Sr.	Practical
Lab-4	**Views** Create Database with Name: **SQL_Views** **Create following table under SQL_Views database. (Using Design Mode)**

1. Simple View

Student		
Column_Name	**DataType**	**Constraints**
Rno	Int	Primary Key
Name	Varchar (50)	Not Null
Branch	Varchar (50)	Not Null
SPI	Decimal (4,2)	Not Null
Bklog	Int	Not Null

RNo	Name	Branch	SPI	Bklog
101	Raju	CE	8.80	0
102	Amit	CE	2.20	3
103	Sanjay	ME	1.50	6
104	Neha	EC	7.65	1
105	Meera	EE	5.52	2
106	Mahesh	EC	4.50	3

From the above given table perform the following queries:

1. Create a view Personal with all columns.
2. Create a view Student_Details having columns Name, Branch & SPI.
3. Create a view Academic having columns RNo, Name, Branch.
4. Create a view Student_Data having all columns but students whose bklogs are more than 2.
5. Create a view Student_Pattern having RNo, Name & Branch columns in which Name consists of four letters.
6. Insert a new record to Academic view. (107, Meet, ME). Remaining all columns must be null.
7. Update the branch of Amit from CE to ME in Student_Details view.
8. Delete a student whose roll number is 104 from Academic view.
9. Create a view that displays information of all students whose spi is above 8.5.
10. Create a view that displays 0 backlog students.

2. Complex View

Customer		
Column_Name	**DataType**	**Constraints**
CustomerID	Int	Primary Key
FirstName	Varchar (50)	Not Null
LastName	Varchar (50)	Not Null

CustomerID	FirstName	LastName
1	John	Doe
2	Jane	Smith
3	Michael	Johnson
4	Mark	Wood
5	Moin	Khan

Account		
Column_Name	**DataType**	**Constraints**
AccountID	Int	Primary Key
CustomerID	Int	Foreign Key, Null
Balance	Decimal (10,2)	Not Null
AccountType	Varchar (50)	Not Null
CreatedDate	Date	Not Null

AccountID	CustomerID	Balance	AccountType	CreatedDate
101	1	5000	Current	2023-01-01
102	1	8000	Saving	2023-02-25
103	2	10000	Saving	2023-03-30
104	4	15000	Current	2020-06-15
105	3	7500	Saving	2021-11-27
106	5	13450	Current	2019-10-13

From the above given tables perform the following queries:

1. Create view that displays all the customers along with their corresponding account balances.
2. Create view that displays total balance for each customer.
3. Create view that displays customers who have multiple accounts.
4. Create a view that displays customer details who have an account created in the last month.
5. Create a view that displays customers who have the highest account balance.

Lab-5 **PL/SQL Programs**

1. Write a PL/ SQL program to print a welcome message on a screen.
2. Write a PL/SQL program to addition of two numbers.
3. Write a PL/SQL program to print maximum number out of three numbers.
4. Write a PL/ SQL program to print number from 1 to 10. (Using while loop)
5. Write a PL/ SQL program to check where given number is ODD or EVEN.
6. Write a PL/ SQL program to print ODD numbers between 1 and 10.
7. Write a PL/ SQL program to print Sum of numbers from 1 to 50.
8. Write a PL/ SQL program to print Sum of even numbers between 1 to 20.
9. Write a PL/ SQL program to check weather given number is prime or not.
10. Write a PL/ SQL program to inserting even numbers into even table & odd numbers into odd table between 1 to 50.

Lab-6 **Stored Procedures**

- Create tables under **SQL_SP** database as per following data.
  - **Student** (RNo int Primary Key, Name varchar (50) Not Null, Branch varchar (50) Not Null)
  - **Result** (RNo int Foreign Key, Null, SPI Decimal (4,2) Not Null)

Student		
Rno	**Name**	**Branch**
101	Raju	CE
102	Amit	CE
103	Sanjay	ME
104	Neha	EC
105	Meera	EE
106	Mahesh	ME

Result	
Rno	**SPI**
101	8.8
102	9.2
103	7.6
104	8.2
105	7.0
107	8.9

From the above given tables perform the following queries:

1. Both tables Insert.
2. Both tables Update.
3. Both tables Delete.
4. Both tables SelectPK.
5. Both tables SelectAll.
6. Create a stored procedure that takes branch as input and returns a table with all the students studying in that department.
7. Create a stored procedure to display Rno, Name and SPI of first 2 students only.
8. Create a stored procedure which displays branch wise maximum and minimum SPI.
9. Create a stored procedure which displays top 5 students based on SPI in descending order.
10. Alter stored procedure of 9th definition to display all the detail in ascending order by student name.

Lab-7 **User Defined Functions (UDF)**

Create following table under **SQL_UDF** database as per following data. (Using Query)

Employee	
Column_Name	**DataType**
EID	Int
EName	Varchar (100)
Gender	Varchar (10)
JoiningDate	Datetime
Salary	Decimal (8,2)
City	Varchar (100)

EID	EName	Gender	JoiningDate	Salary	City
1	Nick	Male	01-JAN-13	4000	London
2	Julian	Female	01-OCT-14	3000	New York
3	Roy	Male	01-JUN-16	3500	London
4	Tom	Male	*NULL*	4500	London
5	Jerry	Male	01-FEB-13	2800	Sydney
6	Philip	Male	01-JAN-15	7000	New York
7	Sara	Female	01-AUG-17	4800	Sydney
8	Emily	Female	01-JAN-15	5500	New York
9	Michael	Male	*NULL*	6500	London
10	John	Male	01-JAN-15	8800	London

From the above given table perform the following queries:

Scalar Valued Functions

1. Create a function which displays total number of employees.
2. Create a function which returns highest salary from Employee table.
3. Create a function to get the age of the employee based on their joining date.
4. Create a function to calculate the net sales based on the quantity, price, and discount value.
5. Create a function that calculates the factorial of a given number.

Table Valued Functions

1. Create a function which retrieve the data of Employee table.
2. Create a function which returns an Employee table with city wise total salary.
3. Create a function which returns an Employee table with gender wise maximum, minimum, total and average salaries.
4. Create a function which return an Employee table with details of employee whose name starts with **J**.
5. Create a function to get all the male employees.