SQL Commands

With Example

Compiled by: Arockia Liborious
Source: Unknown

ALTER TABLE

ALTER TABLE lets you add columns to a table in a database.

```
ALTER TABLE table_name
ADD column_name datatype;
```

SQL

AND

AND is an operator that combines two conditions. Both conditions must be true for the row to be included in the result set.

```
SELECT column_name(s)
FROM table_name
WHERE column_1 = value_1
  AND column_2 = value_2;
```

SQL

WITH

WITH clause lets you store the result of a query in a temporary table using an alias. You can also define multiple temporary tables using a comma and with one instance of the WITH keyword.

```
WITH temporary_name AS (
    SELECT *
    FROM table_name)
SELECT *
FROM temporary_name
WHERE column_name operator value;
```

WHERE

WHERE is a clause that indicates you want to filter the result set to include only rows where the following condition is true.


```
SELECT column_name(s)
FROM table_name
WHERE column_name operator value;
```

SELECT DISTINCT

SELECT DISTINCT specifies that the statement is going to be a query that returns unique values in the specified column(s).

```
SELECT DISTINCT column_name
FROM table_name;
```

SUM

SUM() is a function that takes the name of a column as an argument and returns the sum of all the values in that column.

```
SELECT SUM(column_name)
FROM table_name;
```

UPDATE

UPDATE statements allow you to edit rows in a table.


```
UPDATE table_name
SET some_column = some_value
WHERE some_column = some_value;
```

SELECT

SELECT statements are used to fetch data from a database. Every query will begin with SELECT.

```
SELECT column_name
FROM table_name;
```

ROUND()

ROUND() is a function that takes a column name and an integer as an argument. It rounds the values in the column to the number of decimal places specified by the integer.

```
SELECT ROUND(column_name, integer)
FROM table_name;
```

OUTER JOIN

An outer join will combine rows from different tables even if the join condition is not met. Every row in the left table is returned in the result set, and if the join condition is not met, then NULL values are used to fill in the columns from the right table.

```
SELECT column_name(s)
FROM table_1
LEFT JOIN table_2
ON table_1.column_name =
    table_2.column_name;
```

MIN()

MIN() is a function that takes the name of a column as an argument and returns the smallest value in that column.


```
SELECT MIN(column_name)
FROM table_name;
```

OR

OR is an operator that filters the result set to only include rows where either condition is true.


```sql
SELECT column_name
FROM table_name
WHERE column_name = value_1
    OR column_name = value_2;
```

ORDER BY

ORDER BY is a clause that indicates you want to sort the result set by a particular column either alphabetically or numerically.


```
SELECT column_name
FROM table_name
ORDER BY column_name ASC | DESC;
```

MAX()

MAX() is a function that takes the name of a column as an argument and returns the largest value in that column.


```
SELECT MAX(column_name)
FROM table_name;
```

LIMIT

LIMIT is a clause that lets you specify the maximum number of rows the result set will have.


```
SELECT column_name(s)
FROM table_name
LIMIT number;
```

LIKE

LIKE is a special operator used with the WHERE clause to search for a specific pattern in a column.


```
SELECT column_name(s)
FROM table_name
WHERE column_name LIKE pattern;
```

INNER JOIN

An inner join will combine rows from different tables if the join condition is true.

```
SELECT column_name(s)
FROM table_1
JOIN table_2
  ON table_1.column_name =
      table_2.column_name;
```

INSERT

INSERT statements are used to add a new row to a table.

```
INSERT INTO table_name (column_1,
            column_2, column_3)
VALUES (value_1, 'value_2', value_3);
```

IS NULL / IS NOT NULL

IS NULL and IS NOT NULL are operators used with the WHERE clause to test for empty values.


```
SELECT column_name(s)
FROM table_name
WHERE column_name IS NULL;
```

HAVING

HAVING was added to SQL because the WHERE keyword could not be used with aggregate functions.

```
SELECT column_name, COUNT(*)
FROM table_name
GROUP BY column_name
HAVING COUNT(*) > value;
```

GROUP BY

GROUP BY is a clause in SQL that is only used with aggregate functions. It is used in collaboration with the SELECT statement to arrange identical data into groups.

```
SELECT column_name, COUNT(*)
FROM table_name
GROUP BY column_name;
```

DELETE

DELETE statements are used to remove rows from a table.

```
DELETE FROM table_name
WHERE some_column = some_value;
```