Answer the following Questions

1. What is the difference between stored procedure and stored functions?

Feature	Stored Procedure	Stored Function
Purpose	Used to perform operations and can include multiple SQL statements, logic, and flow control.	Used to compute a value and return it. Primarily designed for computations.
Return Type	May or may not return a value. Typically returns multiple values or no value (using OUT parameters).	Must return a single value (scalar) of a specific data type.
Usage in SQL	Cannot be used directly in SQL queries.	Can be used directly in SQL queries (e.g., in SELECT statements).
Parameters	Supports IN, OUT, and INOUT parameters.	Only supports IN parameters.
Transaction Control	Can contain transaction control commands like COMMIT, ROLLBACK.	Cannot contain transaction control commands.
Execution	Executed using the CALL statement.	Invoked like a regular function in SQL.
Side Effects	Can have side effects, such as modifying database state (INSERT, UPDATE, DELETE).	Should not have side effects; primarily for calculations.
Overloading	Typically supports overloading (same name, different parameters).	Generally supports overloading in some DBMSs.
Error Handling	Can use TRY...CATCH (depending on the DBMS) for error handling.	Limited or no error handling within the function.
Example Use Case	Complex business logic, batch processing.	Simple calculations, data formatting, or aggregations.

2. What is the difference between IN and OUT Parameters in stored Procedure.

Feature	IN Parameter	OUT Parameter
Purpose	Used to pass input values into the stored procedure.	Used to return output values from the stored procedure.
Direction of Data Flow	Data flows **into** the procedure.	Data flows **out of** the procedure.
Default Behavior	This is the default parameter type if no specific type is declared.	Must be explicitly declared as an OUT parameter.

Modifiability	The value of an IN parameter can be read but not modified within the procedure.	The value of an OUT parameter can be modified within the procedure and returned to the caller.
Usage	Used when the procedure needs to accept input from the calling program.	Used when the procedure needs to send output back to the calling program.
Example Use Case	Passing a value to be used in a calculation or query within the procedure.	Returning a calculated result or a fetched value from a database query.
Multiple Parameters	You can have multiple IN parameters in a procedure.	You can have multiple OUT parameters in a procedure.
Initial Value Requirement	Requires an initial value to be passed when calling the procedure.	Does not require an initial value; it's assigned within the procedure.

3. How a stored procedure and Stored function is being called or invoked?

Calling a Stored Procedure

Stored procedures are invoked using the `CALL` statement.

Syntax:

```
CALL procedure_name(parameter1, parameter2, ...);
```

Example: Suppose you have a stored procedure named `CalculateBonus` that takes an employee ID as an input parameter and returns the bonus amount as an output parameter.

```
CALL CalculateBonus(12345, @bonusAmount);
SELECT @bonusAmount; -- To retrieve the output value
```

Calling a Stored Function

Stored functions are invoked as part of an SQL expression. They can be used in `SELECT` statements, `WHERE` clauses, or anywhere an expression is allowed.

Syntax:

```
SELECT function_name(parameter1, parameter2, ...);
```

Example: Suppose you have a stored function named `GetEmployeeBonus` that takes an employee ID and returns the bonus amount.

```
SELECT GetEmployeeBonus(12345) AS BonusAmount;
```

Differences in Invocation

- **Stored Procedure:**
  - Invoked using the `CALL` statement.
  - Cannot be directly used in SQL expressions or queries.
  - Suitable for complex operations, multiple queries, or transaction management.
- **Stored Function:**
  - Invoked as part of an SQL expression.
  - Can be used directly in `SELECT`, `WHERE`, `ORDER BY`, or other SQL clauses.
  - Designed for computations and returning a single value.