SQL Training

Presentation by Uplatz
Contact us: https://training.uplatz.com
Email: info@uplatz.com
Phone: +44 7836 212635

SQL Select

Uplatz

- The SELECT statement is used to select data from a database.
- The data returned is stored in a result table, called the result-set.

```
select * from world.country;
select name, CountryCode, District from world.city;
select * from world.country limit 5;  (To select few records)
select CountryCode from world.city where CountryCode = 'AFG’;
select distinct CountryCode from world.city where CountryCode = 'AFG';
```

SQL WHERE

Uplatz

- The WHERE clause is used to filter records.
- The WHERE clause is used to extract only those records that fulfill a specified condition.

```
select CountryCode from world.city where CountryCode = 'AFG’;
select * from uplatz.persons where age > 20;
```

SQL AND, OR, NOT

The WHERE clause can be combined with AND, OR, and NOT operators.

The AND and OR operators are used to filter records based on more than one condition:

The AND operator displays a record if all the conditions separated by AND are TRUE.

The OR operator displays a record if any of the conditions separated by OR is TRUE.

The NOT operator displays a record if the condition(s) is NOT TRUE.

AND LOGIC

TRUE	TRUE	TRUE
TRUE	FALSE	FALSE
FALSE	TRUE	FALSE
FALSE	FALSE	FALSE

OR LOGIC

TRUE	TRUE	TRUE
TRUE	FALSE	TRUE
FALSE	TRUE	TRUE
FALSE	FALSE	FALSE

NOT LOGIC

TRUE	FALSE
FALSE	TRUE

SQL AND, OR, NOT

Uplatz

```
select * from world.city where CountryCode = 'nld' and population > 100000 and district = 'Noord-Holland’;

select * from world.city where CountryCode = 'nld' and population > 100000 and (district = 'Noord-Holland' OR district = 'Zuid-Holland’);

select * from world.countrylanguage where language = 'Spanish' or language = 'Uzbek’;

Select * from uplatz.persons where not age = 30;
```

SQL ORDER BY

Uplatz

```
Select * from uplatz.persons order by age;
select * from uplatz.persons order by id desc, age;
select * from uplatz.persons order by age, id desc;
```

SQL NULL values

Uplatz

```
insert into uplatz.users values (8, "uplatz", NULL);
select * from uplatz.users;
select * from uplatz.users where age is NULL;
```

SQL delete

Delete from uplatz.users;

Delete from uplatz.users where id = 10;

Thank you

SQL Training
Presentation by Uplatz
Contact us: https://training.uplatz.com
Email: info@uplatz.com
Phone: +44 7836 212635
select * from uplatz.courses

SQL Update

- The UPDATE statement is used to modify the existing records in a table.
- Update uplatz.users set age = 100 where id = 3
- Update uplatz.users set name = ‘uplatz’ where id = 3;

SQL Min/ Max

- The MIN() function returns the smallest value of the selected column.
- The MAX() function returns the largest value of the selected column.
- Select min(age), max(age) from uplatz.persons;
- Select min(lastname), max(lastname) from uplatz.persons; --arranges in a alphabetical order and finds min and max functions
- **MAX, MIN** ignores any **null values**. **MAX** returns **NULL** when there is no row to **select**. For character columns, **MAX** finds the highest **value** in the collating sequence. **MAX** is a deterministic function when used without the OVER and ORDER BY clauses.

SQL Count/Avg/Sum

- The COUNT() function returns the number of rows that matches a specified criterion.
- The AVG() function returns the average value of a numeric column.
- The SUM() function returns the total sum of a numeric column.

```
select count(*), count(1) from uplatz.persons;

select count(*), count(1), count(age) from uplatz.persons; --Taking count of a column returns count of records having nor null value in the column of that table

select count(coalesce(age, 0)) from uplatz.persons;
```

The most common argument of group which supports the use of **count**(**1**) is the assertion that **COUNT**(**1**) is **faster** than **COUNT(*)**. According to this theory **COUNT(*)** takes all columns to **count** rows and **COUNT**(**1**) **counts** using the first column: Primary Key.

SQL Count/Avg/Sum

- select avg(age) from uplatz.persons
- select (30+20+20+100)/4;
- select avg(id) from uplatz.persons;
- select (1+2+3+4+5+10)/6;
- select sum(age) from uplatz.persons;
- select 30+20+20+100;

SQL group by

- The GROUP BY statement groups rows that have the same values into summary rows, like "find the number of customers in each country".
- The GROUP BY statement is often used with aggregate functions (COUNT, MAX, MIN, SUM, AVG) to group the result-set by one or more columns.

```
create table uplatz.demo as select countrycode, count(distinct language) from countrylanguage group by countrycode;

select countrycode, IsOfficial, count(distinct language) from countrylanguage where countrycode = 'ABW' group by countrycode limit 5;

select * from countrylanguage  where countrycode = 'ABW';

select countrycode, IsOfficial, count(distinct language) from countrylanguage where countrycode = 'ABW' group by countrycode, IsOfficial limit 5;
```

SQL group by

```
select * from countrylanguage  where countrycode = 'AFG';

select countrycode, IsOfficial, count(distinct language) from countrylanguage where countrycode = 'AFG' group by countrycode limit 5;

select countrycode, IsOfficial, count(distinct language) from countrylanguage where countrycode = 'AFG' group by countrycode, isofficial limit 5;
```

SQL having

- The HAVING clause was added to SQL because the WHERE keyword could not be used with aggregate functions.

Let us try to find countries with more than 1 language.

select countrycode, count(distinct language) from countrylanguage group by countrycode having count(distinct language) > 1 limit 5;

select countrycode, count(distinct language) from countrylanguage group by countrycode having count(distinct IsOfficial) = 1 limit 5;

select * from countrylanguage where countrycode = 'AGO’;

SQL LIKE

- The LIKE operator is used in a WHERE clause to search for a specified pattern in a column. It can also be used in SELECT clause
- There are two wildcards often used in conjunction with the LIKE operator:
- % - The percent sign represents zero, one, or multiple characters
- _ - The underscore represents a single character
- **Note:** MS Access uses an asterisk (*) instead of the percent sign (%), and a question mark (?) instead of the underscore (_).

Select ‘hello’ = ‘hello’

select 'hello' like '%hell%’;

select 'hello' like '%helloworld%’;

select 'hello_world' like '%_world’;

select 'hello_world' like 'h%d’;

SQL IN

- The IN operator allows you to specify multiple values in a WHERE clause.
- The IN operator is a shorthand for multiple OR conditions.

```
select * from uplatz.persons where id in (5,10);

select * from uplatz.persons where id in (select distinct id from uplatz.persons where age is NULL);
```

SQL between

- The BETWEEN operator selects values within a given range. The values can be numbers, text, or dates.
- The BETWEEN operator is inclusive: begin and end values are included.

```
select * from uplatz.persons where age between 70 and 100;
```

SQL Training
Presentation by Uplatz
Contact us: https://training.uplatz.com
Email: info@uplatz.com
Phone: +44 7836 212635
select * from uplatz.courses

SQL EXISTS

- The EXISTS operator is used to test for the existence of any record in a subquery.
- The EXISTS operator returns true if the subquery returns one or more records.

```
select * from uplatz.persons a where exists (select 1 from uplatz.users b where a.id = b.id);
```

```
select * from uplatz.persons a where exists (select distinct id, lastname from uplatz.users b where a.id = b.id and a.lastname = b.name);
```

SQL NOT EXISTS

The **NOT EXISTS** operator returns true if the subquery returns **no** row. Otherwise, it returns false. Note that the **NOT EXISTS** operator returns false if the subquery returns any rows with a NULL value.

```
select * from uplatz.persons a where NOT exists (select 1 from uplatz.users b where a.id = b.id);
```

```
select * from uplatz.persons a where not exists (select distinct id, lastname from uplatz.users b where a.id = b.id and a.lastname = b.name);
```

SQL Create table like

```
create table uplatz.demo like uplatz.persons;
```

SQL CASE Statement

select lastname, (case when age <30 then 'age<30' when age > 20 then 'age>20' end) as age_bucket from uplatz.persons;

Case expressions with reg expression:

select id, lastname, (case when lastname like '%hello%' then 'world' else 'not world' end) as lastname_updated from uplatz.persons;

SQL NULL FUNCTIONS

```
select IFNULL(NULL, 100);

select ISNULL(NULL);

 select ISNULL('HELLO’);

select COALESCE(NULL, 'hello world’);
```

The **IFNULL** function takes two arguments and returns the first argument if it is not NULL, otherwise, it returns the second argument. The **IFNULL** function works great with two arguments whereas the **COALESCE** function works with n arguments. In case the number of arguments is two, both functions are the same.

SQL String functions

SELECT CHAR_LENGTH("uplatz") AS LengthOfString;

SELECT LENG

LENGTH() returns the **length** of the string measured in bytes. **CHAR_LENGTH**() returns the **length** of the string measured in characters. TH("uplatz") AS LengthOfString;

However, if we change the database column to store the data as unicode: Then you will see the difference.

select * from uplatz.course_name;

select length(course), char_length(course) from uplatz.course_name;

Alter table uplatz.course_name modify column course varchar(255) Unicode;

select length(course), char_length(course) from uplatz.course_name;

SQL CONCAT

The CONCAT() function adds two or more expressions together.

```
SELECT CONCAT("SQL ", "Tutorial ", "is ", "fun!") AS ConcatenatedString;
```

SQL Training
Presentation by Uplatz
Contact us: https://training.uplatz.com
Email: info@uplatz.com
Phone: +44 7836 212635
select * from uplatz.courses

SQL Repeat

The REPEAT() function repeats a string as many times as specified.

```
SELECT REPEAT("UPLATZ SQL ", 3);
SELECT REPEAT("UPLATZ SQL ", 0);
```

SQL Replace

The REPLACE() function replaces all occurrences of a substring within a string, with a new substring.

```
SELECT REPLACE("UPLATZ SQL ", "SQL", “SQL IS THE BEST");
SELECT REPLACE("UPLATZ SQL SQL", "SQL", "IS THE BEST");
```

Note: This function performs a case-sensitive replacement.

```
SELECT REPLACE("UPLATZ SQL", "sql", "IS THE BEST");
```

SQL REVERSE

The REVERSE() function reverses a string and returns the result.

```
SELECT REVERSE("LQS ZTALPU");
```

SQL SPACE

The SPACE() function returns a string of the specified number of space characters.

```
mysql> SELECT SPACE(10);
SELECT CONCAT("UPLATZ", SPACE(10), "SQL");
SELECT CHAR_LENGTH(CONCAT("UPLATZ", "SQL"));
SELECT CHAR_LENGTH(CONCAT("UPLATZ", SPACE(10), "SQL"));
```

SQL STRCMP

The STRCMP() function compares two strings.

- If *string1* = *string2*, this function returns 0
- If *string1* < *string2*, this function returns -1
- If *string1* > *string2*, this function returns 1

```
SELECT STRCMP("SQL", “SQL");

SELECT STRCMP("SQL", "UPLATZ");

SELECT STRCMP("UPLATZSQLISTHEBEST", "UPLATZ");

SELECT STRCMP("A", "B");

SELECT STRCMP("B", "A");
```

Based on sort order.

SQL ABS

The ABS() function returns the absolute (positive) value of a number.

```
SELECT ABS(-2.345);
```

```
SELECT ABS(1234);
```

SQL CEIL

- The CEIL() function returns the smallest integer value that is bigger than or equal to a number.
- **Note:** This function is equal to the CEILING() function.

```
SELECT CEIL(23.1);
SELECT CEIL(23.78);
```

SQL FLOOR

- The FLOOR() function returns the largest integer value that is smaller than or equal to a number.
- **Note:** Also look at the ROUND(), and DIV functions.

```
SELECT FLOOR(25.75);
SELECT FLOOR(25.1);
```

SQL ROUND

The ROUND() function rounds a number to a specified number of decimal places.

```
SELECT ROUND(135.5);
SELECT ROUND(135.2);
```

SQL TRUNCATE

The TRUNCATE() function truncates a number to the specified number of decimal places.

```
SELECT TRUNCATE(135.375, 2);
SELECT TRUNCATE(135.999999, 2);
SELECT ROUND(135.999999, 2);
```

SQL EXP

- The EXP() function returns *e* raised to the power of the specified number.
- The constant *e* (2.718281...), is the base of natural logarithms.
- **Tip:** Also look at the LOG() and LN() functions.

```
SELECT EXP(1);
```

Why joins are used in SQL?

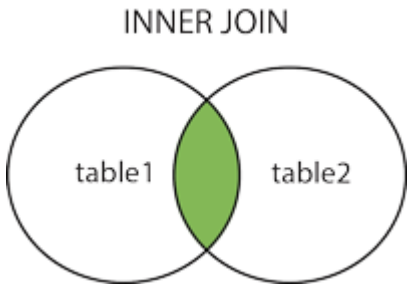
The **SQL Joins** clause is **used** to combine records from two or more tables in a database. A JOIN is a means for combining fields from two tables by using values common to each.

Types of Join:

- Inner Join
- Left Join
- Right Join
- Full Join
- Self Join

Inner Join

- The INNER JOIN keyword selects records that have matching values in both tables.

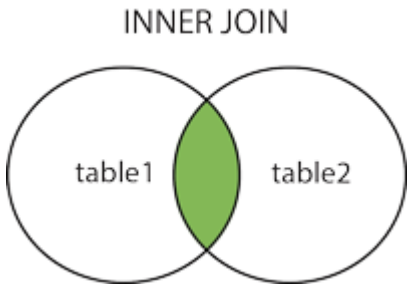


Note: The INNER JOIN keyword selects all rows from both tables as long as there is a match between the columns.

We can also do Inner join of three tables

Full Join

- The INNER JOIN keyword selects records that have matching values in both tables.



Note: The INNER JOIN keyword selects all rows from both tables as long as there is a match between the columns.

We can also do Inner join of three tables

Procedures

- A stored procedure is a prepared SQL code that you can save, so the code can be reused over and over again.
- So if you have an SQL query that you write over and over again, save it as a stored procedure, and then just call it to execute it.
- You can also pass parameters to a stored procedure, so that the stored procedure can act based on the parameter value(s) that is passed.

create procedure uplatz.proc_demo_test as select * from uplatz.users $$;

call uplatz.proc_demo_test();

MYSQL Workbench:

DELIMITER // CREATE PROCEDURE uplatz.param_proc_demo_test(IN ref_id int) BEGIN SELECT * FROM uplatz.users where id = ref_id; END // DELIMITER ;

call uplatz.param_proc_demo_test(5)

View

- In SQL, a view is a virtual table based on the result-set of an SQL statement.
- A view contains rows and columns, just like a real table. The fields in a view are fields from one or more real tables in the database.
- You can add SQL functions, WHERE, and JOIN statements to a view and present the data as if the data were coming from one single table.

```
Create view uplatz.view_demo as select * from uplatz.users;
```