SQL Cheatsheet for Interviews

Basics

```
-- Select all columns
SELECT * FROM employees;

-- Select specific columns
SELECT name, salary FROM employees;

-- Rename column in result
SELECT name AS employee_name FROM employees;

-- Remove duplicates
SELECT DISTINCT department FROM employees;
```

Filtering

```
SELECT * FROM employees WHERE department = 'IT';
SELECT * FROM employees WHERE salary BETWEEN 30000 AND 50000;
SELECT * FROM employees WHERE name LIKE 'A%';
SELECT * FROM employees WHERE department IN ('IT', 'HR');
SELECT * FROM employees WHERE email IS NOT NULL;
```

Sorting

```
SELECT * FROM employees ORDER BY salary DESC;
SELECT * FROM employees ORDER BY department ASC, salary DESC;
```

LIMIT / TOP

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

Aggregation

```
SELECT COUNT(*) FROM employees;
SELECT AVG(salary), MIN(salary), MAX(salary), SUM(salary) FROM employees;
```

GROUP BY & HAVING

```
SELECT department, COUNT(*) AS emp_count FROM employees GROUP BY department;
SELECT department, AVG(salary) FROM employees GROUP BY department HAVING AVG(salary) > 40000;
```

Joins

```
SELECT e.name, d.name FROM employees e INNER JOIN departments d ON e.dept_id = d.id;
SELECT e.name, d.name FROM employees e LEFT JOIN departments d ON e.dept_id = d.id;
SELECT e.name, d.name FROM employees e RIGHT JOIN departments d ON e.dept_id = d.id;
SELECT e.name, d.name FROM employees e FULL OUTER JOIN departments d ON e.dept_id = d.id;
```

Subqueries

```
SELECT name FROM employees WHERE salary > (SELECT AVG(salary) FROM employees);
SELECT dept_avg.dept_id, dept_avg.avg_sal FROM (SELECT dept_id, AVG(salary) AS avg_sal FROM employees GROUP BY dept_id) AS dept_avg;
```

Set Operations

```
SELECT name FROM employees_mumbai UNION SELECT name FROM employees_bangalore;
SELECT name FROM employees_mumbai INTERSECT SELECT name FROM employees_bangalore;
SELECT name FROM employees_mumbai EXCEPT SELECT name FROM employees_bangalore;
```

CASE / IF

```
SELECT name, CASE WHEN salary > 50000 THEN 'High' WHEN salary BETWEEN 30000 AND 50000 THEN 'Medium' ELSE 'Low' END AS salary_grade FROM employees;
```

Insert / Update / Delete

```
INSERT INTO employees (name, salary, department) VALUES ('John', 45000, 'IT');
UPDATE employees SET salary = 50000 WHERE id = 3;
DELETE FROM employees WHERE department = 'HR';
```

Create Table

```
CREATE TABLE employees (id INT PRIMARY KEY AUTO_INCREMENT, name VARCHAR(100), department VARCHAR(50), salary DECIMAL(10,2));
```

Constraints

```
CREATE TABLE users (id INT PRIMARY KEY, email VARCHAR(100) UNIQUE, salary DECIMAL CHECK (salary > 0), dept_id INT, FOREIGN KEY (dept_id) REFERENCES departments(id));
```

Indexes

```
CREATE INDEX idx_email ON users(email);
```

Views

```
CREATE VIEW high_paid_employees AS SELECT * FROM employees WHERE salary > 50000;
SELECT * FROM high_paid_employees;
```

Stored Procedure

```
DELIMITER //
CREATE PROCEDURE GetEmployees()
BEGIN
  SELECT * FROM employees;
END //
DELIMITER ;

CALL GetEmployees();
```