

# Quick SQL Reference: Your Go-To Cheat Sheet (Clear and direct)

Brought to you by programmingvalley.com

## 1. Finding Data

- **SELECT:** Retrieve data from a database.
  - SELECT \* FROM table\_name;
  - SELECT column1, column2 FROM table\_name WHERE condition;
- **DISTINCT:** Return unique values.
  - SELECT DISTINCT column\_name FROM table\_name;
- **WHERE:** Filter rows based on conditions.
  - WHERE condition1 AND condition2
  - WHERE condition1 OR condition2
  - WHERE NOT condition
  - WHERE EXISTS (SELECT column\_name FROM table\_name WHERE condition)
- **ORDER BY:** Sort results (ASC/DESC).
  - ORDER BY column DESC;
- **SELECT TOP/LIMIT:** Limit the number of records returned.
  - SELECT TOP number columns FROM table\_name; (SQL Server)
  - SELECT columns FROM table\_name LIMIT count OFFSET offset; (MySQL)
- **LIKE:** Search for patterns using wildcards (% , \_).
  - WHERE column\_name LIKE 'a%'; (starts with 'a')
  - WHERE column\_name LIKE '%or%'; (contains 'or')
- **IN:** Specify multiple values in a WHERE clause.
  - WHERE column\_name IN (value1, value2);
  - WHERE column\_name IN (SELECT STATEMENT);
- **BETWEEN:** Select values within a range (inclusive).
  - WHERE column\_name BETWEEN value1 AND value2;
- **NULL:** Check for fields with no value.
  - WHERE column\_name IS NULL;
  - WHERE column\_name IS NOT NULL;
- **AS:** Assign temporary names (aliases) to columns or tables.
  - SELECT column\_name AS alias\_name FROM table\_name;
- **UNION / INTERSECT / EXCEPT:** Combine or compare result-sets of SELECT statements.
  - UNION: Combines and returns distinct values. Use UNION ALL for duplicates.
  - INTERSECT: Returns common records.
  - EXCEPT: Returns records from the first query not found in the second.
- **ANY / ALL:** Operators for subquery conditions.
  - ANY: True if any subquery value meets the condition.
  - ALL: True if all subquery values meet the condition.
- **GROUP BY:** Group result-sets with aggregate functions (COUNT, MAX, MIN, SUM, AVG).
  - GROUP BY column\_name ORDER BY COUNT(column\_name) DESC;
- **HAVING:** Filter results of aggregate functions.
  - HAVING COUNT(column\_name) > 5;
- **WITH (CTE):** Define temporary, named result-sets.

## 2. Data Modification

- **INSERT INTO:** Add new records.
  - INSERT INTO table\_name (column1, column2) VALUES (value1, value2);
- **UPDATE:** Modify existing records.
  - UPDATE table\_name SET column1 = value1 WHERE condition;
- **DELETE:** Remove records.
  - DELETE FROM table\_name WHERE condition;
  - DELETE \* FROM table\_name;

## 3. Reporting Queries (Aggregate Functions)

- **COUNT():** Returns the number of occurrences.
  - SELECT COUNT(DISTINCT column\_name) FROM table\_name;
- **MIN() / MAX():** Returns the smallest/largest value.
  - SELECT MIN(column\_name) FROM table\_name;
- **AVG():** Returns the average value.
  - SELECT AVG(column\_name) FROM table\_name;
- **SUM():** Returns the total sum.
  - SELECT SUM(column\_name) FROM table\_name;

## 4. Join Queries

- **INNER JOIN:** Returns records with matching values in both tables.
  - FROM table1 INNER JOIN table2 ON table1.column = table2.column;
- **LEFT (OUTER) JOIN:** Returns all records from the left table, and matched from the right.
- **RIGHT (OUTER) JOIN:** Returns all records from the right table, and matched from the left.
- **FULL (OUTER) JOIN:** Returns all records when there's a match in either table.
- **Self JOIN:** A table joined with itself.
  - FROM table1 T1, table1 T2 WHERE condition;

## 5. View Queries

- **CREATE VIEW:** Create a virtual table based on a query.
  - CREATE VIEW view\_name AS SELECT column1 FROM table\_name;
- **SELECT:** Retrieve data from a view.
  - SELECT \* FROM view\_name;
- **DROP VIEW:** Delete a view.
  - DROP VIEW view\_name;

## 6. Altering Table Queries

- **ADD COLUMN:** Add a new column to a table.
  - ALTER TABLE table\_name ADD column\_name datatype;
- **MODIFY COLUMN:** Change the data type of a column.
  - ALTER TABLE table\_name MODIFY column\_name datatype;
- **DROP COLUMN:** Delete a column from a table.
  - ALTER TABLE table\_name DROP COLUMN column\_name;

## 7. Creating Table Query

- **CREATE TABLE:** Create a new table.
  - CREATE TABLE table\_name (column1 datatype, column2 datatype);

## SQL Joins

**INNER JOIN**



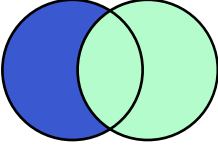
Returns only matching records from both tables.  
`SELECT *  
FROM A  
INNER JOIN B ON A.key = B.key`

**LEFT JOIN**



Returns all records from A, and matching ones from B (if any).  
`SELECT *  
FROM A  
LEFT JOIN B ON A.key = B.key`

**LEFT ANTI JOIN**



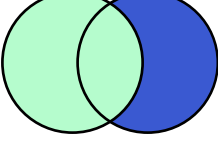
Returns records from A that have no match in B.  
`SELECT *  
FROM A  
LEFT JOIN B ON A.key = B.key WHERE B.key IS NULL`

**RIGHT JOIN**



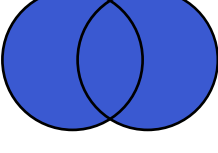
Returns all records from B, and matching ones from A (if any).  
`SELECT *  
FROM A  
RIGHT JOIN B ON A.key = B.key`

**RIGHT ANTI JOIN**



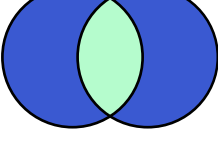
Returns records from B that have no match in A.  
`SELECT *  
FROM A  
RIGHT JOIN B ON A.key = B.key WHERE A.key IS NULL`

**FULL JOIN**



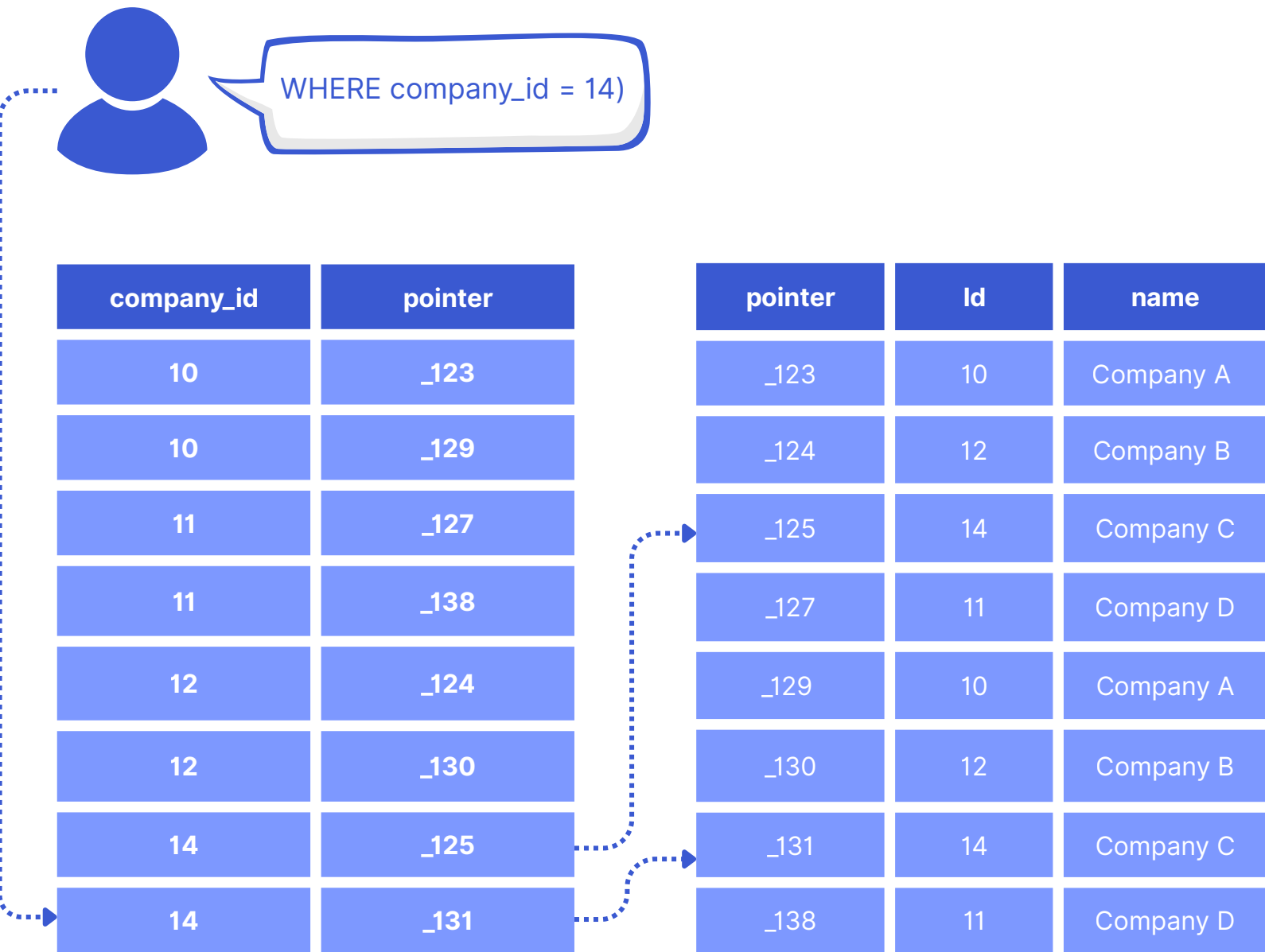
Returns all records from A and B, matched where possible.  
`SELECT *  
FROM A  
FULL JOIN B ON A.key = B.key`

**FULL ANTI JOIN**



Returns all records from A and B that do not match each other.  
`SELECT *  
FROM A FULL JOIN B ON A.key = B.key  
WHERE A.key IS NULL OR B.key IS NULL`

## SQL Indexing



## SQL Transactions

