Comprehensive SQL Guide

1. Basic SQL Commands

- SELECT: Retrieve data from a database.

```
SELECT column1, column2 FROM table_name;
```

- INSERT INTO: Insert new data into a table.

```
INSERT INTO table_name (column1, column2)
VALUES (value1, value2);
```

- UPDATE: Update existing data in a table.

```
UPDATE table_name
SET column1 = value1, column2 = value2
WHERE condition;
```

- DELETE: Delete data from a table.

```
DELETE FROM table_name WHERE condition;
```

- CREATE TABLE: Create a new table.

```
CREATE TABLE table_name (
    column1 datatype,
    column2 datatype,
    ...
);
```

- DROP TABLE: Delete a table from the database.

DROP TABLE table_name;

- ALTER TABLE: Modify an existing table.

ALTER TABLE table_name ADD column_name datatype;

ALTER TABLE table_name DROP COLUMN column_name;

2. Data Types

- INT: Integer

- VARCHAR: Variable-length string

- DATE: Date

- BOOLEAN: True/False value

- DECIMAL: Decimal number

3. Filtering Data

- WHERE: Filter data based on conditions.

SELECT column1, column2 FROM table_name WHERE condition;

- AND / OR: Combine multiple conditions.

SELECT * FROM table_name WHERE column1 = value1 AND column2 = value2;

- BETWEEN: Filter data within a range.

SELECT * FROM table_name WHERE column_name BETWEEN value1 AND value2;

- LIKE: Filter data using a pattern.

SELECT * FROM table_name WHERE column_name LIKE 'pattern%';

- IN: Filter data within a set of values.

SELECT * FROM table_name WHERE column_name IN (value1, value2, value3);

4. Sorting and Grouping

- ORDER BY: Sort data in ascending or descending order.

SELECT column1, column2 FROM table_name ORDER BY column1 ASC;

- GROUP BY: Group data by one or more columns.

SELECT column1, COUNT(*) FROM table_name GROUP BY column1;

- HAVING: Filter groups based on a condition.

SELECT column1, COUNT(*) FROM table_name GROUP BY column1 HAVING COUNT(*) > 5;

5. Joins

- INNER JOIN: Retrieve rows that have matching values in both tables.

SELECT column1, column2 FROM table1 INNER JOIN table2 ON table1.column = table2.column;

- LEFT JOIN: Retrieve all rows from the left table and matching rows from the right table.

SELECT column1, column2 FROM table1 LEFT JOIN table2 ON table1.column = table2.column;

- RIGHT JOIN: Retrieve all rows from the right table and matching rows from the left table.

SELECT column1, column2 FROM table1 RIGHT JOIN table2 ON table1.column = table2.column;

- FULL OUTER JOIN: Retrieve all rows when there is a match in one of the tables.

SELECT column1, column2 FROM table1 FULL OUTER JOIN table2 ON table1.column =

table2.column;

6. Subqueries

A subquery is a query within another query.

SELECT column1 FROM table_name WHERE column2 = (SELECT column2 FROM table_name WHERE condition);

7. Aggregate Functions

- COUNT: Count the number of rows.

SELECT COUNT(*) FROM table_name;

- SUM: Sum the values in a column.

SELECT SUM(column_name) FROM table_name;

- AVG: Calculate the average of a column.

SELECT AVG(column_name) FROM table_name;

- MIN: Find the minimum value.

SELECT MIN(column_name) FROM table_name;

- MAX: Find the maximum value.

SELECT MAX(column_name) FROM table_name;

8. Constraints

- PRIMARY KEY: Uniquely identifies each record in a table.

CREATE TABLE table_name (

```
    column1 datatype PRIMARY KEY,
    column2 datatype
);
```

- FOREIGN KEY: Ensures referential integrity by linking to another table.

```
CREATE TABLE table_name (
    column1 datatype,
    column2 datatype,
    FOREIGN KEY (column2) REFERENCES another_table(column_name)
);
```

- NOT NULL: Ensures that a column cannot have a NULL value.

```
CREATE TABLE table_name (
    column1 datatype NOT NULL,
    column2 datatype
);
```

- UNIQUE: Ensures all values in a column are unique.

```
CREATE TABLE table_name (
    column1 datatype UNIQUE,
    column2 datatype
);
```

- DEFAULT: Sets a default value for a column.

```
CREATE TABLE table_name (
    column1 datatype DEFAULT value,
```

```
    column2 datatype
);
```

9. Transactions

- START TRANSACTION: Begin a transaction.

```
START TRANSACTION;
```

- COMMIT: Save the changes.

```
COMMIT;
```

- ROLLBACK: Undo the changes.

```
ROLLBACK;
```

10. Indexes

Indexes improve the speed of data retrieval.

```
CREATE INDEX index_name ON table_name (column_name);
```