SQL COMMANDS
CHEAT SHEET

SQL Commands

The commands in SQL are called Queries and they are of two types:

- Data Definition Query: The statements which defines the structure of a database, create tables, specify their keys, indexes and so on
- Data manipulation queries: These are the queries which can be edited.

  E.g.: Select, update and insert operation

Command	Syntax	Description
ALTER table	ALTER TABLE table_name ADD column_name datatype;	It is used to add columns to a table in a database
AND	SELECT column_name(s) FROM table_name WHERE column_1 = value_1 AND column_2 = value_2;	It is an operator that is used to combine two conditions
AS	SELECT column_name AS 'Alias' FROM table_name;	It is an keyword in SQL that is used to rename a column or table using an alias name
BETWEEN	SELECT column_name(s) FROM table_name WHERE column_name BETWEEN value_1 AND value_2;	It is an operator used to filter the result within a certain range
CASE	SELECT column_name, CASE WHEN condition THEN 'Result_1' WHEN condition THEN 'Result_2' ELSE 'Result_3' END FROM table_name;	It is a statement used to create different outputs inside a SELECT statement
COUNT	SELECT COUNT(column_name) FROM table_name;	It is a function that takes the name of a column as argument and counts the number of rows when the column is not NULL
Create TABLE	CREATE TABLE table_name (column_1 datatype, column_2 datatype, column_3 datatype);	It is used to create a new table in a database and specify the name of the table and columns inside it

Command	Syntax	Description
GROUP BY	SELECT column_name, COUNT(*) FROM table_name GROUP BY column_name;	It is an clause in SQL used for aggregate functions in collaboration with the SELECT statement
HAVING	SELECT column_name, COUNT(*) FROM table_name GROUP BY column_name HAVING COUNT(*) > value;	It is used in SQL because the WHERE keyword cannot be used in aggregating functions
INNER JOIN	SELECT column_name(s) FROM table_1 JOIN table_2 ON table_1.column_name = table_2.column_name;	It is used to combine rows from different tables if the Join condition goes TRUE
INSERT	INSERT INTO table_name (column_1, column_2, column_3) VALUES (value_1, 'value_2', value_3);	It is used to add new rows to a table
IS NULL/ IS NOT NULL	SELECT column_name(s) FROM table_name WHERE column_name IS NULL;	It is a operator used with the WHERE clause to check for the empty values
LIKE	SELECT column_name(s) FROM table_name WHERE column_name LIKE pattern;	It is an special operator used with the WHERE clause to search for a specific pattern in a column
LIMIT	SELECT column_name(s) FROM table_name LIMIT number;	It is a clause to specify the maximum number of rows the result set must have
MAX	SELECT MAX(column_name) FROM table_name;	It is a function that takes number of columns as an argument and return the largest value among them
MIN	SELECT MIN(column_name) FROM table_name;	It is a function that takes number of columns as an argument and return the smallest value among them
OR	SELECT column_name FROM table_name WHERE column_name = value_1 OR column_name = value_2;	It is an operator that is used to filter the result set to contain only the rows where either condition is TRUE
ORDER BY	SELECT column_name FROM table_name ORDER BY column_name ASC | DESC;	It is a clause used to sort the result set by a particular column either numerically or alphabetically

Command	Syntax	Description
OUTER JOIN	SELECT column_name(s) FROM table_1 LEFT JOIN table_2 ON table_1.column_name = table_2.column_name;	It is sued to combine rows from different tables even if the condition is NOT TRUE
ROUND	SELECT ROUND(column_name, integer) FROM table_name;	It is a function that takes the column name and a integer as an argument, and rounds the values in a column to the number of decimal places specified by an integer
SELECT	SELECT column_name FROM table_name;	It is a statement that is used to fetch data from a database
SELECT DISTINCT	SELECT DISTINCT column_name FROM table_name;	It is used to specify that the statement is a query which returns unique values in specified columns
SUM	SELECT SUM(column_name) FROM table_name;	It is function used to return sum of values from a particular column
UPDATE	UPDATE table_name SET some_column = some_value WHERE some_column = some_value;	It is used to edit rows in a table
WHERE	SELECT column_name(s) FROM table_name WHERE column_name operator value;	It is a clause used to filter the result set to include the rows where the condition is TRUE
WITH	WITH temporary_name AS (SELECT * FROM table_name) SELECT * FROM temporary_name WHERE column_name operator value;	It is used to store the result of a particular query in a temporary table using an alias
DELETE	DELETE FROM table_name WHERE some_column = some_value;	It is used to remove the rows from a table
AVG	SELECT AVG(column_name) FROM table_name;	It is used to aggregate a numeric column and return its average

Commands and syntax for querying data from Single Table	Commands and syntax for querying data from Multiple Table
SELECT c1 FROM t To select the data in Column c1 from table t	SELECT c1, c2 FROM t1 INNER JOIN t2 on condition Select column c1 and c2 from table t1 and perform an inner join between t1 and t2
SELECT * FROM t To select all rows and columns from table t	SELECT c1, c2 FROM t1 LEFT JOIN t2 on condition Select column c1 and c2 from table t1 and perform a left join between t1 and t2
SELECT c1 FROM t WHERE c1 = 'test' To select data in column c1 from table t, where c1=test	SELECT c1, c2 FROM t1 RIGHT JOIN t2 on condition Select column c1 and c2 from table t1 and perform a right join between t1 and t2
SELECT c1 FROM t ORDER BY c1 ASC (DESC) To select data in column c1 from table t either in ascending or descending order	SELECT c1, c2 FROM t1 FULL OUTER JOIN t2 on condition Select column c1 and c2 from table t1 and perform a full outer join between t1 and t2
SELECT c1 FROM t ORDER BY c1LIMIT n OFFSET offset To skip the offset of rows and return the next n rows	SELECT c1, c2 FROM t1 CROSS JOIN t2 Select column c1 and c2 from table t1 and produce a Cartesian product of rows in a table
SELECT c1, aggregate(c2) FROM t GROUP BY c1 To group rows using an aggregate function	SELECT c1, c2 FROM t1, t2 Select column c1 and c2 from table t1 and produce a Cartesian product of rows in a table
SELECT c1, aggregate(c2) FROM t GROUP BY c1HAVING condition Group rows using an aggregate function and filter these groups using 'HAVING' clause	SELECT c1, c2 FROM t1 A INNER JOIN t2 B on condition Select column c1 and c2 from table t1 and join it to itself using INNER JOIN clause

FURTHERMORE:

SQL Developer, SQL DBA Training Masters Program