

SQL CheatSheet

The Complete SQL Cheat Sheet

1. DDL Commands

Data Definition Language - Defining and optimizing database structure.

Command	Description	Syntax	Example
CREATE	The CREATE command creates a new database and objects, such as a table, index, view, or stored procedure.	<pre>CREATE TABLE table_name (column1 datatype1, column2 datatype2, ...);</pre>	<pre>CREATE TABLE employees (employee_id INT PRIMARY KEY, first_name VARCHAR(50), last_name VARCHAR(50), age INT);</pre>
ALTER	The ALTER command adds, deletes, or modifies columns in an existing table.	<pre>ALTER TABLE table_name ADD column_name datatype;</pre>	<pre>ALTER TABLE customers ADD email VARCHAR(100);</pre>
DROP	The DROP command is used to drop an existing table in a database.	<pre>DROP TABLE table_name;</pre>	<pre>DROP TABLE customers;</pre>
TRUNCATE	The TRUNCATE command is used to delete the data inside a table, but not the table itself.	<pre>TRUNCATE TABLE table_name;</pre>	<pre>TRUNCATE TABLE customers;</pre>

2. DML Commands

Data Manipulation Language Commands - Manipulating data rows.

Command	Description	Syntax	Example
SELECT	The SELECT command retrieves data from a database.	<pre>SELECT column1, column2 FROM table_name;</pre>	<pre>SELECT first_name, last_name FROM customers;</pre>

INSERT	The INSERT command adds new records to a table.	<pre>INSERT INTO table_name (column1, column2) VALUES (value1, value2);</pre>	<pre>INSERT INTO customers (first_name, last_name) VALUES ('Mary', 'Doe');</pre>
UPDATE	The UPDATE command is used to modify existing records in a table.	<pre>UPDATE table_name SET column1 = value1, column2 = value2 WHERE condition;</pre>	<pre>UPDATE employees SET employee_name = 'John Doe', department = 'Marketing';</pre>
DELETE	The DELETE command removes records from a table.	<pre>DELETE FROM table_name WHERE condition;</pre>	<pre>DELETE FROM employees WHERE employee_name = 'John Doe';</pre>

3. Querying Data Commands

Command	Description	Syntax	Example
SELECT Statement	The SELECT statement is the primary command used to retrieve data from a database	<pre>SELECT column1, column2 FROM table_name;</pre>	<pre>SELECT first_name, last_name FROM customers;</pre>
WHERE Clause	The WHERE clause is used to filter rows based on a specified condition.	<pre>SELECT * FROM table_name WHERE condition;</pre>	<pre>SELECT * FROM customers WHERE age > 30;</pre>
ORDER BY Clause	The ORDER BY clause is used to sort the result set in ascending or descending order based on a specified column.	<pre>SELECT * FROM table_name ORDER BY column_name ASC DESC;</pre>	<pre>SELECT * FROM products ORDER BY price DESC;</pre>
GROUP BY Clause	The GROUP BY clause groups rows based on the values in a specified column. It is often used with aggregate functions like COUNT, SUM, AVG, etc.	<pre>SELECT column_name, COUNT(*) FROM table_name GROUP BY column_name;</pre>	<pre>SELECT category, COUNT(*) FROM products GROUP BY category;</pre>
HAVING Clause	The HAVING clause filters grouped results based on a specified condition.	<pre>SELECT column_name, COUNT(*) FROM table_name GROUP BY column_name HAVING condition;</pre>	<pre>SELECT category, COUNT(*) FROM products GROUP BY category HAVING COUNT(*) > 5;</pre>

4. Aggregate Functions Commands

Command	Description	Syntax	Example
COUNT()	The COUNT command counts the number of rows or non-null values in a specified column.	<pre>SELECT COUNT(column_name) FROM table_name;</pre>	<pre>SELECT COUNT(age) FROM employees;</pre>
SUM()	The SUM command is used to calculate the sum of all values in a specified column.	<pre>SELECT SUM(column_name) FROM table_name;</pre>	<pre>SELECT SUM(revenue) FROM sales;</pre>
AVG()	The AVG command is used to calculate the average (mean) of all values in a specified column.	<pre>SELECT AVG(column_name) FROM table_name;</pre>	<pre>SELECT AVG(price) FROM products;</pre>
MIN()	The MIN command returns the minimum (lowest) value in a specified column.	<pre>SELECT MIN(column_name) FROM table_name;</pre>	<pre>SELECT MIN(price) FROM products;</pre>
MAX()	The MAX command returns the maximum (highest) value in a specified column.	<pre>SELECT MAX(column_name) FROM table_name;</pre>	<pre>SELECT MAX(price) FROM products;</pre>

5. Joining Commands

Command	Description	Syntax	Example
INNER JOIN	The INNER JOIN command returns rows with matching values in both tables.	<pre>SELECT * FROM table1 INNER JOIN table2 ON table1.column = table2.column;</pre>	<pre>SELECT * FROM employees INNER JOIN departments ON employees.department_id = departments.id;</pre>
LEFT JOIN/LEFT OUTER JOIN	The LEFT JOIN command returns all rows from the left table (first table) and the matching rows from the right	<pre>SELECT * FROM table1 LEFT JOIN table2 ON table1.column = table2.column;</pre>	<pre>SELECT * FROM employees LEFT JOIN departments ON employees.department_id = departments.id;</pre>

	table (second table).		
RIGHT JOIN/RIGHT OUTER JOIN	The RIGHT JOIN command returns all rows from the right table (second table) and the matching rows from the left table (first table).	<pre>SELECT * FROM table1 RIGHT JOIN table2 ON table1.column = table2.column;</pre>	<pre>SELECT * FROM employees RIGHT JOIN departments ON employees.department_id = departments.department_id;</pre>
FULL JOIN/FULL OUTER JOIN	The FULL JOIN command returns all rows when there is a match in either the left table or the right table.	<pre>SELECT * FROM table1 FULL JOIN table2 ON table1.column = table2.column;</pre>	<pre>SELECT * FROM employees LEFT JOIN departments ON employees.employee_id = departments.employee_id UNION SELECT * FROM employees RIGHT JOIN departments ON employees.employee_id = departments.employee_id;</pre>
CROSS JOIN	The CROSS JOIN command combines every row from the first table with every row from the second table, creating a Cartesian product.	<pre>SELECT * FROM table1 CROSS JOIN table2;</pre>	<pre>SELECT * FROM employees CROSS JOIN departments;</pre>
SELF JOIN	The SELF JOIN command joins a table with itself.	<pre>SELECT * FROM table1 t1, table1 t2 WHERE t1.column = t2.column;</pre>	<pre>SELECT * FROM employees t1, employees t2 WHERE t1.employee_id = t2.employee_id;</pre>
NATURAL JOIN	The NATURAL JOIN command matches columns with the same name in both tables.	<pre>SELECT * FROM table1 NATURAL JOIN table2;</pre>	<pre>SELECT * FROM employees NATURAL JOIN departments;</pre>

6. Subqueries in SQL

Command	Description	Syntax	Example
IN	The IN command is used to determine whether a	<pre>SELECT column(s) FROM table WHERE</pre>	<pre>SELECT * FROM customers WHERE city IN</pre>

	value matches any value in a subquery result. It is often used in the WHERE clause.	<code>value IN (subquery);</code>	<code>(SELECT city FROM suppliers);</code>
ANY	The ANY command is used to compare a value to any value returned by a subquery. It can be used with comparison operators like =, >, <, etc.	<code>SELECT column(s) FROM table WHERE value < ANY (subquery);</code>	<code>SELECT * FROM products WHERE price < ANY (SELECT unit_price FROM supplier_products);</code>
ALL	The ALL command is used to compare a value to all values returned by a subquery. It can be used with comparison operators like =, >, <, etc.	<code>SELECT column(s) FROM table WHERE value > ALL (subquery);</code>	<code>SELECT * FROM orders WHERE order_amount > ALL (SELECT total_amount FROM previous_orders);</code>

| *Keep Learning & Keep Exploring!*