**MySQL Hands-On**

**SQL Server Connection Steps**

* Open **File Explorer** → **C Drive** → **Program Files** → **MySQL** → **Server** → **bin**
* Click on the **address bar**, remove everything, and type **cmd**, then press **Enter**
* Run the following command to connect to the SQL server: **mysql -u root -p**
* It will ask for a **password** — enter your password and get started!

**Create Database**

* MySQL offers a DDL Command called as **create** for creating databases, tables.
* ***Syntax:*** Create Database DATBASE\_NAME;
* ***Example:*** Create Database tutorial;

**Create Database If Not Exists**

* Incase if we have a database already called as tutorial and again, we use create command, we will get an error message.
* When we are not sure if we already had a database with same name or not then we should use **create database along with if not exists.**
* If a database with same name exists, the query will be ignored or else a database will be created.
* ***Syntax:*** Create Database If Not Exists DATABASE\_NAME;
* ***Example:*** Create Database If Not Exists tutorial;

**Switching to a Database**

* Whenever we want to **switch** to other database we need to go with **use** command.
* ***Syntax:*** Use DATABASE\_NAME;
* ***Example:*** Use tutorial;

Current Working Database

* To know about **current working database**, use **select Database()** command.
* ***Example:*** Select Database();

**MySQL Show Database**

* It is used to **list out** all the available databases.
* ***Example:*** Show databases;

**MySQL Drop Database**

* It is used to **delete** a database.
* ***Syntax:*** Drop Database DATABASE\_NAME.
* ***Example:*** Drop Database tutorial;

**MySQL Database Export**

* It is very important to create back-up files of our databases for **exporting** them whenever required.
* MySQL offers **mysqldump** command line tool to export databases quickly.
* **Mysqldump** is used to backup **all databases in the host, an entire database, specific tables or specific rows of a table.**

**MySQL Export Single Database**

* While creating an export we need to provide username, database name and output file path using the below syntax.
* ***Syntax:*** mysqldump -u username -p database\_name > output\_file\_path
* ***Example:*** mysqldump -u root -p tutorial > data-dump.sql
* After executing this, we also need to enter our server **password**.

**MySQL Export Particular tables from a database**

* **Syntax:** mysqldump -u username -p db\_name table1\_name > output file path
* **Example:** mysqldump -u root -p tutorial users > data-dump.sql;

**Export all databases within the host**

* **Example:** mysqldump -u root -p --all-databases > database\_dump.sql;

**MySQL Importing Database**

* Like exporting we can also **import** data using **mysql** and all the commands are almost same but only **‘>’** sign changes to **‘<’**
* **Syntax:** mysql -u username -p new\_database\_name < dumpfile\_path
* **Example:** mysql -u root -p new\_tutorial < database\_dump.sql;

**MySQL Create Table**

* It is used to **create** a new table.
* **Syntax:** Create table TABLE\_NAME (COLUMN1 Datatype Constraints, Column2 Datatype Constraints, Column3 Datatype Constraints);
* **Example:** Create table details(id int auto\_increment primary key, name varchar(30), cgpa double(3,2), gmail varchar(30));

**MySQL Show Tables**

* It is used to list out all the tables in a database.
* **Example:** Show tables;

**MySQL Desc**

* It is used provide **description** about a table by displaying its structure.
* It shows **column names, data types, nullability, keys, default values, and extra info** (like AUTO\_INCREMENT).
* **Syntax:** Desc TABLE\_NAME;
* **Example:** Desc users;

**MySQL Insert Into**

* It is used to **insert data** i.e. records to a table.
* **Syntax:** Insert Into TABLE\_NAME(COLUMN1) Values(Value1);
* **Example:** Insert Into users(name, cgpa, email, branch) Values (‘kalyan’, 9.46, ‘2100030959cseh@gmail.com’, ‘cse-h’);

**MySQL Alter Table**

* It is used to **make changes** like changing table name, column data type, column name, to an existing table.
* **Example1:** To change table name from **tutorial** to **my\_tutorial**.
* Alter table tutorial rename to my\_tutorial;
* Rename table tutorial to my\_tutorial;
* **Example2:** To **change column datatypes** and **add constraints**.
* Alter table my\_tutorial modify id bigint, modify gmail varchar(50) not null, modify name varchar(50) not null;

**Adding Columns to Table**

* Using **alter command** with **add** we can **add** columns to an existing table.
* We can either add columns at first using **FIRST** keyword or add after a particular column using **AFTER** keyword.
* **Syntax:**

ALTER TABLE table\_name

ADD [COLUMN] column\_1\_definition [FIRST|AFTER existing\_column],

ADD [COLUMN] column\_2\_definition [FIRST|AFTER existing\_column];

* **Example:**

Alter table users ADD Column dob date FIRST,

ADD Column phone bigint AFTER cgpa;

**Deleting Columns from Table**

* Using **alter command** with **drop** we can **delete** columns.
* **Syntax:**

ALTER TABLE table\_name DROP Column column\_name;

* **Example:**

ALTER TABLE users DROP Column dob, DROP Column phone;

**Rename Columns**

* Using **alter command** with **rename** we can rename columns.
* **Example:**

Alter table users RENAME COLUMN id to reg\_id, RENAME COLUMN gmail to email;

**MySQL Clone Tables**

* It is used to **copy data** from one table to another.
* **Example1:**

Insert Into copyUsers(name, cgpa, branch, email)

Select name, cgpa, branch, email from users;

* **Example2:**

Select \* into copyUsers from users;

* **Example3:**

Create table copyUsers Select \* from users;

Shallow Cloning

* It is used to **copy the structure** of a table and doesn’t copy the data inside table.
* It is done using **Like** keyword.
* **Example:**

Create table usersCopy LIKE users;

**MySQL Truncate**

* It is used to delete the data inside a table but leaves the structure as it is.
* It is used when we want to delete all rows inside a table but not structure.
* **Example:**

Truncate table copyUsers;

**MySQL Show Columns**

It is like **desc**, it provides the visual structure of a table.

**MySQL Drop Table**

* It is used to delete a table completely including both data and structure.
* **Example:**

Drop table copyUsers;

**MySQL Update Table**

* It is used to **modify existing records** in a specific table with **Set &** **Where** clause.
* **Syntax:**

Update table\_Name

Set Column1 = Value1, Column2 = Value2

Where Condition;

* **Example:**

Update users set id = 2100030959 where name = ‘kalyan’;

**MySQL Delete**

* It is used to **delete existing records** in a specific table using delete from with where clause.
* **Syntax:**

Delete from Table\_Name

Where condition;

* **Example:**

Delete from users where id = 2100030959;

**MySQL Constraints**

**Constraints** are used to set certain rules to columns.

**1. NOT NULL**

* Ensures that a column **cannot have NULL values**.
* Used when a field **must always have a value**.

**2. UNIQUE**

* Ensures that **all values in a column are different**.
* Allows **NULLs** (but only one NULL in most databases).

**3. DEFAULT**

* Sets a **default value** for a column if no value is provided during insert.
* **Example:**

Create table users1(name varchar(50) default “john doe”);

**4. PRIMARY KEY**

* Uniquely identifies each row in a table.
* Combines **NOT NULL + UNIQUE**.
* Only **one primary key** per table (can be a single or composite column).

**5. FOREIGN KEY**

* Creates a **link between two tables**.
* Ensures that the value in one table **matches a value in another table** (referential integrity).
* **Example:**

CREATE TABLE ORDERS ( ORDER\_ID int NOT NULL, ORDER\_NUMBER int NOT NULL, CUST\_ID int, FOREIGN KEY (CUST\_ID) REFERENCES CUSTOMERS (CUST\_ID) );

**Primary vs Unique vs Foreign**

* A **primary key** is a constraint that uniquely identifies each record in a table and does not allow NULL or duplicate values; each table can have only one primary key.
* A **unique** constraint also ensures that all values in a column are distinct, but unlike a primary key, it allows one NULL and multiple unique constraints can exist in a table.
* A **foreign key** is used to establish a relationship between two tables by referencing the primary key (or a unique key) of another table, ensuring referential integrity by allowing only values that exist in the referenced column.

**MySQL Replace Into**

* We cannot add **duplicate values** in a table having **primary or unique** key using insert into, but we can use **replace into** to replace a record with unique or primary key.
* **Example:**

Replace into

users (id, name, cgpa, gmail)

values (2100030959, ‘reddy’, 9.46, ‘rvk@gmail.com’);

**MySQL Views**

* **Views** can be defined as the virtual tables that are generated based on the result set of a sql query.
* **Views** are used to simply complex queries and can present data in a way that hides the complexity or sensitive details of the underlying schema.
* **Syntax:**

Create View View\_Name as select statements from table\_name;

* **Example:**

Create View allUsers as select \* from users;

* We can **update, drop, rename views** like tables.

**MySQL Stored Procedures**

* Stored procedures can be defined as a group of **prepared sql queries** that can be reused anytime.
* Stored procedures can be used to perform different database operations such as such as **inserting, updating, or deleting data**.
* **Example:**

CREATE PROCEDURE GetUserInfo()

BEGIN

SELECT \* FROM USERS WHERE ID > 25;

END;

CALL GetUserInfo();

* **Triggers** are special type of stored procedure that are executed automatically when operations like insert, update occur.

Views vs Stored Procedures

A screenshot of a survey

AI-generated content may be incorrect.

**MySQL Group By**

* It is used to arranges data into groups.
* It groups rows that have same values.
* It is often used with aggregate functions.
* In the below example we have grouped the records based on city.
* The below example uses group by to display the count of employees from each city.

A screenshot of a computer

AI-generated content may be incorrect.

**MySQL Having**

* It is used to **filter groups** based on a specified list of conditions.
* It restricts the results returned by group by clause.
* It is often used with aggregate functions.

A screenshot of a computer

AI-generated content may be incorrect.

**Transaction in MySQL**

* A **transaction** is a sequence of one or more SQL operations that are executed as a single unit of work. It ensures data integrity and consistency, especially in multi-step operations.
* The work done using transaction can also be done using normal queries, but transaction offers **more control over commit, can rollback changes,** and **doesn’t support auto commit** like normal queries.

**Key Transaction Commands**

* ***START TRANSACTION or BEGIN:*** Begins a new transaction.
* ***COMMIT:*** Saves all changes made in the transaction permanently.
* ***ROLLBACK:*** Undoes all changes made in the transaction since it began.
* ***SAVEPOINT:*** Sets a point within a transaction to which you can roll back partially.

Scenario Based Example:

A screenshot of a computer code

AI-generated content may be incorrect.

**MySQL Date and Time**

MySQL supports following data types to work with date and time:

* Date: Stored date in the format of **YYYY-MM-DD**.
* DATETIME: A date and time combination in **YYYY-MM-DD HH:MM:SS** format.
* TIME: Stores the time in a **HH:MM:SS** format.
* YEAR(M): Stores a year in a **2-digit or a 4-digit** format.

If the length is specified as 2 (for example YEAR(2)), YEAR can be between 1970 to 2069 (70 to 69).

If the length is specified as 4, then YEAR can be 1901 to 2155. Default length is 4.

**Note**

* **These formats for every type are internal storage formats.**
* **We cannot change the way how they are stored but can change how they appear.**
* **If you want to change how they appear (e.g., for display or export), you can use formatting functions.**

**Built-In Date & Time Functions**

Select CURRENT\_DATE():It displays **current date** in **‘YYYY-MM-DD’** format.

Select DAY(CURRENT\_DATE()): It displays the **day of the month** from provided date.

Select DAYNAME(CURRENT\_DATE()): It displays the **day of the week**.

Select DAYOFWEEK(CURRENT\_DATE()): It displays the **day number of the week(1 - 7)**.

Select DAYOFYEAR(CURRENT\_DATE()):It displays the **day number of the year(1 – 365).**

Select LAST\_DAY(CURRENT\_DATE()):It displays the **last date of present month.**

Select MONTH(CURRENT\_DATE()):It displays the **current month of the year(1 – 12).**

Select MONTHNAME(CURRENT\_DATE()):It displays the **current month name.**

Select WEEK(CURRENT\_DATE()):Displays the **current week number of the year(0-53).**

Select YEAR(CURRENT\_DATE()):It displays the **current year.**

Select CURRENT\_TIME(): It displays the **local system time** in **‘HH:MM:SS’** format.

Select HOUR(CURRENT\_TIME()):It displays the **current hour of the day(0 – 23).**

Select MINUTE(CURRENT\_TIME()):It displays the **current minute of the hour(0 – 59).**

Select NOW():It displays the **current date and time in ‘YYYY-MM-DD HH:MM:SS’.**

**Date\_Format()**

It is used to **format and display** the date according to our requirements.

**Examples:**

* Select DATE\_FORMAT(CURRENT\_DATE(), '%d-%m-%y') AS 'FORMATTED DATE (DD-MM-YYYY)';
* SELECT DATE\_FORMAT(CURRENT\_DATE(), '%D %M %Y') AS 'Formatted Date (e.g., 3rd June 2025)';
* SELECT DATE\_FORMAT(CURRENT\_DATE(), '%W, %e %M %Y') AS 'Formatted Date (Weekday, Day Month Year)';
* SELECT DATE\_FORMAT(NOW(), '%d-%m-%Y %r') AS 'Formatted DateTime (dd-mm-yyyy hh:mm:ss AM/PM)';