

## SQL Cheat Sheet: FUNCTIONS and Implicit JOIN

Command	Syntax (MySQL/DB2)	Description	Example (MySQL/DB2)
COUNT	<code>SELECT COUNT(column_name) FROM table_name WHERE condition;</code>	<code>COUNT</code> function returns the number of rows that match a specified criterion.	<code>SELECT COUNT(dep_id) FROM employees;</code>
AVG	<code>SELECT AVG(column_name) FROM table_name WHERE condition;</code>	<code>AVG</code> function returns the average value of a numeric column.	<code>SELECT AVG(salary) FROM employees;</code>
SUM	<code>SELECT SUM(column_name) FROM table_name WHERE condition;</code>	<code>SUM</code> function returns the total sum of a numeric column.	<code>SELECT SUM(salary) FROM employees;</code>
MIN	<code>SELECT MIN(column_name) FROM table_name WHERE condition;</code>	<code>MIN</code> function returns the smallest value of the SELECTED column.	<code>SELECT MIN(salary) FROM employees;</code>
MAX	<code>SELECT MAX(column_name) FROM table_name WHERE condition;</code>	<code>MAX</code> function returns the largest value of the SELECTED column.	<code>SELECT MAX(salary) FROM employees;</code>
ROUND	<code>SELECT ROUND(2number, decimals, operation) AS RoundValue;</code>	<code>ROUND</code> function rounds a number to a specified number of decimal places.	<code>SELECT ROUND(salary) FROM employees;</code>
LENGTH	<code>SELECT LENGTH(column_name) FROM table;</code>	<code>LENGTH</code> function returns the length of a string (in bytes).	<code>SELECT LENGTH(f_name) FROM employees;</code>
UCASE	<code>SELECT UCASE(column_name) FROM table;</code>	<code>UCASE</code> function displays the column name in each table in uppercase.	<code>SELECT UCASE(f_name) FROM employees;</code>
LCASE	<code>SELECT LCASE(column_name) FROM table;</code>	<code>LCASE</code> function displays the column name in each table in lowercase.	<code>SELECT LCASE(f_name) FROM employees;</code>
DISTINCT	<code>SELECT DISTINCT column_name FROM table;</code>	<code>DISTINCT</code> function is used to display data without duplicates.	<code>SELECT DISTINCT UCASE(f_name) FROM employees;</code>
DAY	<code>SELECT DAY(column_name) FROM table</code>	<code>DAY</code> function returns the day of the month for a given date.	<code>SELECT DAY(b_date) FROM employees where emp_id = 'E1002';</code>
CURRENT_DATE	<code>SELECT CURRENT_DATE;</code>	<code>CURRENT_DATE</code> is used to display the current date.	<code>SELECT CURRENT_DATE;</code>
DATEDIFF()	<code>SELECT DATEDIFF(date1, date2);</code>	<code>DATEDIFF()</code> is used to calculate the difference between two dates or time stamps. The default value generated is the difference in number of days.	<code>SELECT DATEDIFF(CURRENT_DATE, date_column) FROM table;</code>
FROM_DAYS()	<code>SELECT FROM_DAYS(number_of_days);</code>	<code>FROM_DAYS()</code> is used to convert a given number of days to YYYY-MM-DD format.	<code>SELECT FROM_DAYS(DATEDIFF(CURRENT_DATE, date_column)) FROM table;</code>
DATE_ADD()	<code>SELECT DATE_ADD(date, INTERVAL n type);</code>	<code>DATE_ADD()</code> is used to calculate the date after lapse of mentioned number of units of date type, i.e. if n=3 and type=DAY, the result is a date 3 days after what is mentioned in date column. The type valiable can also be months or years.	<code>SELECT DATE_ADD(date, INTERVAL 3 DAY);;</code>
DATE_SUB()	<code>SELECT DATE_SUB(date, INTERVAL n type);</code>	<code>DATE_SUB()</code> is used to calculate the date prior to the record date by mentioned number of units of date type, i.e. if n=3 and type=DAY, the result is a date 3 days before what is mentioned in date column. The type valiable can also be months or years.	<code>SELECT DATE_SUB(date, INTERVAL 3 DAY);;</code>
Subquery	<code>SELECT column_name [, column_name ] FROM table1 [, table2 ] WHERE column_name OPERATOR (SELECT column_name [, column_name ] FROM table1 [, table2 ] [WHERE])</code>	<code>Subquery</code> is a query within another SQL query and embedded within the WHERE clause. A subquery is used to return data that will be used in the main query as a condition to further restrict the data to be retrieved.	<code>SELECT emp_id, f_name, l_name, salary FROM employees where salary &lt; (SELECT AVG(salary) FROM employees);</code>  <code>SELECT * FROM ( SELECT emp_id, f_name, l_name, dep_id FROM employees) AS emp4all;</code>  <code>SELECT * FROM employees WHERE job_id IN (SELECT job_ident FROM jobs);</code>
Implicit Inner Join	<code>SELECT column_name(s) FROM table1, table2 WHERE table1.column_name = table2.column_name;</code>	<code>Implicit Inner Join</code> combines two or more records but displays only matching values in both tables. Inner join applies only the specified columns.	<code>SELECT * FROM employees, jobs where employees.job_id = jobs.job_ident;</code>
Implicit Cross Join	<code>SELECT column_name(s) FROM table1, table2;</code>	<code>Implicit Cross Join</code> is defined as a Cartesian product where the number of rows in the first table is multiplied by the number of rows in the second table.	<code>SELECT * FROM employees, jobs;</code>

### Author(s)

Lakshmi Holla  
Abhishek Gagneja