DQL

By- Saurabh Kumar Sharma

Course Objective

- To retrieve data from MySQL database
- To implement conditions while retrieving the data
- To implement basic functions and explore advance function in MySQL

Session Objective

- DQL –Select
  - Arithmetic operators
  - Comparison conditions
- Order by clause
- Functions – Group functions
- Group by clause
- Having clause

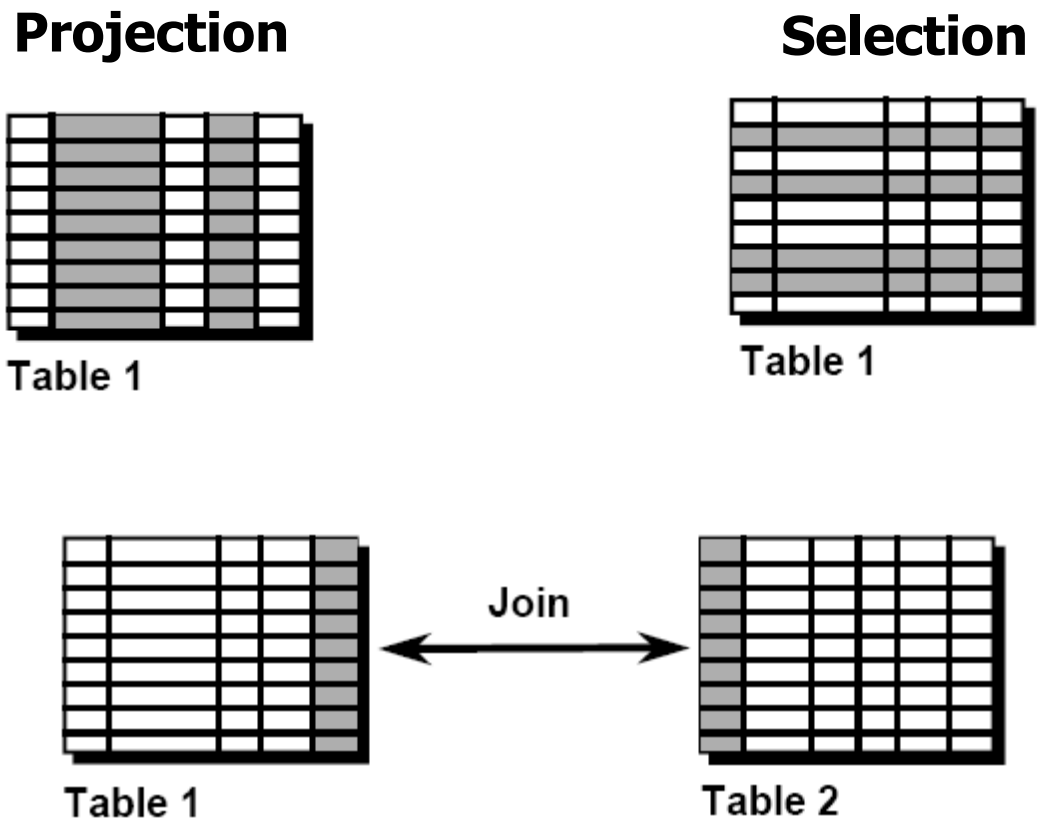
SQL

SQL stands for Structured Query Language
SQL allows you to access a database
SQL is an ANSI standard computer language
SQL can execute queries against a database
SQL can retrieve data from a database
SQL can insert new records in a database
SQL can delete records from a database
SQL can update records in a database

SQL Statements

SELECT	Data retrieval
INSERT UPDATE DELETE MERGE	Data manipulation language (DML)
CREATE ALTER DROP RENAME TRUNCATE	Data definition language (DDL)
COMMIT ROLLBACK SAVEPOINT	Transaction control
GRANT REVOKE	Data control language (DCL)

Capabilities Of SQL Select



Writing SQL Statements

- SQL statements are NOT case sensitive.
- SQL statements can be on one or more lines.
- Keywords cannot be abbreviated or split across lines.
- Clauses are usually placed on separate lines.
- Indents are used to enhance readability.

SQL Select

Syntax

```
SELECT [DISTINCT|ALL ]{*|[columnExpression[AS
newName]][,...]}
FROM TableName[Aliase][,...]
[WHERE condition]
GROUP BY columnList][HAVING condition]
[ORDER BY columnList]
```

Projection Capability

Projection Capability:

- Used to choose the columns in a table that you want returned by your query.
- Can be used to choose as few or as many columns of the table as you require.

Examples:

```
SELECT DISTINCT
Deptno
FROM Emp;
```

```
SELECT Deptno, dname, loc

FROM Dept;
```

```
SELECT *
FROM
Dept;
```

```
SELECT dname, Deptno

FROM Dept;
```

Column Alias Name

Renames a column heading by using the alias name through your query.

Examples:

```
SELECT Deptno AS "Dept No",
dname AS "Dept Name", loc AS
Location FROM Dept
```

```
SELECT Deptno "Dept No", dname
"Dept  Name", loc Location
FROM Dept;
```

```
SELECT Deptno Dept_no, dname
Dept_Name, loc Location
FROM Dept;
```

Arithmetic Operators

We can use arithmetic operators in any clause of a SQL statement except in the FROM clause.

Operators:

Example:

```
SELECT ename, sal,
sal*12

FROM emp;
```

Arithmetic Operators

Operator Precedence :

*	/	+	-

Concatenation Operator And Literals

Concatenation Operator(||):

Concatenates columns or character strings to other columns

Literal:

A literal is a character, a number, or a date that is included in the SELECT list

Examples:

```
SELECT ename||sal AS " NAME And
    SALARY"
FROM emp;
```

```
SELECT ename||' is earning Rs.'|| sal
    ||' per month' AS " NAME And
SALARY"
FROM emp
```

Selection Capability

Selection Capability:

- Used to choose the rows in a table that you want returned by a query.
- Various criteria can be used to restrict the rows that you see.
- Restrict the rows returned, by using the WHERE clause.

Syntax:

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

Operators Used IN Where Clause

Comparison :

= , <>,< , >
,<=, >=

Logical :

AND,
OR ,
NOT

Range:

BETWEEN , NOT BETWEEN

Pattern Match:

LIKE ,
NOT LIKE

Set Membership:

IN ,
NOT IN

Null:
IS NULL ,
IS NOT NULL

Comparison Search Condition

Comparison Conditions :

Conditions that compare one expression to another value or expression.

Examples:

```
SELECT * FROM emp
WHERE sal>1000 AND deptno=20
```

```
SELECT *
FROM emp
WHERE deptno=20;
```

```
SELECT *
FROM emp
WHERE
deptno = 20
OR
Deptno=30
```

Range Search Condition

Range Condition:

You can display rows based on a range of values using the BETWEEN range condition. The range that you specify contains a lower limit and an upper limit

Examples:

```
SELECT ename, job, sal
FROM emp
WHERE sal BETWEEN 3000 AND 5000; -
(includes 3000 and 5000)

SELECT ename, job, sal
FROM emp
WHERE sal NOT BETWEEN 3000 AND 5000
```

Set Membership search Conditions

Set Membership

- Used to test for values in a specified set of values.
- Uses the keyword:

  IN

  NOT IN
- The *membership* condition is also known as IN *condition.*

Examples:

```
SELECT ename, hiredate,job
FROM emp
WHERE job IN (‘MANAGER', 'CLERK',
'ANALYST');
```

```
SELECT ename,hiredate,job
FROM emp
WHERE job NOT IN ('MANAGER', 'CLERK',
'ANALYST‘ );
```

Pattern Match Search Condition

The pattern-matching operation is referred to as a *wildcard* search.

Two symbols can be used to construct the search string.

SQL has two special Pattern Matching symbols (wildcard)

% - represents any sequence of zero or more characters

_ - represents any single character

Examples

```
SELECT
ename,job,hiredate
WHEREFROM emp
 job LIKE  '_A%';
```

```
SELECT ename,job,hiredate
FROM emp
WHERE hiredate LIKE '%82';
```

```
SELECT ename,job,hiredate
FROM emp
WHERE ename LIKE 'A%‘
```

NULL Search Condition

NULL

- Means the value is Unavailable, unassigned ,unknown, or inapplicable.
- Cannot be tested with = because a null cannot be equal or unequal to any value.
- Include the IS NULL condition and the IS NOT NULL condition.
- IS NULL condition tests for nulls.

Examples:

```
SELECT ename, job, comm FROM emp WHERE comm IS NULL
```

```
SELECT ename, job, comm FROM emp WHERE comm IS NOT NULL
```

ORDER BY Clause

Sort rows with the ORDER BY clause

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

The ORDER BY clause comes last in the SELECT statement.

Single column Ordering:

Examples:

```
SELECT ename, job, sal  FROM
emp
ORDER BY sal;
```

```
SELECT ename, job, sal ,hiredate
FROM emp
ORDER BY hiredate DESC
```

ORDER BY Clause

Sort rows with the ORDER BY clause

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

Single column Ordering:

```
SELECT ename, job, sal  FROM emp
ORDER BY sal;
```

```
SELECT ename, job, sal
,hiredate FROM emp
ORDER BY hiredate DESC
```

Multiple column Ordering:

```
SELECT ename, job,
deptno, sal,hiredate
mp ORDER BY deptno
DESC, sal Asc
```

Function -MySQL

MySQL functions including aggregate functions, string functions, date time functions, control flow functions, etc.

Functions
Aggregate
String
Control Flow
Date and Time
Comparison
Numeric

Group Functions

Group functions operate on sets of rows

to give one result per group.

EMPLOYEES

DEPARTMENT_ID	SALARY
90	24000
90	17000
90	17000
60	9000
60	6000
60	4200
50	5800
50	3500
50	3100
50	2600
50	2500
80	10500
80	11000
80	8600
	7000
10	4400

...

20 rows selected.

The maximum salary in the EMPLOYEES table.

MAX(SALARY)
24000

Group Functions

Function Name	Example
Sum	SELECT SUM(salary) AS TotalSalary FROM employee;
Avg	Select Avg(salary) as AVGSalary from employee
Count	Select count(salary) NoOfEmployee from employees
Max	Select max(salary) as MaxSalary from employees
Min	Select min(salary)as MinSalary from employees

Group Functions

COUNT

Examples:

SELECT COUNT(*) AS No_Of_Employees FROM emp;

SELECT COUNT(deptno) AS Departments FROM emp;

SELECT COUNT(DISTINCT deptno) as Departments FROM emp;

COUNT(*) - Counts all rows of a table, regardless of whether nulls or duplicate values occur

The GROUP BY Clause

Divide rows in a table into smaller groups

EMPLOYEES

DEPARTMENT_ID	SALARY
10	4400
20	13000
20	6000
50	5800
50	3500
50	3100
50	2500
50	2600
60	9000
60	6000
60	4200
80	10500
80	8600
80	11000
90	24000
90	17000

...

20 rows selected.

4400

9500

3500

6400

10033

The average salary in EMPLOYEES table for each department.

DEPARTMENT_ID	AVG(SALARY)
10	4400
20	9500
50	3500
60	6400
80	10033.3333
90	19333.3333
110	10150
	7000

The GROUP BY Clause

- Aggregate functions are normally used in conjunction with a GROUP BY clause.
- The GROUP BY clause enables the aggregate functions to answer more complex managerial Queries

Guidelines for Group by Clause

- All columns in the SELECT list that are not in group functions must be in the GROUP BY clause.
- GROUP BY clause does not support the use of column alias, but the actual names.
- GROUP BY clause can only be used with aggregate functions like SUM, AVG, COUNT, MAX, and MIN. If it is used with single row functions,Oracle throws and exception as "ORA-00979: not a GROUP BY expression".
- Aggregate functions cannot be used in a GROUP BY clause. Oracle will return the "ORA-00934: group function not allowed" here error message.

The GROUP BY Clause

Syntax

SELECT [column,] group_function(column), ...
FROM table
[WHERE condition]
[GROUP BY column]
[ORDER BY column];

Examples:

select count(empno) FROM emp GROUP BY deptno

SELECT deptno, COUNT(empno) AS EmployeeCount, SUM(sal) AS Total_Salary FROM emp GROUP BY deptno;

Grouping more than one column:

Examples:

SELECT job,deptno,SUM(sal) AS Total_Salary
FROM emp GROUP BY job,deptno

Restricting Groupings – Having Clause

EMPLOYEES

DEPARTMENT_ID	SALARY
90	24000
90	17000
90	17000
60	9000
60	6000
60	4200
50	5800
50	3500
50	3100
50	2600
50	2500
80	10500
80	11000
80	8600
...	
20	6000
110	12000
110	8300

20 rows selected.

DEPARTMENT_ID	MAX(SALARY)
20	13000
80	11000
90	24000
110	12000

Restricting Groupings – Having Clause

The HAVING clause is used for aggregate functions in the same way that a WHERE clause is used for column names and expressions.

Example:

```
SELECT JOB_ID,SUM (SALARY)
        FROM employees
        GROUP BY JOB_ID
        HAVING SUM (SALARY) > 10000;
```

Execute without error

```
SELECT department_id,
AVG(Salary)
FROM employees
HAVING AVG(Salary) > 33000;
```

ERROR at line 1: ORA-00937: not a single-group group function

Having Clause with Where clause

In the same way that you use the WHERE clause to restrict the rows that you select, you use the HAVING clause to restrict Groups

Syntax:

```
SELECT column, group_function
FROM table
[WHERE condition]
[GROUP BY group_by_expression]
[HAVING group_condition]
[ORDER BY column];
```

Having Clause with Where clause cont…

Example:

```
SELECT city, AVG(salary)  FROM employee
 WHERE salary < 7000  GROUP BY city
 HAVING AVG(salary) > 1500;
```

Using the WHERE, GROUP BY, and HAVING Clauses Together

- The WHERE clause first filters the rows,
- And the remaining rows are grouped into blocks by using GROUP BY clause,
- Finally the row groups are filtered by the HAVING clause.

Readings reference:

- https://www.w3schools.com/sql/sql_ref_mysql.asp
- https://www.techonthenet.com/mysql/functions/
- https://www.w3resource.com/mysql/mysql-functions-and-operators.php
- https://www.tutorialspoint.com/mysql/mysql-useful-functions.htm
- http://www.mysqltutorial.org/mysql-functions.aspx

Questions