Aim : Demonstrate the use of PL/SQL stored procedures and Function

Objectives: To create and work with stored procedures and functions.

Tools Used: MySQL Workbench

Concept:

- **Stored Procedure**

  A stored procedure is a precompiled collection of one or more SQL statements that can be executed as a single unit. Stored procedures are typically used to perform a specific task, such as updating data in a database, and are stored in the database for reuse.

- **Functions**

  A function, on the other hand, is a reusable subprogram in a programming language like PL/SQL. Functions take input parameters, perform calculations or operations, and return a single value. Functions are often used to contain logic that can be reused in different parts of a program or database query.

In summary, stored procedures are used for performing tasks in a database, while functions are used to return a value based on input parameters. Both are essential components of database programming and can help improve code modularity and maintainability.

Lab Assignment on Procedure

Problem Statement:
Use classicmodels. Create a procedure Get_Orders_Status which should accept the status value from the user and show the number of orders for each year for that status.

Stored Procedure :

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `Get_Orders_Status`(IN var1
varchar(30))
BEGIN
select year(orderDate) year, count(status) AS "TOTAL ORDERS" FROM orders where
status=var1 group by year;
END
```

Sql query :

1 . call Get_Orders_Status ('Shipped');

Result Grid | Filter Rows:

	year	TOTAL ORDERS
▶	2003	108
	2004	145
	2005	50

2. call Get_Orders_Status ('cancelled');

Result Grid | Filter Rows:

	year	TOTAL ORDERS
▶	2003	2
	2004	4

3. call Get_Orders_Status ('on Hold');

Result Grid | Filter Rows:

	year	TOTAL ORDERS
▶	2004	1
	2005	3

Lab Assignment on Function

Problem Statement:

1) Create a function to find the cube of a number.

Stored Function :

```
CREATE DEFINER=`root`@`localhost` FUNCTION `Function_cube`(num int)
RETURNS int
   DETERMINISTIC
BEGIN
declare TotalCube int;
set TotalCube =num * num* num;
return TotalCube;
END
```

Sql query :

```
set @ans=Function_cube (15);
select @ans;
```

Result Grid

	@ans
▶	3375

Problem Statement:

2) Use Classicmodels. Create a function which will return the city of the given officeCode.

Stored Function :

```
CREATE DEFINER=`root`@`localhost` FUNCTION `city`(var1 varchar(50)) RETURNS
varchar(50) CHARSET latin1
   DETERMINISTIC
BEGIN
declare city_name varchar(50);
set city_name=(select city from offices where officeCode = var1);
RETURN city_name;
END
```

Sql query :

```
select city(7);
```

Result Grid

	city(7)
▶	London

Problem Statement:

3) Use Classicmodels. Create a function to show the highest MSRP for each product line using window functions.

Stored Function :

```
CREATE DEFINER=`root`@`localhost` FUNCTION `find_highest_MSRP`(var1
varchar(50)) RETURNS decimal(10,2)
DETERMINISTIC
BEGIN
declare highest_MSRP decimal(10, 2);
set highest_MSRP = (select distinct max(MSRP) over (partition by productLine) from
products where productLine =var1);
RETURN highest_MSRP;
END
```

Sql query :

select find_highest_MSRP(‘Classic Cars’);

	find_highest_MSRP("Classic Cars")
▶	214.30

Problem Statement:

4) Use Classicmodels. Create a function to show the customername who has used the highest CreditLimit.

Stored Function :

```
CREATE DEFINER=`root`@`localhost` FUNCTION `cust_name`() RETURNS
varchar(50) CHARSET latin1
   DETERMINISTIC
BEGIN
declare cname varchar(50);
set cname = (select customerName from customers order by creditLimit desc limit 1);
RETURN cname;
END
```

Sql query :

select cust_name();

Result Grid | Filter Rows:

	cust_name()
▶	Euro+ Shopping Channel

Observation :

- I understand how to use the basic building block of PL/SQL.
- Stored procedure was created to perform one or more DML operations on a Database.
- Different functions were performed to retrieve values, format data, or to perform mathematical operations within database queries.