# SQL Interview Cheat Sheet (Bluish Theme)

---

## SAMPLE DATA

**customers Table:**

| id | name | age | city | country | has_subscription |
|---|---|---|---|---|---|
| 1 | Adam | 35 | New York | USA | TRUE |
| 2 | John | 50 | Toronto | Canada | FALSE |
| 3 | Ravi | 28 | New Delhi | India | TRUE |

**orders Table:**

| order_id | cust_id | order_date | cost | discount | status |
|---|---|---|---|---|---|
| 101 | 1 | 2022-01-01 | 100.00 | 0.05 | SHIPPED |
| 102 | 2 | 2022-02-15 | 200.00 | 0.10 | PENDING |
| 103 | 1 | 2022-03-10 | 150.00 | 0.15 | SHIPPED |
| 104 | 3 | 2022-04-20 | 300.00 | 0.20 | TBO |

---

## QUERYING TABLES WITH SELECT

- Fetch all columns from customers table:

```
SELECT * FROM customers;
```

- Fetch name and age from customers:

```
SELECT name, age FROM customers;
```

### Sort Output Using ORDER BY

- Sort by age ascending (default):

```sql
SELECT * FROM customers ORDER BY age;
```

- Sort by age descending:

```sql
SELECT * FROM customers ORDER BY age DESC;
```

### Aliases

- Rename column:

```sql
SELECT name AS customer_name FROM customers;
```

- Rename table:

```sql
SELECT c.name FROM customers c;
```

---

## FILTERING OUTPUT WITH WHERE

### Comparison Operators

```sql
SELECT * FROM customers WHERE age > 35;
```

### Filter Text With LIKE

```sql
SELECT * FROM customers WHERE city LIKE 'New%';
```

### BETWEEN and IN

```sql
SELECT * FROM customers WHERE age BETWEEN 30 AND 50;
SELECT * FROM customers WHERE country IN ('USA', 'Canada');
```

### NOT and NULLs

```sql
SELECT * FROM customers WHERE NOT country = 'USA';
SELECT * FROM customers WHERE age IS NULL;
```

---

## COMBINING MULTIPLE TABLES WITH JOINS

### INNER JOIN

```
SELECT * FROM customers c
INNER JOIN orders o ON c.id = o.cust_id;
```

### LEFT JOIN

```
SELECT * FROM customers c
LEFT JOIN orders o ON c.id = o.cust_id;
```

### RIGHT JOIN

```
SELECT * FROM customers c
RIGHT JOIN orders o ON c.id = o.cust_id;
```

### FULL JOIN

```
SELECT * FROM customers c
FULL OUTER JOIN orders o ON c.id = o.cust_id;
```

### CROSS JOIN

```
SELECT * FROM customers c
CROSS JOIN orders o;
```

### SELF JOIN

```
SELECT a.name, b.name FROM customers a, customers b
WHERE a.city = b.city;
```

---

## AGGREGATION AND GROUPING

### GROUP BY

```
SELECT cust_id, SUM(cost) AS total_spent
FROM orders
GROUP BY cust_id;
```

### HAVING

```
SELECT cust_id, SUM(cost) AS total_spent
FROM orders
GROUP BY cust_id
HAVING SUM(cost) > 200;
```

---

## WINDOW FUNCTIONS

### PARTITION BY

```
SELECT order_id, cust_id, cost,
SUM(cost) OVER (PARTITION BY cust_id) AS total_by_customer
FROM orders;
```

### ORDER BY

```
SELECT order_id, cust_id, cost,
RANK() OVER (PARTITION BY cust_id ORDER BY cost DESC) AS rank_order
FROM orders;
```

### RANK Example

```
SELECT order_id, RANK() OVER (ORDER BY cost DESC) AS rank
FROM orders;
```

### LAG Example

```
SELECT order_id, cust_id, cost,
LAG(cost) OVER (PARTITION BY cust_id ORDER BY order_date) AS prev_cost
FROM orders;
```

---

## SUBQUERIES

### Single Value Subquery

```
SELECT * FROM customers
WHERE age > (SELECT AVG(age) FROM customers);
```

### Multiple Value Subqueries

```
SELECT * FROM customers
WHERE country IN (SELECT DISTINCT country FROM customers);
```

---

## CTEs

```
WITH high_value_orders AS (
  SELECT * FROM orders WHERE cost > 200
)
SELECT * FROM high_value_orders;
```

---

## SET OPERATIONS

### UNION

```
SELECT city FROM customers
UNION
SELECT country FROM customers;
```

### UNION ALL

```
SELECT city FROM customers
UNION ALL
SELECT country FROM customers;
```

---

## CASE STATEMENTS

```
SELECT order_id, cost,
CASE
  WHEN cost > 250 THEN 'High'
  WHEN cost BETWEEN 100 AND 250 THEN 'Medium'
  ELSE 'Low'
END AS cost_category
FROM orders;
```

---

## OTHER SQL COMMANDS

- **CREATE TABLE:**

```
CREATE TABLE new_table (
 id INT PRIMARY KEY,
 name VARCHAR(50)
);
```

- **ALTER TABLE:**

```
ALTER TABLE customers ADD email VARCHAR(100);
```

- **DROP TABLE:**

```
DROP TABLE new_table;
```

- **INSERT:**

```
INSERT INTO customers VALUES (4, 'Sophia', 27, 'London', 'UK', TRUE);
```

- **UPDATE:**

```
UPDATE customers SET age = 30 WHERE id = 1;
```

- **DELETE:**

```
DELETE FROM customers WHERE id = 4;
```

- **TRUNCATE:**

```
TRUNCATE TABLE customers;
```

---

**Practice 200+ FAANG SQL Interview Questions at:** DataLemur.com/questions\ **Learn SQL with 30 Free Interactive Lessons at:** DataLemur.com/sql-tutorial