

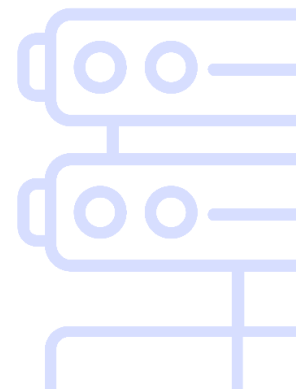


DataImpala



Basic SQL Cheatsheet

By Ahnaf Chowdhury



Basic SQL Cheatsheet

With Easy Explanations for Every Function

What is SQL?

SQL (Structured Query Language) is used to communicate with databases.

It lets you look up data, add new info, update old info, and more.

1. Viewing the Data – SELECT

```
SELECT column1, column2 FROM table_name;
```

Use it to get data from the table.

* means “all columns.”

Example:

```
SELECT * FROM employees;
```

→ Gets all the data from the employees table.

2. Filtering – WHERE

```
SELECT * FROM table_name WHERE condition;
```

Use it to get only rows that meet a condition.

Example:

```
SELECT * FROM employees WHERE age > 30;
```

3. Sorting – ORDER BY

```
SELECT * FROM table_name ORDER BY column_name ASC|DESC;
```

ASC = ascending (small to big)

DESC = descending (big to small)

Example:

```
SELECT * FROM employees ORDER BY salary DESC;
```

4. Picking Specific Rows – LIMIT

```
SELECT * FROM table_name LIMIT number;
```

Shows only a limited number of results.

Example:

```
SELECT * FROM employees LIMIT 5;
```

→ Only shows the first 5 rows.

5. Wildcards & Pattern Matching – LIKE

```
SELECT * FROM table_name WHERE column LIKE 'pattern';
```

% = any number of characters

_ = one character

Example:

```
SELECT * FROM customers WHERE name LIKE 'A%';
```

→ Finds names starting with "A".

6. Math on Columns – COUNT, SUM, AVG, MIN, MAX

COUNT(column) = How many entries

SUM(column) = Total of all values

AVG(column) = Average value

MIN(column) = Lowest value

MAX(column) = Highest value

Example:

```
SELECT COUNT(*) FROM orders;
```

7. Grouping Data – GROUP BY

```
SELECT column, COUNT(*) FROM table_name GROUP BY column;
```

Groups data by a column and lets you do math on each group.

Example:

```
SELECT department, COUNT(*) FROM employees GROUP BY department;
```

🚫 8. Filter Groups – HAVING

```
SELECT column, COUNT(*) FROM table_name GROUP BY column HAVING  
COUNT(*) > 2;
```

Works like **WHERE**, but for groups created with **GROUP BY**.

+ 9. Add New Data – INSERT INTO

```
INSERT INTO table_name (column1, column2) VALUES (value1, value2);
```

Example:

```
INSERT INTO employees (name, age) VALUES ('Tom', 30);
```

✏️ 10. Change Existing Data – UPDATE

```
UPDATE table_name SET column1 = value1 WHERE condition;
```

Example:

```
UPDATE employees SET salary = 5000 WHERE name = 'Tom';
```

✖ 11. Remove Data – DELETE

```
DELETE FROM table_name WHERE condition;
```

Example:

```
DELETE FROM employees WHERE age < 20;
```

🏗 12. Make a New Table – CREATE TABLE

```
CREATE TABLE table_name (  
    column1 datatype,  
    column2 datatype  
);
```

Example:

```
CREATE TABLE students (  
    id INT,  
    name TEXT,  
    grade FLOAT  
);
```

🔧 13. Change Table Structure – ALTER TABLE

```
ALTER TABLE table_name ADD column_name datatype;
```

Example:

```
ALTER TABLE students ADD email TEXT;
```

14. Remove Table – DROP TABLE

```
DROP TABLE table_name;
```

This deletes the table and all its data.

15. Combine Tables – JOIN

Combine rows from two tables based on a common column.

```
SELECT a.column, b.column  
FROM table1 a  
JOIN table2 b ON a.common_column = b.common_column;
```

Types of JOINS:

INNER JOIN: Only matching rows

LEFT JOIN: All rows from left, matching from right

RIGHT JOIN: All rows from right, matching from left

FULL JOIN: All rows from both sides (if supported)

16. Check for Missing Data – IS NULL

```
SELECT * FROM employees WHERE email IS NULL;
```

17. Aliases – AS

```
SELECT column_name AS short_name FROM table_name;
```

Rename a column or table for display.

18. Subqueries

A query inside another query.

```
SELECT name FROM employees WHERE salary > (SELECT AVG(salary) FROM employees);
```

19. Basic Data Types (used when creating tables)

Type	Description
INT	Whole numbers
FLOAT	Decimal numbers
TEXT	Strings or letters
DATE	Calendar dates
BOOLEAN	True or false values

