**UMER HASSAN KHAN**

**SQL Server Built In Functions**

**ASCII**

Returns the ASCII value for the specific character

[**CHAR**](https://www.w3schools.com/sql/func_sqlserver_char.asp)

Returns the character based on the ASCII code

[**CHARINDEX**](https://www.w3schools.com/sql/func_sqlserver_charindex.asp)

Returns the position of a substring in a string

[**CONCAT**](https://www.w3schools.com/sql/func_sqlserver_concat.asp)

Adds two or more strings together

[**Concat with +**](https://www.w3schools.com/sql/func_sqlserver_concat_with_plus.asp)

Adds two or more strings together

[**CONCAT\_WS**](https://www.w3schools.com/sql/func_sqlserver_concat_ws.asp)

Adds two or more strings together with a separator

[**DATALENGTH**](https://www.w3schools.com/sql/func_sqlserver_datalength.asp)

Returns the number of bytes used to represent an expression

[**DIFFERENCE**](https://www.w3schools.com/sql/func_sqlserver_difference.asp)

Compares two SOUNDEX values, and returns an integer value

[**FORMAT**](https://www.w3schools.com/sql/func_sqlserver_format.asp)

Formats a value with the specified format

[**LEN**](https://www.w3schools.com/sql/func_sqlserver_len.asp)

Returns the length of a string

[**LOWER**](https://www.w3schools.com/sql/func_sqlserver_lower.asp)

Converts a string to lower-case

[**QUOTENAME**](https://www.w3schools.com/sql/func_sqlserver_quotename.asp)

Returns a Unicode string with delimiters added to make the string a valid SQL Server delimited identifier

[**REPLACE**](https://www.w3schools.com/sql/func_sqlserver_replace.asp)

Replaces all occurrences of a substring within a string, with a new substring

[**REPLICATE**](https://www.w3schools.com/sql/func_sqlserver_replicate.asp)

Repeats a string a specified number of times

[**REVERSE**](https://www.w3schools.com/sql/func_sqlserver_reverse.asp)

Reverses a string and returns the result

[**RIGHT**](https://www.w3schools.com/sql/func_sqlserver_right.asp)

Extracts a number of characters from a string (starting from right)

[**SOUNDEX**](https://www.w3schools.com/sql/func_sqlserver_soundex.asp)

Returns a four-character code to evaluate the similarity of two strings

[**SPACE**](https://www.w3schools.com/sql/func_sqlserver_space.asp)

Returns a string of the specified number of space characters

[**STR**](https://www.w3schools.com/sql/func_sqlserver_str.asp)

Returns a number as string

[**STUFF**](https://www.w3schools.com/sql/func_sqlserver_stuff.asp)

Deletes a part of a string and then inserts another part into the string, starting at a specified position

[**SUBSTRING**](https://www.w3schools.com/sql/func_sqlserver_substring.asp)

Extracts some characters from a string

[**UNICODE**](https://www.w3schools.com/sql/func_sqlserver_unicode.asp)

Returns the Unicode value for the first character of the input expression

[**UPPER**](https://www.w3schools.com/sql/func_sqlserver_upper.asp)

Converts a string to upper-case

[**ABS**](https://www.w3schools.com/sql/func_sqlserver_abs.asp)

Returns the absolute value of a number

[**ACOS**](https://www.w3schools.com/sql/func_sqlserver_acos.asp)

Returns the arc cosine of a number

[**ASIN**](https://www.w3schools.com/sql/func_sqlserver_asin.asp)

Returns the arc sine of a number

[**ATAN**](https://www.w3schools.com/sql/func_sqlserver_atan.asp)

Returns the arc tangent of a number

[**ATN2**](https://www.w3schools.com/sql/func_sqlserver_atn2.asp)

Returns the arc tangent of two numbers

[**AVG**](https://www.w3schools.com/sql/func_sqlserver_avg.asp)

Returns the average value of an expression

[**CEILING**](https://www.w3schools.com/sql/func_sqlserver_ceiling.asp)

Returns the smallest integer value that is >= a number

[**COUNT**](https://www.w3schools.com/sql/func_sqlserver_count.asp)

Returns the number of records returned by a select query

[**COS**](https://www.w3schools.com/sql/func_sqlserver_cos.asp)

Returns the cosine of a number

[**COT**](https://www.w3schools.com/sql/func_sqlserver_cot.asp)

Returns the cotangent of a number

[**DEGREES**](https://www.w3schools.com/sql/func_sqlserver_degrees.asp)

Converts a value in radians to degrees

[**EXP**](https://www.w3schools.com/sql/func_sqlserver_exp.asp)

Returns e raised to the power of a specified number

[**FLOOR**](https://www.w3schools.com/sql/func_sqlserver_floor.asp)

Returns the largest integer value that is <= to a number

[**LOG**](https://www.w3schools.com/sql/func_sqlserver_log.asp)

Returns the natural logarithm of a number, or the logarithm of a number to a specified base

[**LOG10**](https://www.w3schools.com/sql/func_sqlserver_log10.asp)

Returns the natural logarithm of a number to base 10

[**MAX**](https://www.w3schools.com/sql/func_sqlserver_max.asp)

Returns the maximum value in a set of values

[**MIN**](https://www.w3schools.com/sql/func_sqlserver_min.asp)

Returns the minimum value in a set of values

[**PI**](https://www.w3schools.com/sql/func_sqlserver_pi.asp)

Returns the value of PI

[**POWER**](https://www.w3schools.com/sql/func_sqlserver_power.asp)

Returns the value of a number raised to the power of another number

[**RADIANS**](https://www.w3schools.com/sql/func_sqlserver_radians.asp)

Converts a degree value into radians

[**RAND**](https://www.w3schools.com/sql/func_sqlserver_rand.asp)

Returns a random number

[**ROUND**](https://www.w3schools.com/sql/func_sqlserver_round.asp)

Rounds a number to a specified number of decimal places

[**SQUARE**](https://www.w3schools.com/sql/func_sqlserver_square.asp)

Returns the square of a number

[**SUM**](https://www.w3schools.com/sql/func_sqlserver_sum.asp)

Calculates the sum of a set of values

[**TAN**](https://www.w3schools.com/sql/func_sqlserver_tan.asp)

Returns the tangent of a number

[**CURRENT\_TIMESTAMP**](https://www.w3schools.com/sql/func_sqlserver_current_timestamp.asp)

Returns the current date and time

[**DATEADD**](https://www.w3schools.com/sql/func_sqlserver_dateadd.asp)

Adds a time/date interval to a date and then returns the date

[**DATEDIFF**](https://www.w3schools.com/sql/func_sqlserver_datediff.asp)

Returns the difference between two dates

[**DATENAME**](https://www.w3schools.com/sql/func_sqlserver_datename.asp)

Returns a specified part of a date (as string)

[**DATEPART**](https://www.w3schools.com/sql/func_sqlserver_datepart.asp)

Returns a specified part of a date (as integer)

[**DAY**](https://www.w3schools.com/sql/func_sqlserver_day.asp)

Returns the day of the month for a specified date

[**GETDATE**](https://www.w3schools.com/sql/func_sqlserver_getdate.asp)

Returns the current database system date and time

[**GETUTCDATE**](https://www.w3schools.com/sql/func_sqlserver_getutcdate.asp)

Returns the current database system UTC date and time

[**ISDATE**](https://www.w3schools.com/sql/func_sqlserver_isdate.asp)

Checks an expression and returns 1 if it is a valid date, otherwise 0

[**MONTH**](https://www.w3schools.com/sql/func_sqlserver_month.asp)

Returns the month part for a specified date (a number from 1 to 12)

[**SYSDATETIME**](https://www.w3schools.com/sql/func_sqlserver_sysdatetime.asp)

Returns the date and time of the SQL Server

[**YEAR**](https://www.w3schools.com/sql/func_sqlserver_year.asp)

Returns the year part for a specified date

**SQL Server Advanced Functions**

[**CAST**](https://www.w3schools.com/sql/func_sqlserver_cast.asp)

Converts a value (of any type) into a specified datatype

[**COALESCE**](https://www.w3schools.com/sql/func_sqlserver_coalesce.asp)

Returns the first non-null value in a list

[**CONVERT**](https://www.w3schools.com/sql/func_sqlserver_convert.asp)

Converts a value (of any type) into a specified datatype

[**CURRENT\_USER**](https://www.w3schools.com/sql/func_sqlserver_current_user.asp)

Returns the name of the current user in the SQL Server database

[**IIF**](https://www.w3schools.com/sql/func_sqlserver_iif.asp)

Returns a value if a condition is TRUE, or another value if a condition is FALSE

[**ISNULL**](https://www.w3schools.com/sql/func_sqlserver_isnull.asp)

Return a specified value if the expression is NULL, otherwise return the expression

[**ISNUMERIC**](https://www.w3schools.com/sql/func_sqlserver_isnumeric.asp)

Tests whether an expression is numeric

[**NULLIF**](https://www.w3schools.com/sql/func_sqlserver_nullif.asp)

Returns NULL if two expressions are equal

[**SESSION\_USER**](https://www.w3schools.com/sql/func_sqlserver_session_user.asp)

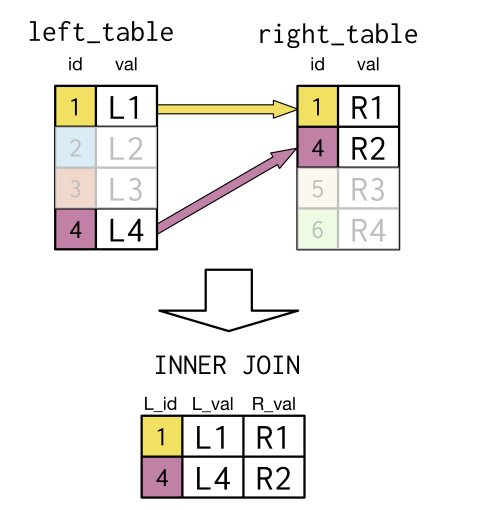
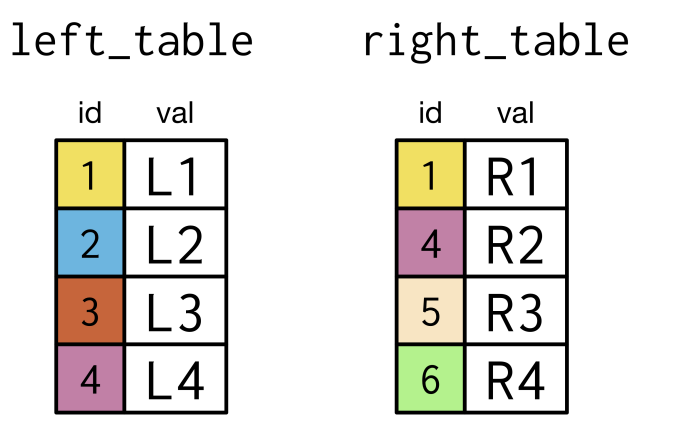
Returns the name of the current user in the SQL Server database

**JOINS**

**Inner Join**

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

****

****

SELECT p1.id AS L\_id, p1.val AS L\_val, p2.val AS R\_val

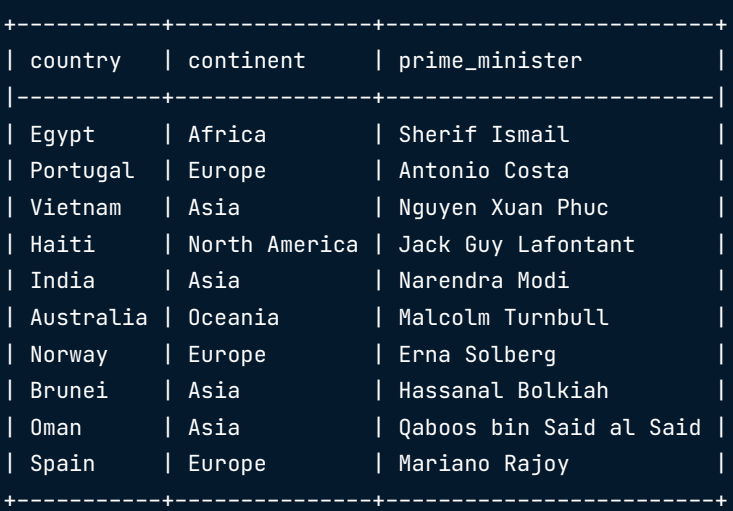
FROM left\_table AS p1

INNER JOIN right\_table AS p2

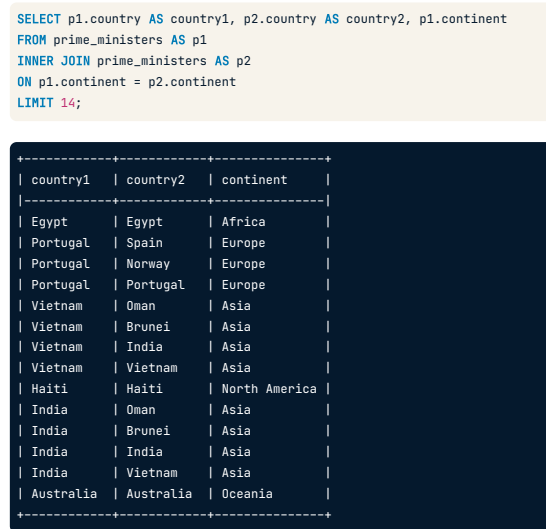
ON p1.id = p2.id;

**SELF JOIN**

A self join is a regular join, but the table is joined with itself.

****

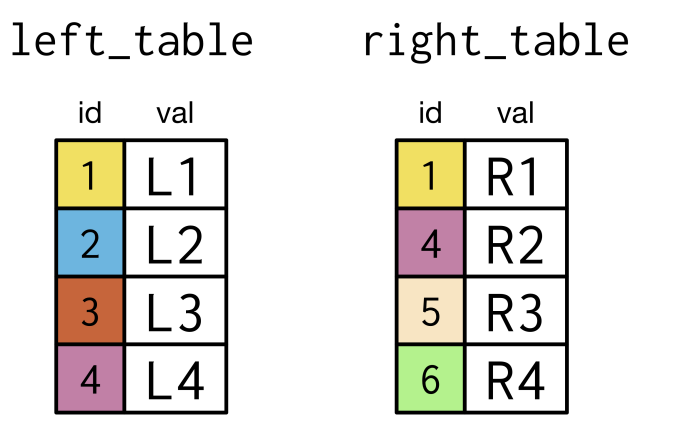
**SELF JOIN EXAMPLE**

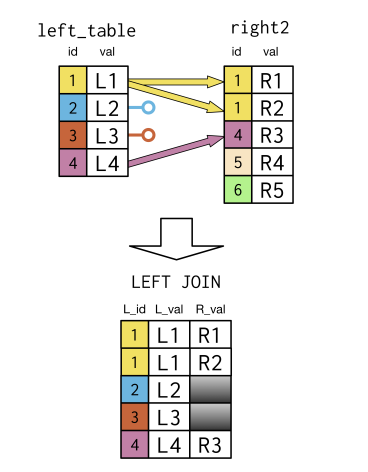
****

**Left Join**

The LEFT JOIN command returns all rows from the left table, and the matching rows from the right table. The result is NULL from the right side, if there is no match.

****

****

****

SELECT p1.id AS L\_id, p1.val AS L\_val, p2.val AS R\_val

FROM left\_table AS p1

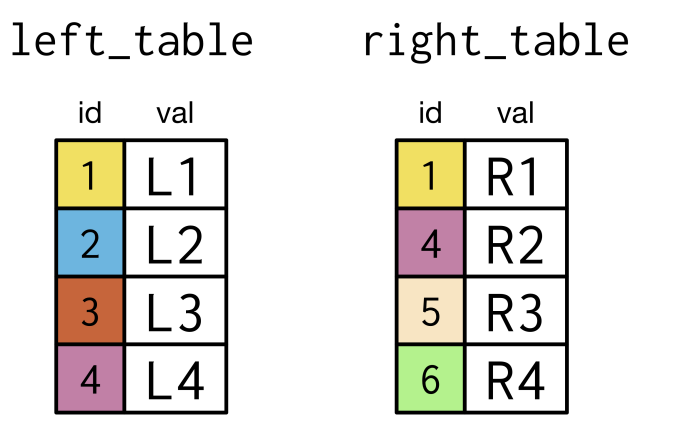
LEFT JOIN right\_table AS p2

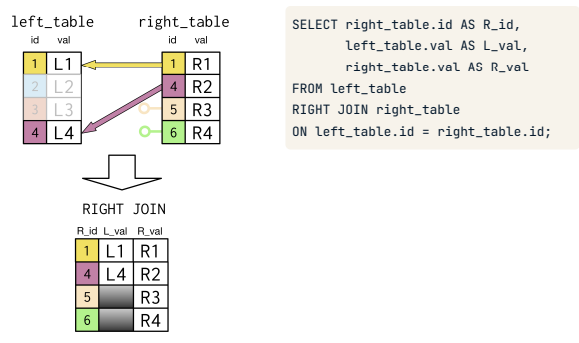
ON p1.id = p2.id;

**Right Join**

The RIGHT JOIN keyword returns all records from the right table (table2), and the matching records from the left table (table1). The result is 0 records from the left side, if there is no match.

****

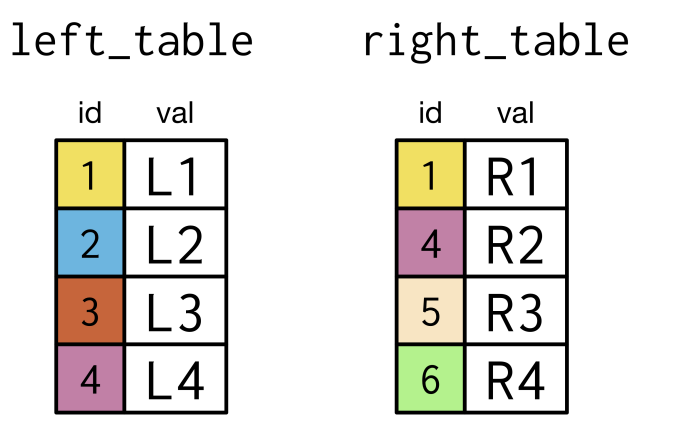
****

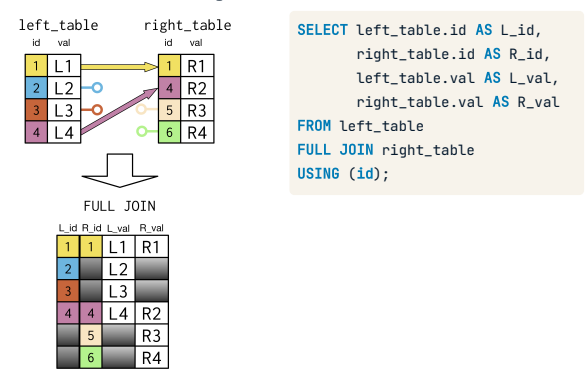
****

**Full Join**

The FULL OUTER JOIN keyword returns all records when there is a match in left (table1) or right (table2) table records.

****

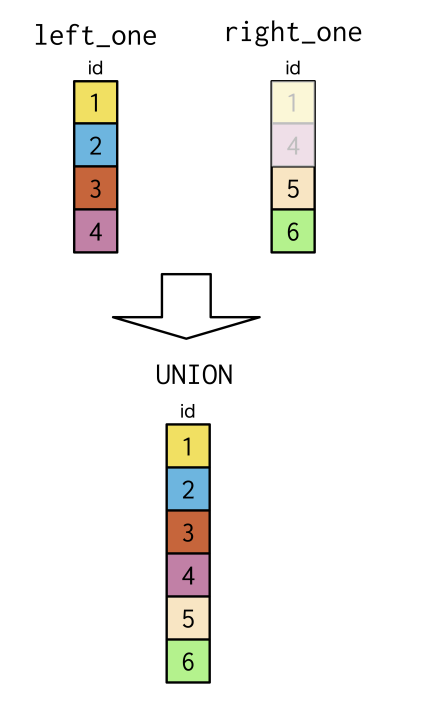
****

****

**SQL unions**

The UNION operator is used to combine the result-set of two or more SELECT statements.

* Every SELECT statement within UNION must have the same number of columns
* The columns must also have similar data types
* The columns in every SELECT statement must also be in the same order

****

SELECT id

FROM left\_one AS p1

UNION

SELECT id

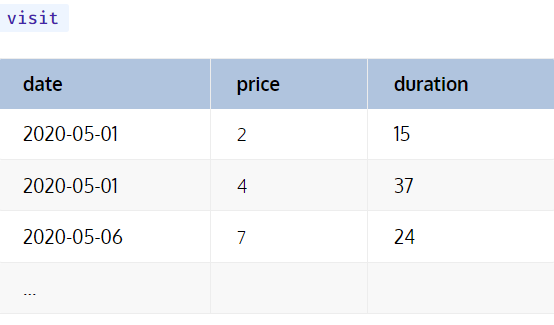
FROM right\_one AS p2

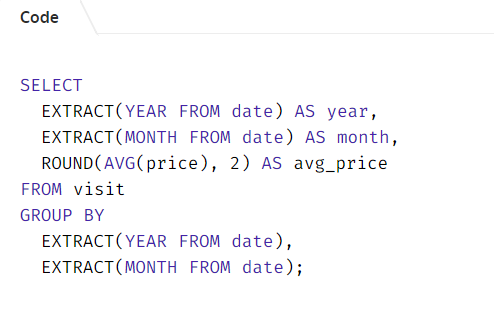
ORDER BY id

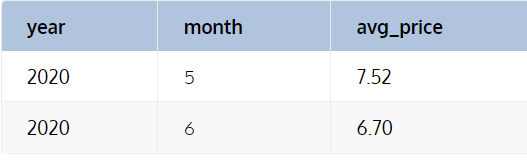
**GROUP BY STATEMENT**

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.

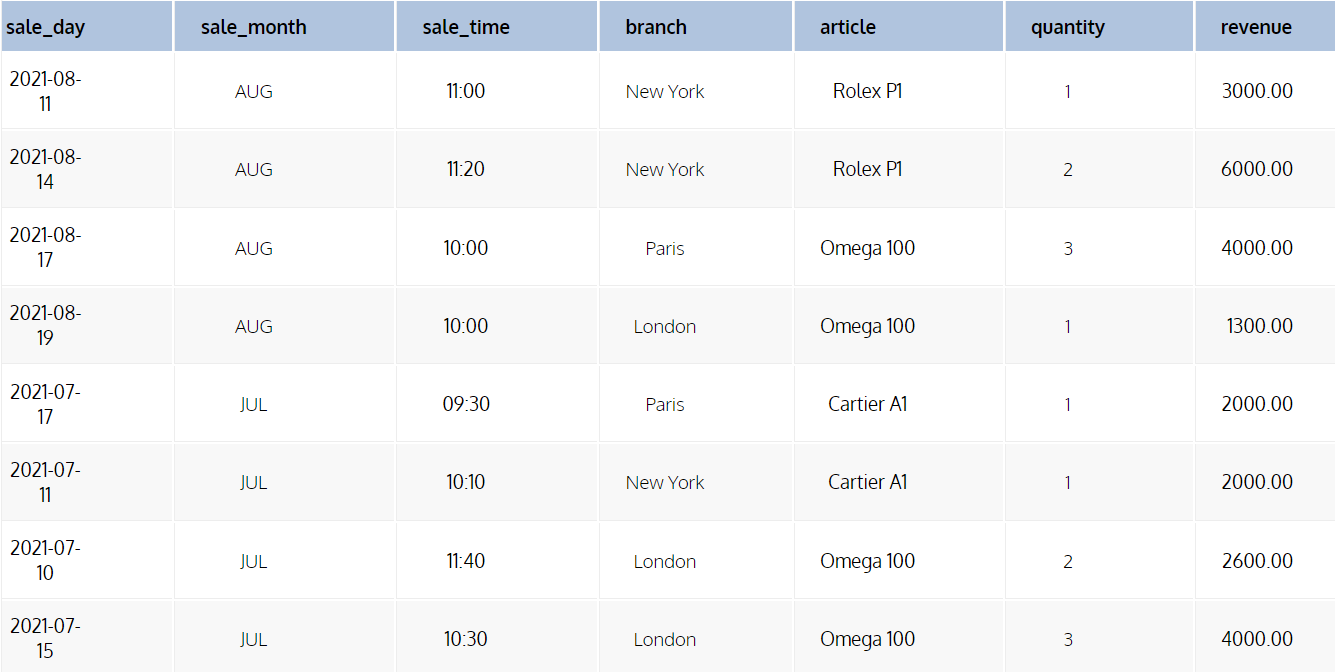


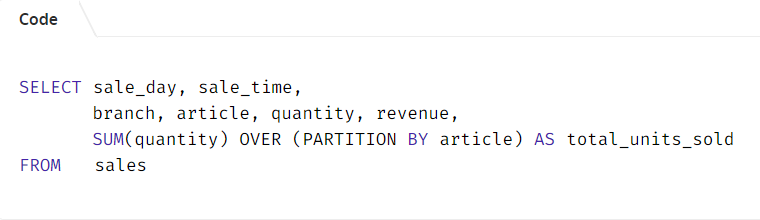
**

**

**OVER()**

The over clause combined with PARTITION BY is used to break up data into partitions. OVER clause defines a window or user-specified set of rows within a query result set.







**DDL (Data Definition Language):**

DDL or Data Definition Language is a set of SQL commands used to create, modify, and delete database structures but not data. These commands are normally not used by a general user, who should be accessing the database via an application.

**List of DDL commands:**

**CREATE:** This command is used to create the database or its objects (like table, index, function, views, store procedure, and triggers).

**DROP:** This command is used to delete objects from the database.

**ALTER:** This is used to alter the structure of the database.

**TRUNCATE:** This is used to remove all records from a table, including all spaces allocated for the records are removed.

**COMMENT:** This is used to add comments to the data dictionary.

**RENAME:** This is used to rename an object existing in the database.

**DQL (Data Query Language):**

We can define DQL as follows it is a component of SQL statement that allows getting data from the database and imposing order upon it. It includes the SELECT statement. This command allows getting the data out of the database to perform operations with it. When a SELECT is fired against a table or tables the result is compiled into a further temporary table, which is displayed or perhaps received by the program i.e. a front-end.

**List of DQL commands:**

**SELECT:** It is used to retrieve data from the database.

**DML (Data Manipulation Language):**

The SQL commands that deals with the manipulation of data present in the database belong to DML or Data Manipulation Language and this includes most of the SQL statements. It is the component of the SQL statement that controls access to data and to the database.

**List of DML commands:**

**INSERT:** It is used to insert data into a table.

**UPDATE:** It is used to update existing data within a table.

**DELETE:** It is used to delete records from a database table.

**LOCK:** Table control concurrency.

**CALL:** Call a PL/SQL or JAVA subprogram.

**EXPLAIN PLAN:** It describes the access path to data.

**Subqueries**

A subquery is a SQL query nested inside a larger query.

* A subquery may occur in :
  + - A SELECT clause
  + - A FROM clause
  + - A WHERE clause



**Views:**

A view is nothing more than a SQL statement that is stored in the database with an associated name. A view is actually a composition of a table in the form of a predefined SQL query.

A view can contain all rows of a table or select rows from a table. A view can be created from one or many tables which depends on the written SQL query to create a view.

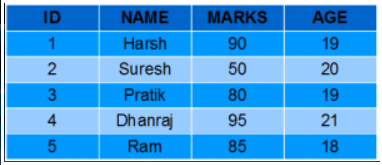
Views, which are a type of virtual tables allow users to do the following −

* Structure data in a way that users or classes of users find natural or intuitive.
* Restrict access to the data in such a way that a user can see and (sometimes) modify exactly what they need and no more.
* Summarize data from various tables which can be used to generate reports.

StudentDetails



StudentMarks





**Syntax for deleting views:**

DROP VIEW view\_name;

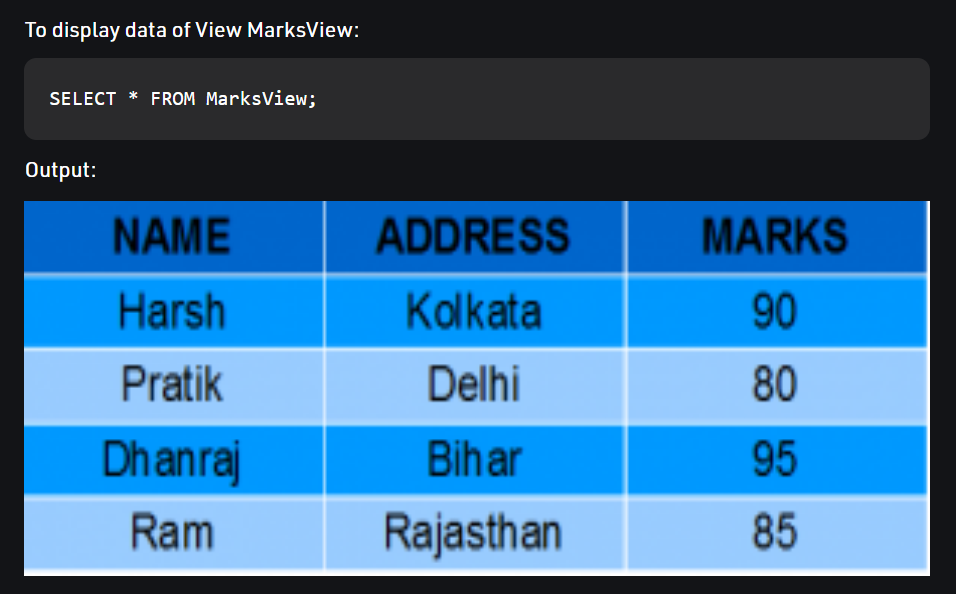
**Syntax for updating views:**

CREATE OR REPLACE VIEW MarksView AS

SELECT StudentDetails.NAME, StudentDetails.ADDRESS, StudentMarks.MARKS, StudentMarks.AGE

FROM StudentDetails, StudentMarks

WHERE StudentDetails.NAME = StudentMarks.NAME;

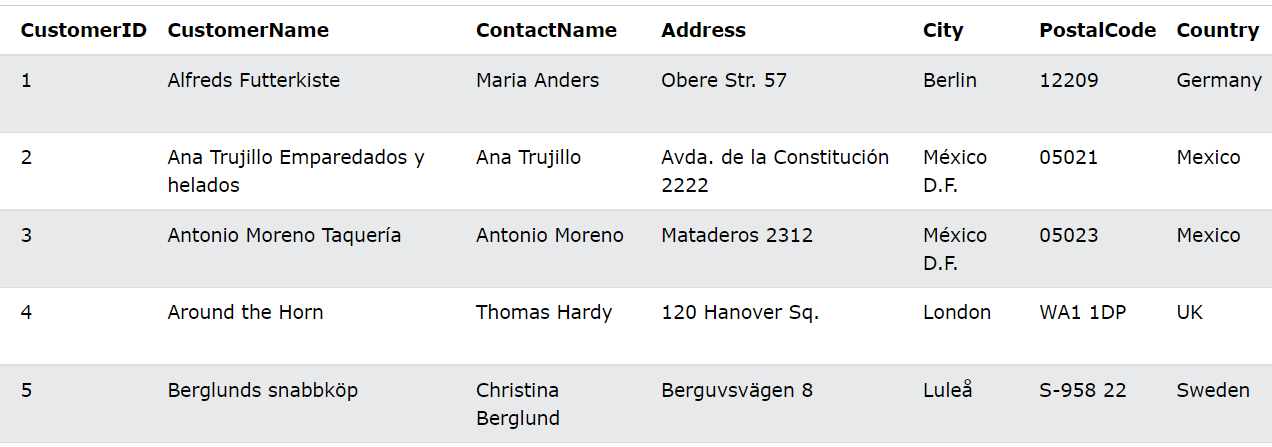


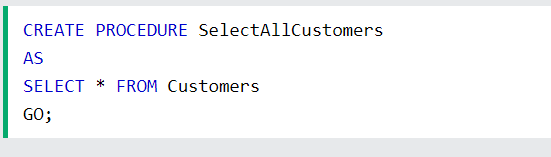
**Stored Procedure**

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.





Stored Procedure With Multiple Parameters

