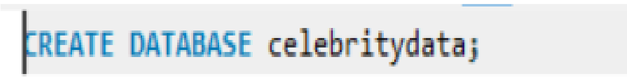
MYSQL DOCUMENTATION.

MySQL is a powerful tool used to store, organize, and manage data. Below is a step-by-step explanation of how MySQL is typically used in data-related projects.

1. Create a Database:

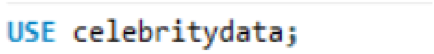
A database is like a digital filing cabinet where your data will be stored.



```
CREATE DATABASE celebritydata;
```

2. Select the Database to Work With:

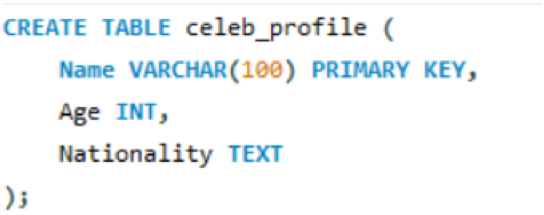
Before you can create tables or insert data, you need to choose the database you'll be working in.



```
USE celebritydata;
```

3. Create Tables:

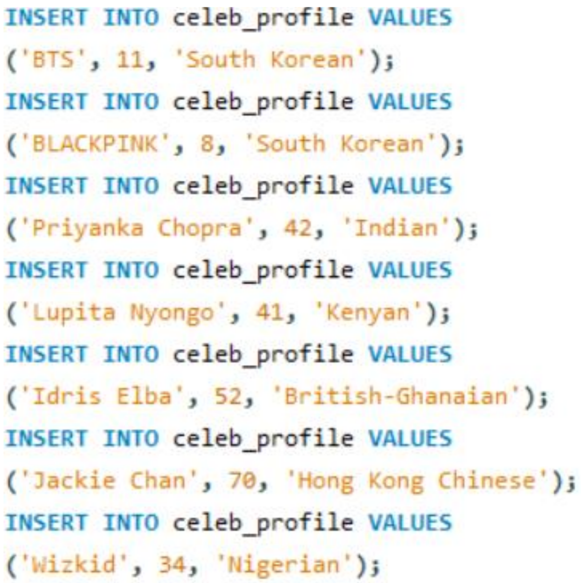
Tables are where data is actually stored. Each table holds related information (e.g., employees, sales, products).



```
CREATE TABLE celeb_profile (
    Name VARCHAR(100) PRIMARY KEY,
    Age INT,
    Nationality TEXT
);
```

4. Insert Data into Tables:

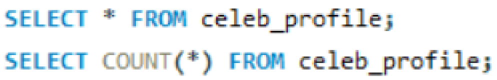
Once the table is created, you can start adding data to it.



```
INSERT INTO celeb_profile VALUES
('BTS', 11, 'South Korean');
INSERT INTO celeb_profile VALUES
('BLACKPINK', 8, 'South Korean');
INSERT INTO celeb_profile VALUES
('Priyanka Chopra', 42, 'Indian');
INSERT INTO celeb_profile VALUES
('Lupita Nyongo', 41, 'Kenyan');
INSERT INTO celeb_profile VALUES
('Idris Elba', 52, 'British-Ghanaian');
INSERT INTO celeb_profile VALUES
('Jackie Chan', 70, 'Hong Kong Chinese');
INSERT INTO celeb_profile VALUES
('Wizkid', 34, 'Nigerian');
```

5. Retrieve Data (Read/Query)

You can view the data stored in the tables using a **SELECT** query.



```
SELECT * FROM celeb_profile;
SELECT COUNT(*) FROM celeb_profile;
```

6. Update Existing Data:

If a record needs to be corrected or changed, you can update it.

Example SQL command:

UPDATE celeb_profile

SET Age = 71

WHERE id = 6;

7. Delete Data:

You can remove data that is no longer needed.

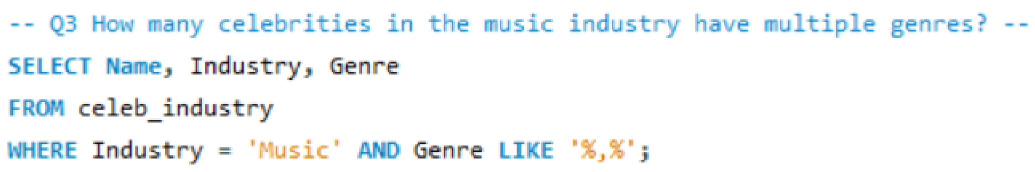
Example SQL command:

DELETE FROM celeb_profile

WHERE id = 5;

8. Use Filters and Conditions:

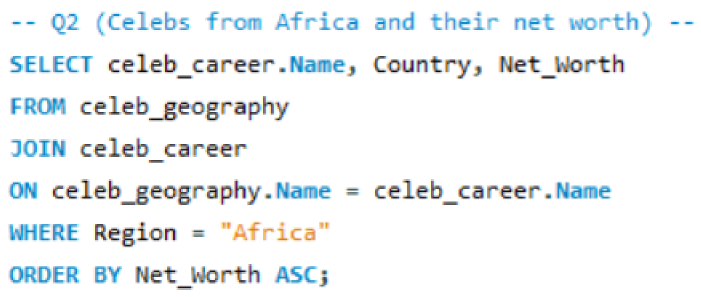
You can query specific data by adding conditions.



```
-- Q3 How many celebrities in the music industry have multiple genres? --
SELECT Name, Industry, Genre
FROM celeb_industry
WHERE Industry = 'Music' AND Genre LIKE '%,%';
```

9. Join Multiple Tables:

MySQL lets you connect different tables to analyze related data together.



```
-- Q2 (Celebs from Africa and their net worth) --
SELECT celeb_career.Name, Country, Net_Worth
FROM celeb_geography
JOIN celeb_career
ON celeb_geography.Name = celeb_career.Name
WHERE Region = "Africa"
ORDER BY Net_Worth ASC;
```

Conclusion:

MySQL is an essential tool for organizing, storing, and analyzing data in a structured way. By following the steps outlined—from creating a database to manipulating and querying data—you can manage information efficiently and make informed decisions. Whether you're handling customer details, sales records, or any other type of data, MySQL provides the flexibility and power needed to keep everything organized and accessible. With consistent practice, even complex data relationships can be handled with ease.