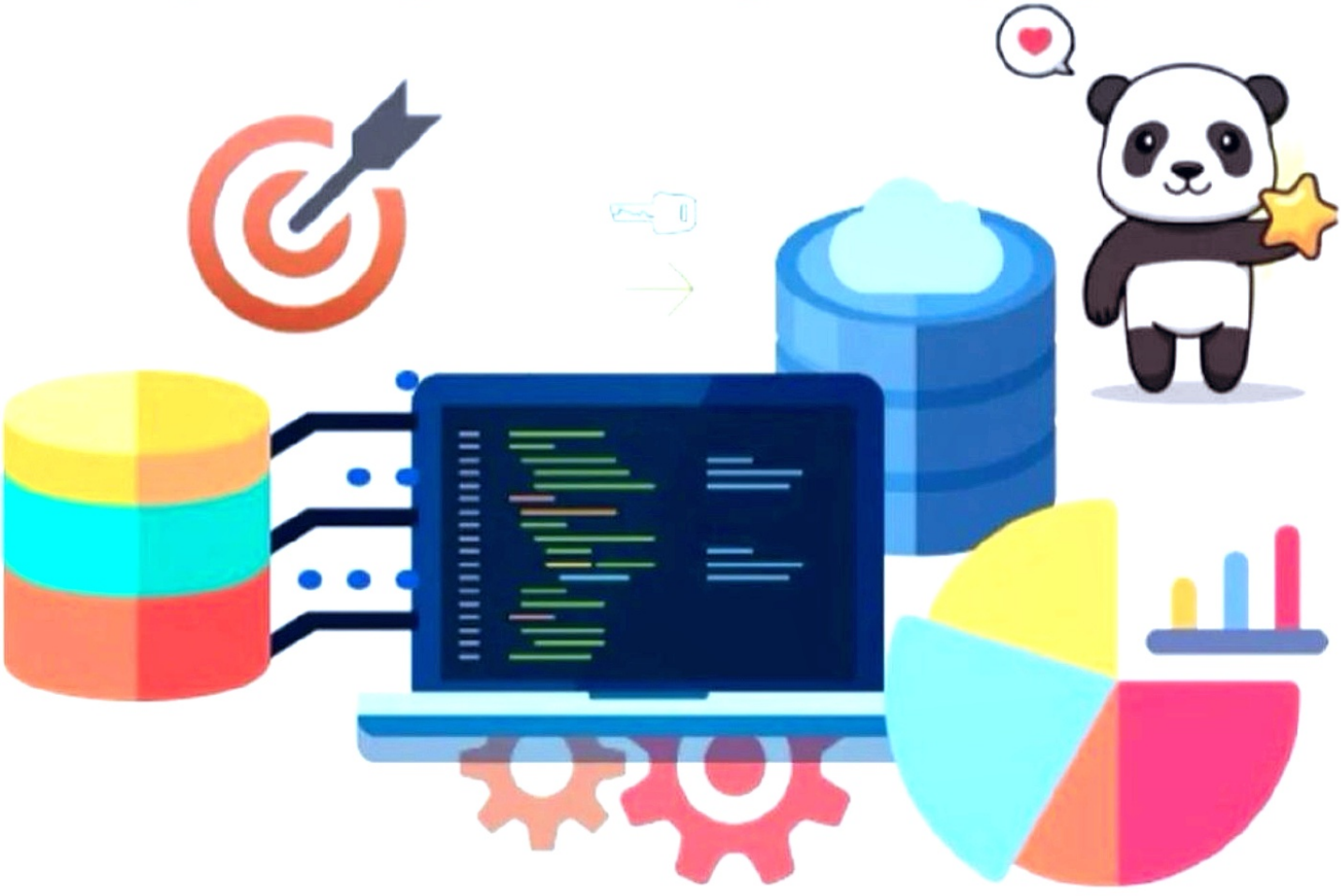
MySQL Cheat Sheet



BEGINNER'S SQL GUIDE

1. Database Operations

- **Create database:** CREATE DATABASE dbname;
- **Drop database:** DROP DATABASE dbname;
- **Select database:** USE dbname;
- **List all databases:** SHOW DATABASES;

2. Table Operations

- **Create table**: CREATE TABLE tablename (id INT, data VARCHAR(100));
- **Drop table**: DROP TABLE tablename;
- **Rename table**: RENAME TABLE oldname TO newname;
- **List all tables**: SHOW TABLES;
- **Describe table structure:** DESCRIBE tablename;
- **Truncate table:**TRUNCATE TABLE tablename;
- **Add column**: ALTER TABLE tablename ADD col_name datatype;
- **Drop column**: ALTER TABLE tablename DROP col_name;

- **Rename column:** ALTER TABLE tablename CHANGE old_col_name new_col_name datatype;
- **Modify column type:** ALTER TABLE tablename MODIFY col_name new_datatype;
- **Add a primary key**: ALTER TABLE tablename ADD PRIMARY KEY (col_name);
- **Drop a primary key**: ALTER TABLE tablename DROP PRIMARY KEY;
- **Add a unique constraint:** ALTER TABLE tablename ADD UNIQUE (col_name);
- **Add a foreign key**: ALTER TABLE tablename ADD CONSTRAINT fk_name FOREIGN KEY (col_name) REFERENCES other_table(col_name);
- **Drop a foreign key:** ALTER TABLE tablename DROP FOREIGN KEY fk_name;
- **Create an index:** CREATE INDEX index_name ON tablename (col_name);
- **Drop an index**: DROP INDEX index_name ON tablename;

3. Data Manipulation

- **Insert row:** INSERT INTO tablename (col1, col2) VALUES (value1, value2);
- **Update rows:** UPDATE tablename SET col1 = value1 WHERE condition;
- **Delete rows:** DELETE FROM tablename WHERE condition;
- **Select data:** SELECT * FROM tablename;
- **Select data with condition:** SELECT * FROM tablename WHERE condition;
- **Select and order data:** SELECT * FROM tablename ORDER BY col ASC/DESC;
- **Select distinct rows:** SELECT DISTINCT col FROM tablename;
- **Count rows:** SELECT COUNT(*) FROM tablename;
- **Sum:** SELECT SUM(col) FROM tablename;
- **Average:** SELECT AVG(col) FROM tablename;
- **Limit & Offset:** SELECT * FROM tablename LIMIT number OFFSET number;
- **Group data:** SELECT col, COUNT(*) FROM tablename GROUP BY col;
- **Having clause:** SELECT col1, col2, COUNT(*) FROM tablename GROUP BY col1, col2 HAVING COUNT(*) > 1;

4. JOINs

- **LEFT Join:** SELECT * FROM table1 LEFT JOIN table2 ON table1.col = table2.col;
- **RIGHT Join:** SELECT * FROM table1 RIGHT JOIN table2 ON table1.col = table2.col;
- **INNER Join:** SELECT * FROM table1 INNER JOIN table2 ON table1.col = table2.col;
- **FULL Join:** SELECT * FROM table1 LEFT JOIN table2 ON table1.col = table2.col UNION SELECT * FROM table1 RIGHT JOIN table2 ON table1.col = table2.col;
- **Cross Join:** SELECT * FROM table1 CROSS JOIN table2;
- **Self Join:** SELECT a.col, b.col FROM table a JOIN table b ON a.common_col = b.common_col WHERE condition;
- **Natural Join:** SELECT * FROM table1 NATURAL JOIN table2;

5. Subqueries

- **Scalar Subquery (Returns single value):** SELECT col_name FROM table_name WHERE col_name = (SELECT col_name FROM another_table WHERE condition);
- **Row Subquery (Returns single row):** SELECT col1, col2 FROM table_name WHERE (col1, col2) = (SELECT col1, col2 FROM another_table WHERE condition);
- **Col Subquery (Returns single col):** SELECT col_name FROM table_name WHERE col_name IN (SELECT col_name FROM another_table WHERE condition);
- **Table Subquery (Returns a table):** SELECT * FROM (SELECT col1, col2 FROM table_name WHERE condition) AS subquery_alias;
- **Correlated Subquery (Reference to a col from the outer query):** SELECT col_name FROM table_name outer_table_alias WHERE col_name_operator (SELECT col_name FROM another_table WHERE condition = outer_table_alias.col_name);
- **Exists Subquery (Checks for the existence of rows in a subquery):** SELECT col_name FROM table_name WHERE EXISTS (SELECT col_name FROM another_table WHERE condition);
- **NOT EXISTS Subquery (Checks for the non-existence of rows in a subquery):** SELECT col_name FROM table_name WHERE NOT EXISTS (SELECT col_name FROM another_table WHERE condition);

6. Text and String Functions

- **Concatenate string:** SELECT CONCAT(col1, ' ', col2) AS col12 FROM tablename;
- **Uppercase:** SELECT UPPER(col) FROM tablename;
- **Lowercase:** SELECT LOWER(col) FROM tablename;
- **Substring:** SELECT SUBSTRING(col, 1, 10) FROM tablename;
- **Replace text:** SELECT REPLACE(col, 'old', 'new') FROM tablename;
- **Length of a string:** SELECT LENGTH(col) FROM tablename;
- **Trim spaces:** SELECT TRIM(col) FROM tablename;
- **Find position of substring:** SELECT INSTR(col, 'substring') FROM tablename;

7. Numeric and Date Functions

- **Round number:** SELECT ROUND(col, decimals) FROM tablename;
- **Get current date:** SELECT CURDATE();
- **Get current time:** SELECT CURTIME();
- **Extract year from date:** SELECT YEAR(col) FROM tablename;
- **Extract month from date**: SELECT MONTH(col) FROM tablename;
- **Date difference:** SELECT DATEDIFF(date1, date2) FROM tablename;
- **Add days to a date:** SELECT DATE_ADD(col, INTERVAL 10 DAY) FROM tablename;
- **Format date**: SELECT DATE_FORMAT(col, '%Y-%m-%d') FROM tablename;

8. Set Operations

- **Union:** SELECT col FROM table1 UNION SELECT col FROM table2;
- **Union All:** SELECT col FROM table1 UNION ALL SELECT col FROM table2;
- **Except:** SELECT col FROM table1 WHERE NOT EXISTS (SELECT col FROM table2 WHERE table1.col = table2.col);

9. Aggregate Functions

- **Minimum value**: SELECT MIN(col) FROM tablename;
- **Maximum value:** SELECT MAX(col) FROM tablename;
- **Average:** SELECT AVG(col) FROM tablename;
- **Standard deviation:** SELECT STDDEV(col) FROM tablename;
- **Variance**: SELECT VARIANCE(col) FROM tablename;
- **Group concat:** GROUP_CONCAT(expression SEPARATOR 'separator');
- **Sum over:** SUM(expression) OVER (PARTITION BY col ORDER BY col);

10. Window Functions

- **ROW_NUMBER():** ROW_NUMBER() OVER (ORDER BY col_name)
- **RANK():** RANK() OVER (ORDER BY col_name)
- **DENSE_RANK():** DENSE_RANK() OVER (ORDER BY col_name)
- **NTILE():** NTILE(num_buckets) OVER (ORDER BY col_name)
- **LAG():** LAG(col_name, offset, default_value) OVER (ORDER BY col_name)
- **LEAD():** LEAD(col_name, offset, default_value) OVER (ORDER BY col_name)
- **FIRST_VALUE():** FIRST_VALUE(col_name) OVER (ORDER BY col_name)
- **LAST_VALUE():** LAST_VALUE(col_name) OVER (ORDER BY col_name)
- **CUME_DIST():** CUME_DIST() OVER (ORDER BY col_name)
- **PERCENT_RANK():** PERCENT_RANK() OVER (ORDER BY col_name)
- **PERCENTILE_CONT():** PERCENTILE_CONT(percent) WITHIN GROUP (ORDER BY col_name)
- **PERCENTILE_DISC():** PERCENTILE_DISC(percent) WITHIN GROUP (ORDER BY col_name)
- **NTH_VALUE():** NTH_VALUE(col_name, n) WITHIN GROUP (ORDER BY col_name)

11. Stored Procedure & CTE

- **Stored Procedure:** CREATE PROCEDURE procedure_name ([parameters])

```
BEGIN
  -- SQL statements;
END;)
```

- **Common Table Expression (CTE):** WITH cte_name AS (

```
  -- CTE query here )
SELECT * FROM cte_name;
```

12. Conditional Expressions

- **IF function:** SELECT IF(condition, value_if_true, value_if_false) FROM tablename;
- **Simple CASE statement:** SELECT col, CASE WHEN value1 THEN result1 ELSE default_result END FROM tablename;
- **Searched CASE statement:** SELECT col, CASE WHEN condition1 THEN result1 ELSE default_result END FROM tablename;
- **COALESCE function:** SELECT COALESCE(col, 'default_value') FROM tablename;
- **NULLIF function**: SELECT NULLIF(col, 'default_value') FROM tablename;
- **IFNULL function**: SELECT IFNULL(col, 'default_value') FROM tablename;
- **NULLIFNULL function**: SELECT NULLIFNULL(col, 'default_value') FROM tablename;

13. User and Permissions

- **Create user:** CREATE USER 'user'@'host' IDENTIFIED BY 'password';
- **Grant permissions:** GRANT ALL PRIVILEGES ON dbname.* TO 'user'@'host';
- **Revoke permissions:** REVOKE ALL PRIVILEGES ON dbname.* FROM 'user'@'host';
- **Set password:** SET PASSWORD FOR 'user'@'host' = PASSWORD('newpassword');
- **Change password**: ALTER USER 'user'@'host' IDENTIFIED BY 'newpassword';
- **Show grants**: SHOW GRANTS FOR 'user'@'host';
- **Drop user:** DROP USER 'user'@'host';
- **Flush privileges:** FLUSH PRIVILEGES;

14. Backup and Recovery

- Backup a database: mysqldump -u username -p dbname > backupfile.sql;
- Restore a database: mysql -u username -p dbname < backupfile.sql;

SQL CIRCLE


• Order By ASC
• Order By DESC
Order
• Inner Join
• Left Join
• Right Join
• Full Join
Joins
• AVG()
• MAX()
• MIN()
• SUM()
• COUNT()
Functions
• Group By Column
• Having
Group
Alias
• As
Where
• Like
• In
• Between
• Any
• Exist, All

DCL COMMANDS:

1. GRANT :

It is used to give user access privileges to a database.

Syntax:
GRANT SELECT, UPDATE ON MY_TABLE TO SOME_USER, ANOTHER_USER;

2. REVOKE :

GRANT SELECT, UPDATE ON MY_TABLE TO SOME_USER< ANOTHER_USER;

Syntax:
REVOKE SELECT, UPDATE ON MY_TABLE FROM USER!, USER2;

TCL COMMANDS:

1. COMMIT :

Commits a Transaction. The COMMIT command saves all the transactions to the database since the last COMMIT or ROLLBACK command.

Syntax:
COMMIT;

Example:
DELETE FROM Student WHERE AGE = 21;
COMMIT:

2. ROLLBACK :

If any error occurs with any of the SQL-grouped statements, all changes need to be aborted. The process of reversing changes is called rollback

Syntax:
ROLLBACK;

Example:
DELETE FROM Student WHERE AGE = 21;
ROLLBACK;