My SQL Stored Procedures

What is Store Procedure?

- A stored procedure is a prepared SQL code that you can save, so the code can be reused over and over again
- The stored procedure may contain a conditional statement like IF or CASE or the Loops
- The stored procedure helps to prevent the database from SQL Injection
- Multiple SQL Statements are encapsulated in a stored procedure
- The stored procedure are reusable. We can implement the business logic within
- The stored procedures are more secure than the AdHoc queries

What is an ad hoc query?

- **Ad hoc** is **latin** for "for this purpose"
- An ad hoc query is a single query not included in a stored procedure and not parameterized or prepared
- An ad hoc query is a loosely typed command/query whose value depends upon some variable
- Each time the command is executed, the result is different, depending on the value of the variable
- An ad hoc query is short lived and is created at runtime

Insert Store Procedure

- 1st Change DELIMITER Like (DELIMITER //)

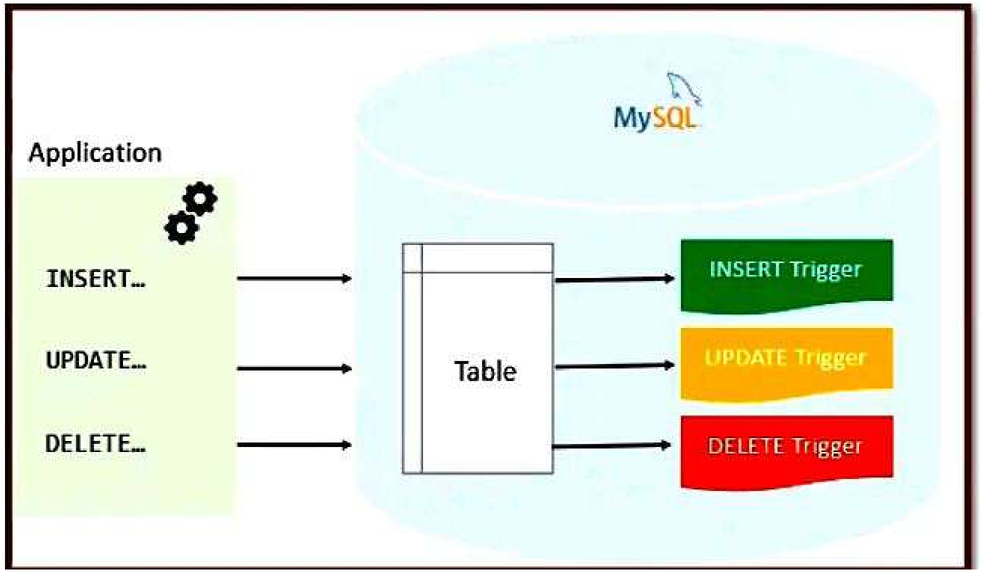
```
create PROCEDURE InserTData
(
          IN id int,
          IN name varchar(20),
          IN age int
)

BEGIN

          INSERT INTO   tech  VALUES(id, name, age);
          select * from aptech;
end
```

```
Call InserTData (1, 'Amjad', 33)
```

TRIGGER IN MYSQL

What is Trigger?

- A trigger in MySQL is a set of SQL statements that reside in a system catalog
- **It is a special type of stored procedure that is invoked automatically in response to an event**
- Each trigger is associated with a table, which is activated on any DML statement such as **INSERT, UPDATE,** or **DELETE**



Use of Trigger……..

- Enforce business rules
- Validate input data
- Generate a unique value for a newly-inserted row in a different file
- Write to other files for audit trail purposes
- Query from other files for cross-referencing purposes
- Access system functions
- Replicate data to different files to achieve data consistency

Types of Triggers in MySQL?

- **Before Insert** : It is activated before the insertion of data into the table
- **After Insert** : It is activated after the insertion of data into the table
- **Before Update:** It is activated before the update of data in the table
- **After Update** : It is activated after the update of the data in the table
- **Before Delete** : It is activated before the data is removed from the table
- **After Delete** : It is activated after the deletion of data from the table

NEW and **OLD** Modifiers

- To distinguish between the value of the columns BEFORE and AFTER the DML has fired, you use the **NEW** and **OLD** modifiers

- For example, if you update the column User_Name, in the trigger body, you can access the value of the User_Name before the update OLD. User_Name and the new value NEW. User_Name

Trigger Event	OLD	NEW
INSERT	No	Yes
UPDATE	Yes	Yes
DELETE	Yes	No