MySQL Example Codes for Various SQL Operations

Table of Contents

Starting with Database management

 # Creating database

```
CREATE DATABASE mydatabase;
```

 # Selecting database

```
USE mydatabase;
```

 # Modifying database

```
ALTER DATABASE mydatabase
   CHARACTER SET = utf8mb4
   COLLATE = utf8mb4_unicode_ci;
```

 # Deleting database

```
DROP DATABASE mydatabase;
```

Basic SQL Queries

 # SELECT-FROM

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

 # DISTINCT

```
SELECT DISTINCT column1
FROM table_name;
```

 # AS

```
SELECT column1 AS alias_name
FROM table_name;
```

 # WHERE

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

Managing Tables in Database System

 # CREATE

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

 # NOT NULL

```
ALTER TABLE table_name
MODIFY column_name datatype NOT NULL;
```

 # UNIQUE

```
ALTER TABLE table_name
ADD CONSTRAINT constraint_name UNIQUE (column_name);
```

 # INSERT INTO

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

 # UPDATE

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

 # DELETE

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

 # TRUNCATE

```
TRUNCATE TABLE table_name;
```

 # DROP

```
DROP TABLE table_name;
```

Working with columns and constraints

 # ADD COLUMN

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

 # MODIFY COLUMN

ALTER TABLE table_name

MODIFY column_name datatype;

 # RENAME COLUMN

ALTER TABLE table_name

CHANGE old_column_name new_column_name datatype;

 # DROP COLUMN

ALTER TABLE table_name

DROP COLUMN column_name;

 # ADD CONSTRAINTS

ALTER TABLE table_name

ADD CONSTRAINT constraint_name CHECK (condition);

 # ADD CONSTRAINTS...REFERENCES

ALTER TABLE table_name

ADD CONSTRAINT constraint_name FOREIGN KEY (column_name) REFERENCES other_table(column_name);

Working with Indexing Operation

 # CREATE INDEX

CREATE INDEX index_name

ON table_name (column1, column2);

 # CREATE UNIQUE INDEX

```
CREATE UNIQUE INDEX index_name
ON table_name (column1);
```

 # DROP INDEX

```
DROP INDEX index_name
ON table_name;
```

Dealing with NULL/MISSING values

 # IS NULL

```
SELECT column1
FROM table_name
WHERE column1 IS NULL;
```

 # IS NOT NULL

```
SELECT column1
FROM table_name
WHERE column1 IS NOT NULL;
```

Various Aspects of Filtering Data

 # AND

```
SELECT column1, column2
FROM table_name
WHERE condition1 AND condition2;
```

 # OR

```
SELECT column1, column2
FROM table_name
WHERE condition1 OR condition2;
```

 # NOT

```
SELECT column1, column2
FROM table_name
WHERE NOT condition;
```

 # BETWEEN

```
SELECT column1
FROM table_name
WHERE column1 BETWEEN value1 AND value2;
```

 # LIKE

```
SELECT column1
FROM table_name
WHERE column1 LIKE 'pattern';
```

 # IN

```
SELECT column1
FROM table_name
WHERE column1 IN (value1, value2, ...);
```

 # LIMIT

```
SELECT column1
FROM table_name
LIMIT number;
```

IMPORTANT MySQL String Functions

 # CHAR_LENGTH

```
SELECT CHAR_LENGTH(column1)
FROM table_name;
```

 # CONCAT

```
SELECT CONCAT(column1, column2)
FROM table_name;
```

 # LOWER

```
SELECT LOWER(column1)
FROM table_name;
```

 # UPPER

```
SELECT UPPER(column1)
FROM table_name;
```

 # TRIM

```
SELECT TRIM(column1)
FROM table_name;
```

 # REPLACE

```
SELECT REPLACE(column1, 'old_string', 'new_string')
FROM table_name;
```

IMPORTANT MySQL Arithmetic Functions

 # ABS

```
SELECT ABS(column1)
FROM table_name;
```

 # SUM

```
SELECT SUM(column1)
FROM table_name;
```

 # AVG

```
SELECT AVG(column1)
FROM table_name;
```

 # COUNT

```
SELECT COUNT(column1)
FROM table_name;
```

 # MIN

```
SELECT MIN(column1)
FROM table_name;
```

 # MAX

```
SELECT MAX(column1)
FROM table_name;
```

IMPORTANT MySQL Transformation Functions

 # POWER

```
SELECT POWER(column1, 2)
FROM table_name;
```

 # ROUND

```
SELECT ROUND(column1, 2)
FROM table_name;
```

IMPORTANT MySQL Date Functions

 # DATEDIFF

```
SELECT DATEDIFF(date1, date2)
FROM table_name;
```

 # DATE_FORMAT

```
SELECT DATE_FORMAT(date_column, '%Y-%m-%d')
FROM table_name;
```

 # DAY

```
SELECT DAY(date_column)
FROM table_name;
```

 # MONTH

```
SELECT MONTH(date_column)
FROM table_name;
```

 # YEAR

```
SELECT YEAR(date_column)
FROM table_name;
```

Grouping and Sorting data in SQL

 # GROUP BY

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

 # ORDER BY

```
SELECT column1, column2
FROM table_name
ORDER BY column1 ASC, column2 DESC;
```

JOINS for Data Retrievals IN SQL

 # INNER JOIN

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

 # LEFT JOIN

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

 # RIGHT JOIN

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

 # CROSS JOIN

```
SELECT column1, column2
FROM table1
CROSS JOIN table2;
```

Advanced operations in SQL

 # HAVING

```
SELECT column1, COUNT(column2)
FROM table_name
GROUP BY column1
HAVING COUNT(column2) > 1;
```

 # EXISTS

```
SELECT column1
FROM table_name1
WHERE EXISTS (SELECT column2 FROM table_name2 WHERE table_name1.id =
table_name2.id);
```

 # ANY

```
SELECT column1
FROM table_name
WHERE column1 operator ANY (SELECT column1 FROM another_table WHERE
condition);
```

 # CASE

```
SELECT column1,
  CASE
    WHEN condition1 THEN 'Result1'
    WHEN condition2 THEN 'Result2'
    ELSE 'Result'
  END
FROM table_name;
```

Stored Procedure and Comments

 # SQL Comments systems

-- This is a comment in SQL

/* This is a comment in SQL

This is a comment in SQL*/

 # Storing & executing procedures

```
DELIMITER //
CREATE PROCEDURE procedure_name()
BEGIN
  -- Procedure statements
END //
DELIMITER ;
```