Введение в SQL (Structured Query Language)

github.com/HowProgrammingWorks

Timur Shemsedinov

Chief Technology Architect at Metarhia
Lecturer at Kiev Polytechnic Institute

github.com/tshemsedinov

SELECT Statement

The SELECT statement is used to query the database and retrieve selected data that match the criteria that you specify.

```
SELECT Id, Login, FullName
FROM SystemUser
WHERE Login = “Marcus”;
```

SELECT Clauses

The SELECT statement has five main clauses to choose from, although, FROM is the only required clause. Each of the clauses have a vast selection of options, parameters, etc.

SELECT..FROM, WHERE,
GROUP BY, HAVING, ORDER BY

SELECT statement format

```
SELECT [ALL | DISTINCT] column1[, column2]...
FROM table1[, table2]...
[WHERE <conditions>]
[GROUP BY <columns>]
[HAVING <conditions>]
[ORDER BY <columns> [ASC | DESC] ]
```

SELECT..WHERE: Comparison Operators

=	Equal
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to
!=	Not equal to (also <>)
LIKE	String comparison test

SELECT..WHERE: LIKE %

This will select all records with Login starts with “Mar”

```
SELECT Id, Login, FullName
FROM SystemUser
WHERE Login LIKE “Mar%”;
```

SELECT..WHERE: LIKE _

This will select all records with Login
starts with “Mar”, ends with “us”
and contains any 4th letter in the middle

```
SELECT Id, Login, FullName
FROM SystemUser
WHERE Login LIKE “Mar_us”;
```

SELECT DISTINCT

If you would like to retrieve just the unique records in specified columns, you can use the "DISTINCT" keyword. DISTINCT will discard the duplicate records for the columns you specified after the "SELECT" statement:

```
SELECT DISTINCT FullName FROM SystemUser;
```

SELECT ALL

ALL will display "all" of the specified columns including all of the duplicates. The ALL keyword is the default if nothing is specified.

```
SELECT Id, Login, FullName FROM SystemUser
WHERE Id =
    ALL (SELECT UserId WHERE IP = “127.0.0.1”);
```

SELECT ANY

The ANY operator is used with a WHERE or HAVING clause. The ANY operator returns true if any of the subquery values meet the condition.

```
SELECT Id, Login, FullName FROM SystemUser
WHERE Id =
    ANY (SELECT UserId WHERE IP = “127.0.0.1”);
```

Aggregate Functions

MIN	the smallest value in a given column
MAX	the largest value in a given column
SUM	the sum of the values in a given column
AVG	the average value of a given column
COUNT	the total number of values in a col.
COUNT(*)	the number of rows in a table

Aggregate Functions

Aggregate functions are used to compute against a "returned column of numeric data" from your SELECT. They basically summarize the results of a particular column of selected data.
Aggregate functions are required for the "GROUP BY", these functions can be used without the "GROUP BY" clause.

Aggregate Functions

```
SELECT avg(Salary) FROM Employee;

SELECT avg(Salary) FROM Employee
GROUP BY Department;

SELECT count(*) FROM Employee;
```

GROUP BY clause

The GROUP BY clause will gather all of the rows together that contain data in the specified column(s) and will allow aggregate functions to be performed on the one or more columns.

```
SELECT Item, sum(Price)
FROM Goods GROUP BY Supplier;
```

GROUP BY clause

Following statement will select the maximum salary for the people in each department.

```
SELECT max(Salary), Department
FROM Employee GROUP BY Department;
```

HAVING clause

The HAVING clause allows you to specify conditions on the rows for each group. In other words, which rows should be selected will be based on the conditions you specify.

SELECT column1, SUM(column2) FROM <tables>
GROUP BY <columns> HAVING <condition>;

HAVING clause

Let's say you have an employee table with cols: Name, Department, Salary. If you would like to select the average salary for each Employee in each Department if their salary is over 200:

```
SELECT Department, avg(Salary) FROM Employee
GROUP BY Department HAVING avg(Salary) > 200;
```

ORDER BY clause

ORDER BY is an optional clause which will allow you to display the results of your query in a sorted order (ascending or descending) based on the columns specified column.

SELECT column1, SUM(column2) FROM <tables>
ORDER BY <columns> [ASC | DESC];

ORDER BY clause

This statement will select the Id, Department, Name, and Salary from the Info table where the Department equals 'Sales' and will list the results in Ascending order based on Salary:

```
SELECT Id, Department, Name, Salary FROM Info
WHERE Department = 'Sales' ORDER BY Salary;
```

ORDER BY clause

You can order based on multiple columns.
To do so you can specify columns list:

```
SELECT Id, Department, Name, Salary, Age
FROM Info WHERE Department = 'Sales'
ORDER BY Salary,  Age DESC;
```

Combining Conditions

The AND operator can be used to join two or more conditions in the WHERE clause.
Both sides of the AND condition must be true for those rows to be displayed.

SELECT column1, SUM(column2) FROM <tables>
WHERE <condition1> AND <condition2>;

Combining Conditions

The OR operator can be used to join two or more conditions in the WHERE clause also. With the OR operator, either side can be true or both.

```
SELECT Id, Name, Title, Salary FROM Info
WHERE Title = “Sales” OR
    (Title = “Programmer” AND Salary >= 45000);
```

WHERE..IN condition

The IN conditional operator is really a set membership test operator, whether or not a value is "in" the set.

SELECT col1, SUM(col2) FROM <tables>
WHERE col3 IN (<values>);

WHERE..IN condition

To select all employees from listed departments:

```
SELECT Id, Department, Salary FROM Employee
WHERE Department IN ('Software', QA');
```

```
SELECT Id, Department, Salary FROM Employee
WHERE   Department = 'Software' OR Department = QA';
```

WHERE..BETWEEN condition

The BETWEEN conditional operator is used to check whether a value is "between" the two values specified after BEFORE and separated AND.
You can also use NOT BETWEEN.

```
SELECT col1, SUM(col2) FROM <tables>
WHERE col3 BETWEEN value1 AND value2;
```

WHERE..BETWEEN condition

```
SELECT Id, Age, LastName, Salary
FROM Employes
WHERE Age BETWEEN 30 AND 40;
```

```
SELECT Id, Age, LastName, Salary
FROM Employee
WHERE Age >= 30 AND Age <= 40;
```