2020

Lab Manual

Databases lab

Retrieving Data Using the SQL SELECT Statement-2-

Safa Khader Alwan
Al-Balqa Applied University
2/22/2020

Student information:

Student name	
Student number	
Grade	

Objectives

After completing this lesson, you should be able to do the following:

- Limit the rows that are retrieved by a query
- Sort the rows that are retrieved by a query

Limiting Rows Using a Selection

Restrict the rows that are returned by using the WHERE clause:

```
SELECT * |{[DISTINCT] column | expression [alias],...}
FROM table
[WHERE condition (s) ] ;
```

The WHERE clause follows the FROM clause. A WHERE clause is generally made up of three elements:

1. Column name
2. Comparison operator (logical operator)
3. Column name/literal

A simple SELECT statement, such as:

```
SELECT employee id, last name, job id, department id
FROM employees
```

will return all rows from the emp table, but if you only want a list of employees who work in department 10, you would use the WHERE clause. The WHERE clause MUST appear after the FROM clause. You specify conditions in the WHERE clause that must be met if the row is to be returned. Conditions are basically comparisons of columns/literals using logical operators and SQL operators. Here is an example:

Example 1:

```
SELECT employee id, last name, job id, department id
FROM employees
WHERE department_id = 10
```

Row Restriction using Logical Operators

The following logical operators are available:

Operator	Meaning
=	Equal to
>	Greater than
>=	Greater than or equal to
<	Less than
<=	Less than or equal to
<>	Not equal to
BETWEEN ...AND ...	Between two values (inclusive)
IN (set)	Match any of a list of values
LIKE	Match a character pattern
IS NULL	Is a null value

Using Comparison Conditions

Example 2:

```
SELECT employee_id
FROM employees
WHERE hiredate <= '01-Jan-98';
```

Example 3:

```
SELECT employee_id
FROM employees
WHERE  job = 'CLERK';
```

Example 4:

```
SELECT employee_id
FROM employees
WHERE department_id = 10;
```

Example 5:

```
SELECT employee_id
FROM employees
WHERE  salary>10000;
```

Using the BETWEEN Condition

Use the BETWEEN condition to display rows based on a range of values:

Example 6:

```
SELECT employee_id
FROM employees
WHERE  department_id BETWEEN 20 AND 50;
```

Example 7:

```
SELECT employee_id
FROM employees
WHERE  department_id BETWEEN 20 AND 50;
```

Example 7:

```
SELECT employee_id,last_name,salary
FROM employees
WHERE  salary  BETWEEN 10000 AND 1500;
```

Using the IN Condition

Use the IN membership condition to test for values in a list:

Example 8:

```
SELECT employee_id,last_name,salary
FROM employees
WHERE  department_id in(10,50,60,70,90);
```

Using the LIKE Condition

Sometimes you may not know the exact value to search for: for example, you may want a list of all employees whose name begins with the letter S. You perform this kind of comparison using the LIKE operator. The LIKE operator expects a wildcard search pattern. A wildcard is basically a way to specify parts of a string which you do not know. Wildcard Symbols Available

1. % : denotes zero or many characters.
2. _ denotes one characters.

Example 9: To list all employees whose names begin with S:

```
SELECT employee_id,last_name,salary
FROM employees
WHERE  last_name like 'S%';
```

Example 9: To list all employees whose names end with S:

```
SELECT employee_id,last_name,salary
FROM employees
WHERE  last_name like '%S';
```

Example 9: To list all employees whose names contain S:

```
SELECT employee_id,last_name,salary
FROM employees
WHERE  last_name like '%S%';
```

Using the NULL Conditions

Example 10:

```
SELECT employee_id,last_name,salary
FROM employees
WHERE  commission_pct is null;
```

Logical Conditions

Operator	Meaning
AND	Returns **TRUE** if *both* component conditions are true
OR	Returns **TRUE** if *either* component condition is true
NOT	Returns **TRUE** if the following condition is false

Example 11:

```
SELECT employee_id,last_name,salary
FROM employees
WHERE  commission_pct is NULL AND salary>10000;
```

Example 11: select all rows where the department is 10 AND the salary is greater than 1000:

```
SELECT employee_id,last_name,salary
FROM employees
WHERE  department_id=10 AND salary>10000;
```

Example 12: select all rows where the department is 10 or the salary is greater than 1000:

```
SELECT employee_id,last_name,salary
FROM employees
WHERE  department_id=10 OR salary>10000;
```

Example 12: To select all employees who were hired before July 1999 whose commission is more than £1,000 but less than £1,500:

```
SELECT employee_id,last_name,salary
FROM employees
WHERE  hire_date < '01-jul-99'
AND
NVL(commission_pct,0) BEWTEEN 1001 AND 1499;
```

Exercise 1: To select all employees who are not clerks but do work in departments 10, 40 or 50:

Exercise 2: To select employees who work in department 10 who earn more than £10,000 per annum, or employees who work in department 30 who earn more than £15,000 per annum:

Rules of Precedence

When constructing a WHERE clause you need to be aware of operator precedence. This determines how a condition is evaluated, and it can greatly affect the results. Operators are evaluated in a strict order, as follows:

Operator	Meaning
1	Arithmetic operators
2	Concatenation operator
3	Comparison conditions
4	IS [NOT] NULL, LIKE, [NOT] IN
5	[NOT] BETWEEN
6	Not equal to
7	NOT logical condition
8	AND logical condition
9	OR logical condition

To select all managers in any department AND all clerks in department 10:

```
SELECT *
FROM employees
WHERE department_id = 50
AND job_id='ST_CLERK' OR JOB_ID ='MANAGER' ;
```

The following would produce different results:

```
SELECT *
FROM employees
WHERE department_id = 50
AND (job_id='ST_CLERK' OR JOB_ID ='MANAGER' );
```

The second statement says "select all rows where department _idis 50 and job is either CLERK or MANAGER -this would only give rows where department is 10 AND job is MANAGEROR CLERK.

Using the ORDER BY Clause

The order of rows returned by a SELECT statement is, by default, undefined. You can use the ORDER BY clause to sort the rows.

- ASC: ascending order, default
- DESC: descending order

- The ORDER BY clause comes last in the SELECT statement:

Example 13:

```
SELECT employee_id
FROM employees
ORDER BY employee_id;
```

will give something like:

```
employee_id
----------
10001
10002
10003
10004
. . .
```

Default order is ascending - use DESC after the column name to change order

- There is no limit on the number of sort columns
- There is no need to SELECT sort column
- You can sort using expressions and aliases
- NULL values are sorted high

Example 14:

```
SELECT last_name , salary*12 + NVL(commission_pct,0) renum
FROM employees
ORDER BY renum DESC;
```

Example 14:

```
SELECT department_id , hire_date , last_name
FROM employees
ORDER BY department_id, hire_date DESC ,last_name
```

Example 13:

```
SELECT employee_id
FROM employees
ORDER BY employee_id;
```

Summary

To summarise, we have seen the basic syntax for the SELECT statement:

```
SELECT [DISTINCT]{*,COLUMN [ALIAS]….}

FROM TABLE

WHERE CONDITION(S)

ORDER BY {COLUMN|EXPRESSION}[ASC|DESC]};
```

We have covered the following:

- Basic SELECT clause
- The DISTINCT keyword
- Column Aliases
- The ORDER BY clause
- The WHERE clause
  - Single/Multiple conditions
  - Logical/SQL Operators
  - Negating conditions
  - Multiple conditions
  - Operator Precedence

Lab exercises:

Exercise 1:

Below is a selection from the "Customers" table:

CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitución 2222	México D.F.	05021	Mexico
3	Antonio Moreno Taquería	Antonio Moreno	Mataderos 2312	México D.F.	05023	Mexico
4	Around the Horn	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK
5	Berglunds snabbköp	Christina Berglund	Berguvsvägen 8	Luleå	S-958 22	Sweden

1. selects all customers with a City starting with the letter "s";
2. selects all customers with a City ending with the letter "s";
3. selects all customers with a Country containing the pattern "land";
4. selects all customers with a Country NOT containing the pattern "land";
5. selects all customers with a City of "Paris" or "London":

Exercise 2: To list all employees whose names have exactly 4 characters:

Exercise 3: write the SQL for the following:

1. Select all rows from the salgrade table.
2. Select all rows from the emp table.
3. Select all employees who have a salary between 1600 and 3000.
4. List department number and department name in name order from the dept table.
5. Display all the different job types, in reverse order.
6. List the names and hiredate of all clerks in department 20.
7. List all employees whose name begins with S.
8. Display the name, job, mgr and sal for all employee who have a manager. Sort the list by sal descending.
9. List the name and total remuneration of all employees.
10. Display the name, salary, annual salary and commission of all sales people whose monthly salary is less than their commission. The output should be sorted by salary, highest first.