в економіці, промисловості, менеджменті, медицині, банківській сфері, управлінні та ін. Завдання, пов'язані зі зберіганням і обробкою інформації, прийнято називати інформаційними. Це системи резервування квитків, банківські системи, бухгалтерські та облікові системи, системи планування і управління та ін. Інформаційні системи мають справу з базами даних.

База даних (БД) – це спільний набір логічно пов'язаних даних (та опис цих даних), призначений для задоволення інформаційних потреб деякої організації. Для управління такими даними було розроблено спеціальне програмне забезпечення – систему управління базами даних.

Система управління базами даних (СУБД) – це комплекс програмних і мовних засобів, необхідних для створення баз даних, підтримання їх в актуальному стані та організація пошуку в них необхідної інформації.

За можливостями роботи в мережі СУБД можна розрізнити на так звані настільні СУБД, які не розраховані для роботи в мережі, і серверні (багатокористувацькі) СУБД.

Настільні СУБД призначені для роботи з невеликими централізованими базами даних. До них належать такі СУБД, як Access, FoxPro та інші.

Серверні СУБД призначені для роботи великої кількості користувачів в мережі з одними і тими ж даними. При цьому самі дані можуть бути розподілені за різними вузлами мережі. До серверів баз даних належать: MySQL, Oracle, MS SQL Server, Informix та інші.

За способом подання даних найбільш поширеними на сьогодні є СУБД, що працюють з реляційними базами даних. У реляційних БД дані і зв’язок між ними реалізовані у таблицях. Кожна таблиця такої БД подається як сукупність рядків і стовпців, де рядки відповідають екземпляру об’єкта, а стовпці (поля) – атрибутам (ознакам, характеристикам, параметрам) об’єкта, інформацію про який треба зберігати та обробляти. Всі перераховані вище СУБД є реляційними.

Клієнтські програми з робочих станцій звертаються до програми серверу за послугами з обробки даних. У реляційних СУБД мовою запитів до серверу є мова SQL (Structured QueryLanguage) – мова структурованих запитів.

Основним завданням даних методичних вказівок є розгляд можливостей та прикладів використання найважливішого оператора мови запитів СУБД MySQL – оператора SELECT.

1. ЗАГАЛЬНІ ВІДОМОСТІ ПРО СУБД MySQL

СУБД MySQL є клієнт-серверною системою, що включає багатопотоковий SQL-сервер та підтримує різні операційні системи, інструменти адміністрування і широкий діапазон програмних інтерфейсів.

Програмне забезпечення MySQL складається з безлічі програм, які можна віднести до однієї з трьох груп:

1) сервер MySQL і сценарії його запуску;

2) клієнтські програми для роботи з сервером;

3) програми, що працюють незалежно від сервера.

Сервер MySQL має ім’я mysqld в UNIX-подібних операційних системах або mysqld.exe (mysqld-nt.exe) в Windows.

Як основні клієнтські програми для роботи з сервером MySQL використовуються такі програми:

1) mysql – консольне прикладнення, що в режимі командного рядка дозволяє вводити команди SQL для передачі їх серверу.

2) mysqladmin – утиліта для виконання адміністративних функцій.

До утиліт, які можуть функціонувати без підключення до серверу MySQL, належать програми, що дозволяють виконувати різні операції над таблицями: стиснення, перевірку, оптимізацію, відновлення даних у разі пошкодження та інш.

При запуску утиліти за умовчанням як користувач вибирається root (адміністратор баз даних). Якщо при установці MySQL не був заданий пароль адміністратора, то при запуску утиліти достатньо натиснути на клавішу <Enter> для під’єднання до сервера. Після чого з’явиться запрошення MySQL до введення команд:

mysql>.

У загальному випадку запуск утиліти mysql виконується з командного рядка таким чином:

> mysql-u <User>-h <Host>-P <Port>-p <Password>.

де ключі позначають так:

u – ім’я користувача

h – ім’я хоста

P – номер порту

p – пароль.

Наприклад:

> mysql-u root-h localhost-P 3306-p12345

У рядку із запрошенням до введення можна вводити команди, що відправляються базі даних, і завершувати їх натисканням на клавішу <Enter>. Крім цього, є ряд команд, що інтерпретуються безпосередньо сервером MySQL. Щоб вивести список цих команд, необхідно ввести help або \ h після запрошення mysql>.

Нижче описані деякі різновиди запрошення до введення.

1) mysql> – очікування введення команди;

2) -> очікування введення наступного рядка команди;

3) '> очікування продовження рядка, який починається з відкритої одиничної лапки;

4) "> очікування продовження рядка, який починається з відкритої подвійної лапки.

Найбільш часто використовувані команди, які можна вводити в рядку запрошення MySQL, такі:

1) (exit) quit – завершення роботи утиліти;

2) use <імя_бази_данних> – перехід до використання зазначеної бази даних;

3) show databases – виведення списку доступних баз даних на сервері;

3) show tables – виведення списку доступних таблиць у рамках потокової бази даних;

4) describe <ім’я таблиці> – виведення описання стовпців таблиці;

5) status – виведення номера версії MySQL і відомостей про стан сервера;

6) source <ім’я файлу> – виконання команд зазначеного файлу як сценарію.

При використанні набору команд краще записати ці команди в текстовий файл з розширенням *.sql і "запустити" його за допомогою команди source.

Щоб повернутися до команд MySQL, що виконувалися раніше, досить просто скористатися клавішею «стрілка вгору».

Приклад 1.1. Зробимо сценарій (shop.sql), який реалізує створення БД електронного магазина з ім’ям Shop і однією таблицею Products.

"shop.sql"

```
CREATE DATABASE Shop;
USE Shop;
CREATE TABLE Products(
ID              int             NOT NULL PRIMARY KEY,
Category        varchar(15)     NOT NULL,
Product         varchar(30)     NOT NULL,
Number          int             NOT NULL,
Price           decimal(7,2)    NOT NULL);
```

Запустимо цей сценарій на виконання з допомогою команди source.

```
mysql> source c:\mysql5\data\shop.sql
```

Отримаємо:

```
Query OK, 1 row affected (0.00 sec)
Database changed
Query OK, 1 row affected (0.02 sec)
```

Як бачимо, після виконання команди сервер MySQL передає клієнту інформацію об успішності виконання команди, далі ця інформація виводиться клієнтом MySQL у консольне вікно.

Далі у багатьох випадках, щоб не ускладнювати виклад матеріалу надлишковими даними (наприклад, час виконання запиту) буде наводитися неповне виведення інформації об успішності виконання команд та операторів MySQL у консольне вікно.

2. ОПЕРАТОР SELECT

Оператор SELECT призначений для формування запиту до бази даних (БД), тобто для вибірки інформації з таблиць БД, що задовольняє заданим критеріям.

2.1. Синтаксис оператора SELECT

Базовий синтаксис оператора SELECT такий:

SELECT <список полів (стовпців)>

FROM <ім’я таблиці (таблиць)>

[WHERE <умова>]

[GROUP BY <ім’я стовпця або вирази з використанням колонки в списку вибору>]

[HAVING <обмежувальна умова на основі групи за результатами>]

[ORDER BY <список полів (стовпців)>

[ASC | DESC]];

Усі конструкції повинні йти в запиті саме в тому порядку, в якому вони перераховані вище. Квадратні дужки вказують, що дана складова оператора не є обов’язковою.

Розглянемо за частинами синтаксичне визначення оператора SELECT на прикладі вибірки даних з таблиці Products бази даних Shop (табл. 2.1). Дана таблиця зберігає інформацію про товари. що є в наявності.

Таблиця 2.1 – Products

ID	Category	Product	Number	Price
1	TV	LCD-TV PHILIPS 32PFL5606	10	1253.45
2	Monitor	LCD 16" Philips 166V3LSB	25	655.12
3	Videocamera	Canon Legria FS406	12	722.33
4	Photo	CANON EOS 550D	11	511.43
5	Monitor	LCD 20" Viewsonic VA2014w	5	455.14
6	Videocamera	JVC GZ-HM445	8	450.67
7	Photo	Nikon D5100	12	220.45
8	Videocamera	SONY HDR-CX560E	24	1554.22
	Photo	SONY Alpha A290	43	1453.18
10	TV	LCD-TV PHILIPS 37PFL6606T	52	899.99
11	Monitor	LCD 21.5" ASUS MS228H	11	544.00
12	Monitor	LCD 21.5" Philips 221TE2LB	85	545.32
13	TV	LCD-TV SAMSUNG UE40D6530WSXUA	56	990.56
14	Monitor	LCD 23.6" AOC E2450Swd	22	323.19
15	Photo	SONY Alpha A35	41	350.77
16	TV	LCD-TV SHARP LC-46LE630E	14	860.33
17	Monitor	LCD 23" AOC E2343F	33	585.67

У стовпцях таблиці Products зберігається така інформація:

– ID – номер товару;

– Category – категорія товару;
– Product – найменування товару;
– Number – кількість;
– Price – ціна за одиницю.

2.2. Конструкція FROM

Стовпці, задані безпосередньо після ключового слова SELECT, називаються списком вибірки. Імена стовпців у списку відокремлюються комами. Конструкція FROM <вихідна таблиця(і)> використовується для зазначення імені таблиці (таблиць), що є джерелом даних.

Приклад 2.1. Виберемо всі дані зі стовпців Product і Category.

```
SELECT Product, Category
FROM Products;
```

Отримаємо:

Product	Category
LCD-TV PHILIPS 32PFL5606	TV
LCD 16" Philips 166V3LSB	Monitor
Canon Legria FS406	Videocamera
CANON EOS 550D	Photo
LCD 20" Viewsonic VA2014w	Monitor
JVC GZ-HM445	Videocamera
Nikon D5100	Photo
SONY HDR-CX560E	Videocamera
SONY Alpha A290	Photo
LCD-TV PHILIPS 37PFL6606T	TV
LCD 21.5" ASUS MS228H	Monitor
LCD 21.5" Philips 221TE2LB	Monitor
LCD-TV SAMSUNG UE40D6530WSXUA	TV
LCD 23.6" AOC E2450Swd	Monitor
SONY Alpha A35	Photo
LCD-TV SHARP LC-46LE630E	TV
LCD 23" AOC E2343F	Monitor

Слід зазначити, що стовпці виведені в тому порядку, в якому вони присутні в запиті, а не в таблиці. Тепер виберемо дані з усіх стовпців таблиці. Це можна зробити з використанням символу * замість перерахування імен усіх стовпців у списку.

Приклад 2.2. Виберемо всі дані з таблиці Products.

```
SELECT *
FROM Products;
```

Отримаємо:

ID	Category	Product	Number	Price
1	TV	LCD-TV PHILIPS 32PFL5606	10	1253.45
2	Monitor	LCD 16" Philips 166V3LSB	25	655.12
3	Videocamera	Canon Legria FS406	12	722.33
4	Photo	CANON EOS 550D	11	511.43
5	Monitor	LCD 20" Viewsonic VA2014w	5	455.14
6	Videocamera	JVC GZ-HM445	8	450.67
7	Photo	Nikon D5100	12	220.45
8	Videocamera	SONY HDR-CX560E	24	1554.22
9	Photo	SONY Alpha A290	43	1453.18
10	TV	LCD-TV PHILIPS 37PFL6606T	52	899.99
11	Monitor	LCD 21.5" ASUS MS228H	11	544.00
12	Monitor	LCD 21.5" Philips 221TE2LB	85	545.32
13	TV	LCD-TV SAMSUNG UE40D6530WSXUA	56	990.56
14	Monitor	LCD 23.6" AOC E2450Swd	22	323.19
15	Photo	SONY Alpha A35	41	350.77
16	TV	LCD-TV SHARP LC-46LE630E	14	860.33
17	Monitor	LCD 23" AOC E2343F	33	585.67

Часто розробники прагнуть створювати якомога коротші запити і використовують символ *. Це приводить до зниження продуктивності прикладень баз даних і збільшення мережевого трафіку. Тому для досягнення високої швидкості обробки сервером запитів слід завжди дотримуватися такого правила – здійснювати вибірку тільки необхідної інформації, ні більше ні менше.

2.3. Конструкція WHERE

Конструкція WHERE <умова> дозволяє сформулювати умови, що визначають, яку саме інформацію має повернути запит. Найбільш часто застосовувані операції, які можуть використовуватися в конструкції WHERE для завдання умови вибірки, подані в табл. 2.2.

Таблиця 2.2 – Операції, що використовуються в конструкції WHERE

Операція	Приклади використання	Призначення
1	2	3
=, >, <, >=, <=, <>, !=, !>, !<	<Column Name> = <Other Column Name> <Column Name> = ‘Bob’	Стандартні операції порівняння. Ці операції діють в основному так само, як і в будь-якій іншій мові програмування. Проте слід підкреслити деякі відмітні особливості. 1) Результати виконання операцій порівняння "більше" (>), "менше" (<) і "дорівнює" можуть змінюватися залежно від обраної схеми упорядкування. Наприклад, якщо в базі данних вибрана схема упорядкування, нечутлива до регістра, то ‘ROM’ = ‘rom’, а якщо регістр враховується, то ‘ROM’<>‘rom’; 2) Операції != та <> означають "не дорівнює". 3) Операції !>, !< Означають "не більше" та "не менше" відповідно
AND, OR, NOT	<Column1> = <Value1> AND <Column2>= < Value2> <Column1> = ‘MyLiteral’ AND <Column2= ‘MyOtherLiteral’	Стандартні булеві логічні операції. Вони можуть використовуватися для об’єднання декількох умов в одній конструкції WHERE. У першу чергу, виконується операція NOT, потім AND, а після цього OR. Якщо потрібно змінити порядок виконання операцій, то можна ввести круглі дужки. Слід зазначити, що операція XOR в безпосередньому вигляді не підтримується
BETWEEN	<Column1>BETWEEN 1 AND 5	Виконання цієї операції порівняння приводить до отримання значення TRUE, якщо перший операнд менше або дорівнює другому і більше або дорівнює третьому. Це функціональний еквівалент виразу A >= B AND A <= C. Будь-яке з цих значень може бути іменами стовпців, змінними або літералами

Продовження таблиці 2.2

1	2	3
LIKE	<Column1>LIKE'ROM%'	Операція порівняння, що дозволяє використовувати символи "%" та "_" як символи узагальнення. Символ "%" вказує, що замість нього може бути підставлено строкове значення будь-якої довжини. Символ "_" вказує, що замість нього може бути підставлений будь-який символ
IN	<Column1>IN (List of Numbers) <Column1> IN ('A', 'b', '345')	Операція порівняння, яка повертає TRUE, якщо значення, що знаходиться ліворуч від ключового слова IN, узгоджується з одним із значень в списку, що знаходиться праворуч від ключового слова IN. Ця операція часто використовується в підзапитах

Приклад 2.3. Виведемо всю інформацію про телевізори.

```
SELECT *
FROM Products
WHERE Category = 'TV';
```

Отримаємо:

```
-------------------------------------------------------------------------------
ID    Category    Product                          Number      Price
-------------------------------------------------------------------------------
1     TV          LCD-TV PHILIPS 32PFL5606         10          1253.45
10    TV          LCD-TV PHILIPS 37PFL6606T        52          899.99
13    TV  LCD-TV SAMSUNG UE40D6530WSXUA  56          990.56
16    TV          LCD-TV SHARP LC-46LE630E         14          860.33
-------------------------------------------------------------------------------
```

Приклад 2.4. Отримаємо інформацію про товари, кількість яких лежить в діапазоні від 20 до 30.

```
SELECT *
FROM Products
WHERE Number> = 20 AND Number <= 30;
```

Отримаємо:

ID	Category	Product	Number	Price
2	Monitor	LCD 16" Philips 166V3LSB	25	655.12
8	Videocamera	SONY HDR-CX560E	24	1554.22
14	Monitor	LCD 23.6" AOC E2450Swd	22	323.19

Приклад 2.5. Вирішимо задачу прикладу 2.4, використовуючи операцію BETWEEN.

```
SELECT *
FROM Products
WHERE Number BETWEEN 20 AND 30;
```

Отримаємо:

ID	Category	Product	Number	Price
2	Monitor	LCD 16" Philips 166V3LSB	25	655.12
8	Videocamera	SONY HDR-CX560E	24	1554.22
14	Monitor	LCD 23.6" AOC E2450Swd	22	323.19

Приклад 2.6. Отримаємо інформацію про товари фірми Sony.

```
SELECT *
FROM Products
WHERE Product LIKE ‘% Sony%’;
```

Отримаємо:

ID	Category	Product	Number	Price
8	Videocamera	SONY HDR-CX560E	24	1554.22
9	Photo	SONY Alpha A290	43	1453.18
15	Photo	SONY Alpha A35	41	350.77

Приклад 2.7. Отримаємо інформацію про товари, найменування яких має довжину 15 символів.

```
SELECT *
FROM Products
WHERE Product LIKE '_____';
```

Отримаємо:

ID	Category	Product	Number	Price
8	Videocamera	SONY HDR-CX560E	24	1554.22
9	Photo	SONY Alpha A290	43	1453.18

Приклад 2.8. Отримаємо інформацію про товари, кількість яких дорівнює 5, 11, 22.

```
SELECT *
FROM Products
WHERE Number IN (5, 11, 22);
```

Отримаємо:

ID	Category	Product	Number	Price
4	Photo	CANON EOS 550D	11	511.43
5	Monitor	LCD 20" Viewsonic VA2014w	5	455.14
11	Monitor	LCD 21.5" ASUS MS228H	11	544.00
14	Monitor	LCD 23.6" AOC E2450Swd	22	323.19

2.4. Конструкція ORDER BY

Зазвичай вибірка даних відбувається з урахуванням або фізичної послідовності даних у таблиці, або з урахуванням структури індексів, які використовуються для пошуку даних.

Конструкція ORDER BY <список полів (стовпців)> [ASC | DESC] використовується для сортування результатів запиту на ім'я заданого поля (полів) таблиці. За замовчуванням сортування виконується за зростанням. Якщо необхідно явно задати порядок сортування, треба використовувати ключові слова ASC і DESC, ASC – сортування за зростанням, DESC – за спаданням.

У конструкції ORDER BY можна використовувати будь-які поєднання стовпців за умови, що ці стовпчики знаходяться в таблицях, перерахованих у конструкції FROM. Слід зазначити, що конструкція ORDER BY може включати стовпець будь-якої таблиці (зі списку конструкції FROM), незалежно від того, чи згадується стовпець у списку вибірки конструкції SELECT.

Приклад 2.9. Отримаємо інформацію про всі товари, відсортувавши їх за ціною зростання.

```
SELECT *
FROM Products
ORDER BY Price;
```

Отримаємо:

ID	Category	Product	Number	Price
7	Photo	Nikon D5100	12	220.45
14	Monitor	LCD 23.6" AOC E2450Swd	22	323.19
15	Photo	SONY Alpha A35	41	350.77
6	Videocamera	JVC GZ-HM445	8	450.67
5	Monitor	LCD 20" Viewsonic VA2014w	5	455.14
4	Photo	CANON EOS 550D	11	511.43
11	Monitor	LCD 21.5" ASUS MS228H	11	544.00
12	Monitor	LCD 21.5" Philips 221TE2LB	85	545.32
17	Monitor	LCD 23" AOC E2343F	33	585.67
2	Monitor	LCD 16" Philips 166V3LSB	25	655.12
3	Videocamera	Canon Legria FS406	12	722.33
16	TV	LCD-TV SHARP LC-46LE630E	14	860.33
10	TV	LCD-TV PHILIPS 37PFL6606T	52	899.99
13	TV	LCD-TV SAMSUNG UE40D6530WSXUA	56	990.56
1	TV	LCD-TV PHILIPS 32PFL5606	10	1253.45
9	Photo	SONY Alpha A290	43	1453.18
8	Videocamera	SONY HDR-CX560E	24	1554.22

Приклад 2.10. Отримаємо інформацію про всі фотоапарати, відсортувавши їх за назвою за спаданням

```
SELECT *
FROM Products
WHERE Category = ‘Photo’ ORDER BY Product DESC;
```

Отримаємо:

ID	Category	Product	Number	Price
15	Photo	SONY Alpha A35	41	350.77
9	Photo	SONY Alpha A290	43	1453.18
7	Photo	Nikon D5100	12	220.45
4	Photo	CANON EOS 550D	11	511.43

Приклад 2.11. Отримаємо інформацію про всі товари, виконавши первинне сортування за категорією товару за зростанням і вторинне (внутрішне) сортування – за ціною за спаданням.

```
SELECT *
FROM Products
ORDER BY Category ASC, Price DESC;
```

Отримаємо:

ID	Category	Product	Number	Price
2	Monitor	LCD 16" Philips 166V3LSB	25	655.12
17	Monitor	LCD 23" AOC E2343F	33	585.67
12	Monitor	LCD 21.5" Philips 221TE2LB	85	545.32
11	Monitor	LCD 21.5" ASUS MS228H	11	544.00
5	Monitor	LCD 20" Viewsonic VA2014w	5	455.14
14	Monitor	LCD 23.6" AOC E2450Swd	22	323.19
9	Photo	SONY Alpha A290	43	1453.18
4	Photo	CANON EOS 550D	11	511.43
15	Photo	SONY Alpha A35	41	350.77
7	Photo	Nikon D5100	12	220.45
1	TV	LCD-TV PHILIPS 32PFL5606	10	1253.45

13	TV	LCD-TV SAMSUNG UE40D6530WSXUA	56	990.56
10	TV	LCD-TV PHILIPS 37PFL6606T	52	899.99
16	TV	LCD-TV SHARP LC-46LE630E	14	860.33
8	Videocamera	SONY HDR-CX560E	24	1554.22
3	Videocamera	Canon Legria FS406	12	722.33
6	Videocamera	JVC GZ-HM445	8	450.67

2.5. Агрегування даних

Агрегування (угруповання, збирання, накопичення) даних у запиті реалізується за допомогою конструкції GROUP BY і функцій агрегування. Слід зазначити, що NULL-значення (невизначені значення, будуть розглянуті в наступних лекціях) ігноруються в усіх агрегуючих функціях, крім COUNT (*), цей момент треба враховувати при аналізі результатів запиту. Зокрема, багато хто вважає, що при обчисленні середніх величин в стовпцях з числовими даними NULL-значення розглядаються як рівні нулю, але NULL-значення не рівні нулю і не повинні використовуватися як такі. Якщо функція AVG або інша агрегуюча функція застосовується до колонки з NULL-значеннями, то ці значення не увійдуть до складу результатів операції агрегування. Розглянемо деякі функції агрегування.

Агрегуюча функція COUNT().

Функція COUNT() призначена для обчислення кількості рядків у результаті запиту.

Приклад 2.12. Обчислимо загальну кількість рядків у таблиці.

```
SELECT COUNT (*)
FROM Products;
```

Отримаємо:

```
 + ------------ +
| COUNT (*) |
|              17 |
+ ------------ +
```

Приклад 2.13. Обчислимо кількість найменувань товарів, ціна яких перевищує 1000 умовних одиниць. Результату привласнемо псевдонім nProducts за допомогою ключового слова AS.

```
SELECT COUNT (*) AS nProducts
FROM Products
WHERE Price> 1000;
```

Отримаємо:

```
+ ---------- +
| NProducts |
|              3 |
+ ---------- +
```

Агрегуюча функція SUM().

Функція SUM () призначена для обчислення суми значень деякого стовпця.

Приклад 2.14. Обчислимо кількість усіх відеокамер у магазині.

```
SELECT SUM (Number) AS Sum FROM Products
WHERE Category = 'Videocamera'; Отримаємо:
+-------- +
|     Sum |
|       44 |
+-------- +
```

Агрегуючі функції MIN() і MAX()

Функції MIN (), MAX () призначені відповідно для визначення мінімального і максимального значень деякого стовпця.

Приклад 2.15. Визначимо найдешевший і найдорожчий товар.

```
SELECT MIN (Price) AS MinPrice, Max (Price) AS MaxPrice
FROM Products;
```

Отримаємо:

```
+ ---------- + ---------- +
| MinPrice | MaxPrice |
|    220.45 |   1554.22 |
+ ---------- + ---------- +
```

Агрегуюча функція AVG()

Функція AVG() призначена для визначення середнього значення деякого стовпця.

Приклад 2.16. Обчислимо середню ціну товарів на складі.

```
SELECT AVG (Price) AS Avg
FROM Products;
```

Отримаємо:

```
+ ------------ +
|          Avg |
| 727.989412 |
+ ------------ +
```

2.6. Конструкція GROUP BY

Конструкція GROUP BY <ім'я стовпця або виразу з використанням колонки в списку вибору> призначена для агрегування (групування) інформації у запиті. Групування даних за допомогою конструкції GROUP BY може здійснюватися за значеннями одного і більше стовпців або виразів. У разі використання кількох стовпців у списку угрупування вони розділяються комами. При використанні конструкції GROUP BY кожен стовпець у списку вибору оператора SELECT повинен являти собою або стовпець, зазначений у конструкції GROUP BY, або результат агрегування. Як правило, дана конструкція використовується спільно з функціями агрегування (COUNT(), SUM(), MAX() та ін.) Якщо не використовується конструкція GROUP BY, функції агрегування обробляють значення всіх рядків, що задовольняють запит. У разі, коли конструкція GROUP BY використовується, агрегуючі функції обробляють значення рядків окремо в кожній групі.

Приклад 2.17. Визначимо, скільки найменувань товарів відповідає кожній категорії.

```
SELECT Category, COUNT (Number) AS NProduct
FROM Products
GROUP BY Category;
```

Отримаємо:

```
+ ---------------------- +
| Category |  NProduct |
+ ---------------------- +
| Monitor        |      6 |
| Photo          |      4 |
| TV             |      4 |
| Videocamera |      3 |
+ ---------------------- +
```

Приклад 2.18. Визначимо кількість товарів в кожній категорії.

```
SELECT Category, SUM (Number) AS Number
FROM Products
GROUP BY Category;
```

Отримаємо:

```
+ ------------------------- +
| Category       |  Number |
+ ------------------------- +
| Monitor        |    181    |
| Photo          |    107    |
| TV             |    132    |
| Videocamera |     44    |
+ ------------------------- +
```

Приклад 2.19. Визначимо максимальну та мінімальну ціну товарів у кожній категорії

```
SELECT Category, MIN (Price) AS MinPrice, MAX (Price) AS MaxPrice
FROM Products
GROUP BY Category;
```

Отримаємо:

```
+ ---------------------------------------- +
| Category       | MinPrice | MaxPrice |
+ ---------------------------------------- +
| Monitor        | 323.19   |   655.12 |
| Photo          | 220.45   |  1453.18 |
| TV             | 860.33   |  1253.45 |
| Videocamera | 450.67   |  1554.22 |
+ ---------------------------------------- +
```

2.7. Конструкція HAVING

Конструкція HAVING призначена для накладення умов на групи, отримані за допомогою конструкції GROUP BY. Тому конструкція HAVING використовується, тільки якщо в запиті є конструкція GROUP BY.

Конструкція HAVING за призначенням «схожа» на конструкцію WHERE, але остання застосовується до кожного рядка ще до того, як цей

рядок увійшов до складу групи, а конструкція HAVING – до агрегованого «значення» групи.

Приклад 2.20. Визначимо категорії товарів, які включають більше трьох найменувань.

```
SELECT Category, COUNT(Product) AS NProduct
FROM Products
GROUP BY Category
HAVING NProduct> 3;
```

Отримаємо:

```
+ ---------- + ---------- +
| Category | NProduct |
+ ---------- + ---------- +
| Monitor|      6       |
| Photo      |   4       |
| TV         |   4       |
+ ---------- + ---------- +
```

Приклад 2.21. Визначимо категорії товарів, середня ціна яких перевищує 800 умовних одиниць

```
SELECT Category, AVG (Price) AS AvgPrice
FROM Products
GROUP BY Category
HAVING AvgPrice > 800;
```

Отримаємо:

```
+-----------------+----------------+
| Category        | AvgPrice       |
+-----------------+----------------+
| TV              | 1001.082500 |
| Videocamera  | 909.073333   |
+-----------------+----------------+
```

2.8. Предикат DISTINCT

Предикат DISTINCT призначений для усунення дубльованих даних в результаті запиту. При виявленні повторюваних значень будь-який наступний екземпляр не враховується (якщо конструкція DISTINCT використову-

ється разом з функцією COUNT(), то підрахунок унікальних значень також здійснюється один раз).

Предикат DISTINCT застосовується на початку списку вибірки або в операторі COUNT(), якщо потрібно виконати підрахунок кількості неповторюваних значень.

Приклад 2.22. Виведемо категорії товарів.

```
SELECT Category FROM Products;
```

Отримаємо:

```
+----------------+
| Category       |
+----------------+
| TV             |
| Monitor        |
| Videocamera |
| Photo          |
| Monitor        |
| Videocamera |
| Photo          |
| Videocamera |
| Photo          |
| TV             |
| Monitor        |
| Monitor        |
| TV             |
| Monitor        |
| Photo          |
| TV             |
| Monitor        |
+----------------+
```

Приклад 2.23. Виведемо категорії товарів, усунувши надмірність даних у результаті запиту.

```
SELECT DISTINCT Category
FROM Products;
```

Отримаємо:

```
+----------------+
| Category       |
+----------------+
| TV             |
| Monitor        |
| Videocamera |
| Photo          |
+----------------+
```

Приклад 2.24. Визначимо кількість категорій товарів.

```
SELECT COUNT (DISTINCT Category)
FROM Products;
```

Отримаємо:

```
+-------------------------------------+
| COUNT (DISTINCT Category) |
+-------------------------------------+
|                                   4 |
+-------------------------------------+
```

Контрольні запитання і завдання

1. Що являє собою БД?

2. Що являє собою СУБД? Наведіть приклади СУБД.

3. Що являє собою мова SQL?

4. Укажіть призначення оператора SELECT. Наведіть синтаксис оператора.

5. Укажіть призначення конструкції FROM в операторі SELECT. Наведіть приклад використання.

6. Укажіть призначення конструкції WHERE в операторі SELECT, операції, які можуть використовуватися в конструкції WHERE. Наведіть приклад використання.

7. Укажіть призначення конструкції ORDER BY в операторі SELECT. Наведіть приклад використання.

8. Укажіть призначення агрегуючих функцій COUNT(), SUM(), MIN(), MAX(), AVG() в операторі SELECT. Наведіть приклад використання.

9. Укажіть призначення конструкції GROUP BY в операторі SELECT. Наведіть приклад використання.

10. Укажіть призначення конструкції HAVING в операторі SELECT. Наведіть приклад використання.

11. Укажіть призначення конструкції предиката DISTINCT в операторі SELECT. Наведіть приклад використання.

12. Наведіть SELECT-запит для вибирання категорій товарів без повторення з таблиці Products (табл. 2.1).

13. Наведіть SELECT-запит для вибирання загальної кількості телевізорів з таблиці Products (табл. 2.1).

14. Наведіть SELECT-запит для вибирання загальної суми вартості (грошей) стереосистем з таблиці Products (табл. 2.1).

15. Наведіть SELECT-запит для вибирання максимальної та мінімальної ціни телевізора з таблиці Products (табл. 2.1).

16. Наведіть SELECT-запит для вибирання максимальної та мінімальної кількості поставок моніторів з таблиці Products (табл. 2.1).

17. Наведіть SELECT-запит для вибирання загальної кількості поставок для кожної категорії товару з сортуванням за кількістю поставок з таблиці Products (табл. 2.1).

18. Наведіть SELECT-запит для вибирання найменування найдешевшого телевізора з таблиці Products (табл. 2.1).

19. Наведіть SELECT-запит для вибирання найменування найдорожчого монітора з таблиці Products (табл. 2.1).

20. Наведіть SELECT-запит для вибирання найменувань товарів довжиною 14 символів з таблиці Products (табл. 2.1).

21. Наведіть SELECT-запит для вибирання кількості найменувань товарів довжиною 20 символів з таблиці Products (табл. 2.1).

Порядок виконання роботи

1. Отримати у викладача завдання на лабораторну роботу – список SELECT-запитів, які необхідно сконструювати.

2. Зробити сценарій з запитами.

3. Запустити на персональному комп'ютері клієнтську програму mysql для роботи з сервером MySQL.

4. Запустити зроблений сценарій на виконання за допомогою команди source в командному рядку програми mysql.

Список літератури

1. Роланд Ф.Д. Основные концепции баз данных / Ф.Д. Роланд. – М. : Издательский дом "Вильямс", 2002. – 256 с.

2. Кренке Д. Теория и практика построения баз данных / Д. Кренке. – СПб. : Питер, 2005. – 800 с.

3) Карпова Т.С. Базы данных: модели, разработка, реализация / Т.С. Карпова. – СПб. : Питер, 2001. – 304 с.

4) Кузнецов М.В. MySQL 5 /М.В. Кузнецов, И.В. Симдяев. – СПб. : БХВ-Петербург, 2010. – 1024 с.

ЗМІСТ

Навчальне видання

Методичні вказівки до виконання лабораторної роботи за темою
“Використання оператора SELECT в СУБД MySQL”
з курсів «Бази даних та знань», «Бази даних та інформаційні системи»
для студентів спеціальностей «Інформатика», «Прикладна математика»

Укладачі: БАГМУТ Іван Олександрович,
ГАЛАС Олег Сергійович

Відповідальний за випуск Д. В. Бреславський

Редактор Н. В. Верстюк

План 2013 р., поз. 15

Підп. до друку 2013 р. Формат 60х84 1/16. Папір Mogpa.
Riso-друк. Гарнітура Таймс. Ум. друк. арк.
Наклад 150 прим. Зам.№ .Ціна договірна.

Видавничий центр НТУ «ХПІ».

Свідоцтво про державну реєстрацію ДК № 3657 від 24.12.2009 р.

61002, Харків, вул. Фрунзе, 21

Друкарня НТУ “ХПІ”,
61002, Харків, вул. Фрунзе, 21