Sr. | **Practical**

Lab-1

Create Database with Name: **Student_Info**

Create following table under Student_Info database. (Using Design Mode)

Student	
Column_Name	**DataType**
StuID	Int
Name	Varchar (100)
EnrollmentNo	Varchar (12)
Division	Varchar (50)
Sem	Int
BirthDate	Datetime
Email	Varchar (100)
ContactNo	Varchar (50)

StuID	Name	EnrollmentNo	Division	Sem.	BirthDate	Email	ContactNo
101	Naimish Patel	090200107051	BCX-3	3	1992-12-06	naimishp49@gmail.com	8866205253
102	Firoz A. S.	090200107090	BCY-3	3	1994-05-03	Firoz.me@gmail.com	8885999922
103	Krunal Vyas	090243107101	BCZ-5	5	1984-03-01	Krunal.vyas@gmail.com	9990888877
104	Vijay Patel	090200107102	BCX-5	5	1985-02-15	Vijay.patel123@gmail.com	8787878787
105	Vimal Trivedi	090200107103	BCY-3	3	1988-01-20	Maulik123@gmail.com	8789564512

From the above given table perform the following queries:

Part - A

1. Display Name of Student who belongs to either Semester 3 or 5. (Use IN & OR)
2. Find Student Name & Enrollment No in which Student Id between 103 to 105.
3. Find Student Name & Enrollment No with their Email Who belongs to 5th Semester.
4. Display All the Details of first three students.
5. Display Name & Enrollment no of first 30% Students who's contact number ends with 7.
6. Display Unique Semesters.
7. Retrieve All the Students who have no Enrollment.
8. Find All Students whose Name does not start with 'V'.
9. Find All Students in which Email Contains '3@G' & Only Six Characters.
10. Find Out All the Students whose First Name Starts with F And third character must be R.

Part - B

11. Find All the Student Details whose Contact No contains 877.
12. Display Student Name in Which Student belongs to Semester 3 & Contact Number Does Not Contains 8 & 9.
13. Find Students who born after date 01-01-1990.
14. Update Division to BCX-5 & Semester 5 whose Student Id Is 102.
15. Change the Student's Name to Firoz Sherasiya in which Email is Firoz.Me@Gmail.Com & Contact No is 8885999922.

Part - C

16. Add one more Column City Varchar (50) in Student Table.
17. Remove All BCX-3 Division Students.
18. Change Column Name Email to EmailAddress.
19. Remove All the Data from Student Table Using Truncate.
20. Write an SQL query to clone a new table Student_New from Student table with all data.

Lab-2

Create Database with Name: **Employee_Info**

Create following table under Employee_Info database. (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:

Part - A

1. Display all the employees whose name starts with “m” and 4th character is “h”.
2. Find the value of 3 raised to 5. Label the column as output.
3. Write a query to subtract 20 days from the current date.
4. Write a query to display name of employees whose name starts with “j” and contains “n” in their name.
5. Display 2nd to 9th character of the given string “SQL Programming”.

Part - B

6. Display name of the employees whose city name ends with “ney” &contains six characters.
7. Write a query to convert value 15 to string.
8. Add department column with varchar (20) to Employee table.
9. Set the value of department to Marketing who belongs to London city.
10. Display all the employees whose name ends with either “n” or “y”.

Part - C

11. Find smallest integer value that is greater than or equal to 63.1, 63.8 and -63.2.
12. Display all employees whose joining date is not available.
13. Display name of the employees in capital letters and city in small letters.
14. Change the data type of Ename from varchar (30) to char (30).
15. Display city wise maximum salary.

Lab-3

Consider the same Employee table of Lab-2 and perform the following queries:

Part - A

1. Produce output like <Ename> works at <city> and earns <salary>.
2. Find total number of employees whose salary is more than 5000.
3. Write a query to display first 4 & last 3 characters of all the employees.
4. List the city having total salaries more than 15000 and employees joined after 1st January, 2014.
5. Write a query to replace “u” with “oo” in Ename.

Part - B

6. Display city with average salaries and total number of employees belongs to each city.
7. Display total salaries paid to female employees.
8. Display name of the employees and their experience in years.
9. Remove column department from employee table.
10. Update the value of city from syndey to null.

Part - C

11. Retrieve all Employee name, Salary & Joining date.
12. Find out combine length of Ename & Gender.
13. Find the difference between highest & lowest salary.
14. Rename a column from Ename to FirstName.
15. Rename a table from Employee to EmpMaster.

Lab-4

Create Database with Name: **Person_Info**

Create following table under Person_Info database. (Using Design Mode)

Person

Column_Name	DataType	Constraints
PersonID	Int	Primary Key
PersonName	Varchar (100)	Not Null
DepartmentID	Int	Foreign Key, Null
Salary	Decimal (8,2)	Not Null
JoiningDate	Datetime	Not Null
City	Varchar (100)	Not Null

PersonID	PersonName	DepartmentID	Salary	JoiningDate	City
101	Rahul Tripathi	2	56000	01-01-2000	Rajkot
102	Hardik Pandya	3	18000	25-09-2001	Ahmedabad
103	Bhavin Kanani	4	25000	14-05-2000	Baroda
104	Bhoomi Vaishnav	1	39000	08-02-2005	Rajkot
105	Rohit Topiya	2	17000	23-07-2001	Jamnagar
106	Priya Menpara	*NULL*	9000	18-10-2000	Ahmedabad
107	Neha Sharma	2	34000	25-12-2002	Rajkot
108	Nayan Goswami	3	25000	01-07-2001	Rajkot
109	Mehul Bhundiya	4	13500	09-01-2005	Baroda
110	Mohit Maru	5	14000	25-05-2000	Jamnagar

Department

Column_Name	DataType	Constraints
DepartmentID	Int	Primary Key
DepartmentName	Varchar (100)	Not Null, Unique
DepartmentCode	Varchar (50)	Not Null, Unique
Location	Varchar (50)	Not Null

DepartmentID	DepartmentName	DepartmentCode	Location
1	Admin	Adm	A-Block
2	Computer	CE	C-Block
3	Civil	CI	G-Block
4	Electrical	EE	E-Block
5	Mechanical	ME	B-Block

From the above given table perform the following queries:

Part - A

1. Find all persons with their department name & code.
2. Give department wise maximum & minimum salary with department name.
3. Find all departments whose total salary is exceeding 100000.
4. Retrieve person name, salary & department name who belongs to Jamnagar city.
5. Find all persons who does not belongs to any department.

Part - B

6. Find department wise person counts.
7. Find average salary of person who belongs to Ahmedabad city.
8. Produce Output Like: <PersonName> earns <Salary> from department <DepartmentName> monthly. (In Single Column)
9. List all departments who have no persons.
10. Find city & department wise total, average & maximum salaries.

Part - C

11. Display Unique city names.
12. List out department names in which more than two persons.
13. Combine person name's first three characters with city name's last three characters in single column.
14. Give 10% increment in Computer department employee's salary.
15. Display all the person name's who's joining dates difference with current date is more than 365 days.

Lab-5

Views

Create Database with Name: **SQL_Views**

Create following table under SQL_View 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:

Part – A

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.

Part – B

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.

Part – C

11. Create a view Computer that displays CE branch data only.
12. Create a view Result_EC that displays the name and SPI of students with SPI less than 5 of branch EC.
13. Update the result of student Sanjay to 4.90 in Result_EC view.
14. Create a view Stu_Bklog with RNo, Name and Bklog columns in which name starts with 'M' and having bklogs more than 5.
15. Drop Computer view form the database.

2. Complex View

Create following tables under SQL_View database. (Using Design Mode)

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:

Part – A

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.

Part – B

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.

Part – C

6. Create a view that displays name of the customers whose account balance is between 5000 to 10000 and account type is Saving.
7. Create a view that displays minimum and maximum balance for each customer.

Lab-6

Views (Complex View)

Create following tables and solve given queries:

Country		
Column_Name	**DataType**	**Constraints**
Country_ID	Int	Primary Key
Country_Name	Varchar (100)	Not Null, Unique
Population	Int	Not Null
Area_sq_km	Float	Not Null
Capital	Varchar (100)	Not Null
Currency	Varchar (50)	Not Null

State		
Column_Name	**DataType**	**Constraints**
State_ID	Int	Primary Key
State_Name	Varchar (100)	Not Null, Unique
Population	Int	Not Null
Area_sq_km	Float	Not Null
Capital	Varchar (100)	Not Null
Country_ID	Varchar (50)	Foreign Key, Null

Customer_ID	Customer_Name	Population	Area_sq_km	Capital	Currency
1	USA	331002651	9833520	Washington, D.C.	USD
2	CANADA	38005238	9984670	Ottawa	CAD
3	BRAZIL	212559417	8515767	Brasília	BRL
4	INDIA	1380004385	3287263	New Delhi	INR
5	RUSSIA	145934462	17098246	Moscow	RUB
6	CHINA	1439323776	9706961	Beijing	CNY
7	AUSTRALIA	25499881	7692024	Canberra	AUD
8	ARGENTINA	45195777	2780400	Buenos Aires	ARS
9	GERMANY	83783942	357022	Berlin	EUR
10	SOUTH AFRICA	59308690	1221037	Pretoria	ZAR

State_ID	State_Name	Population	Area_sq_km	Capital	Customer_ID
1	California	39538223	423972	Sacramento	1
2	Texas	28995881	695662	Austin	1
3	Ontario	14734014	917741	Toronto	2
4	São Paulo	46289333	248209	São Paulo	3
5	Maharashtra	114063427	307713	Mumbai	4
6	Moscow Oblast	7694989	443562	Moscow	5
7	Beijing	21542000	16410	Beijing	6
8	New South Wales	8160062	800642	Sydney	7
9	Buenos Aires Province	17700000	307571	La Plata	8

10	Bavaria	13076721	70550	Munich	9

Part – A

1. Create a view that displays the top 5 countries with the highest population, along with their population figures.
2. Create a view that lists countries that do not have any states.

Part – B

3. Create a view that displays the capital cities of countries with a population greater than 10 million.
4. Create a view that displays the state with the highest population for each country, along with its population figure.

Part – C

5. Create a view that lists states that do not have a designated capital.
6. Create a view that displays countries with more than one capital city.

Lab-7 **Views (Simple View and Complex View)**

Create following tables and solve given queries:

Customer		
Column_Name	**DataType**	**Constraints**
CustomerID	Int	Primary Key
FirstName	Varchar (50)	Not Null
LastName	Varchar (50)	Not Null
Email	Varchar (50)	Not Null
Phone	Nvarchar (20)	Not Null

CustomerID	FirstName	LastName	Email	Phone
1	John	Doe	john.doe@example.com	1234567890
2	Jane	Smith	jane.smith@example.com	9876543210
3	Mike	Johnson	mike.johnson@example.com	1112223333
4	Emily	Williams	emily.williams@example.com	4445556666
5	David	Brown	david.brown@example.com	7778889999

Orders		
Column_Name	**DataType**	**Constraints**
OrderID	Int	Primary Key
CustomerID	Int	Foreign Key, Null
OrderDate	Date	Not Null
TotalAmount	Decimal (10,2)	Not Null

OrderID	CustomerID	OrderDate	TotalAmount
1001	1	2023-07-01	100.50
1002	1	2023-07-02	75.20
1003	3	2023-07-03	250.75
1004	4	2023-07-04	50.00
1005	5	2023-07-05	300.00

Part – A

1. Create a view AllCustomers to Get All Customers.
2. Create a view AllOrdersView to Get All Orders with customer name.
3. Create a view to Get Customers with No Email Addresses.
4. Create a view to return total number of Customers as Total_Customer.
5. Create a view to return sum of total amount of order as total_amount.

Part – B

6. Insert data into AllCustomers view (6, 'Marry' ,'com', 'marry. com@example.com ', 7778889999).
7. Create a view to Get All Orders with customer name.
8. Create a view to Get Customers with Their Total Order Amount.
9. Create a view to Get Orders with Customer Contact Information.
10. Create a view to Get Customers with Their Latest Order Date.

Part – C

11. Create a view to Get Customers with No Orders.
12. Create a view to Get Customers with Their Total Number of Orders.
13. Create a view to Get Customers with High-Value Orders.
14. Getting Customers with more than 1 order Placed.
15. Getting Customers with Orders in Date Range 2023-07-01 to 2023-07-04.

Lab-8 **PL/SQL Programs**

Part – A

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.

Part – B

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.

Part – C

11. Write a PL/ SQL program to calculate the factorial of N number and display the result.
12. Write a PL/ SQL program to sum of digits of N number and display the result.
13. Write a PL/ SQL program to reverse a string and display the reversed string.
14. Write a PL/ SQL program to generate the Fibonacci series up to N number and display the series.
15. Write a PL/ SQL program to check given year is leap year or not.

Lab-9 **Stored Procedures**

Create tables under SQL_SP database as per following data.

Student		
Column_Name	**DataType**	**Constraints**
RNo	Int	Primary Key
Name	Varchar (50)	Not Null
Branch	Varchar (50)	Not Null

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

Result		
Column_Name	**DataType**	**Constraints**
RNo	Int	Foreign Key, Null
SPI	Decimal (4,2)	Not Null

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:

Part – A

1. Both tables Insert.
2. Both tables Update.
3. Both tables Delete.
4. Both tables SelectPK.
5. Both tables SelectAll.

Part – B

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.

Part – C

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-10 **Stored Procedures**

Consider the following tables and solve the given queries:

Department		
Column_Name	**DataType**	**Constraints**
DepartmentID	Int	Primary Key
DepartmentName	Varchar (100)	Not Null, Unique

Designation		
Column_Name	**DataType**	**Constraints**
DesignationID	Int	Primary Key
DesignationName	Varchar (100)	Not Null, Unique

Person		
Column_Name	**DataType**	**Constraints**
WorkerID	Int	Primary Key, Auto Increment
FirstName	Varchar (100)	Not Null
LastName	Varchar (100)	Not Null
Salary	Decimal (8,2)	Not Null
JoiningDate	Datetime	Not Null
DepartmentID	Int	Foreign Key, Null

DesignationID	Int	Foreign Key, Null

DepartmentID	DepartmentName
1	Admin
2	IT
3	HR
4	Account

DesignationID	DesignationName
11	Jobber
12	Welder
13	Clerk
14	Manager
15	CEO

WorkerID	FirstName	LastName	Salary	JoiningDate	DepartmentID	DesignationID
101	Rahul	Anshu	56000	01-01-1990	1	12
102	Hardik	Hinsu	18000	25-09-1990	2	11
103	Bhavin	Kamani	25000	14-05-1991	*NULL*	11
104	Bhoomi	Patel	39000	20-02-2014	1	13
105	Rohit	Rajgor	17000	23-07-1990	2	15
106	Priya	Mehta	25000	18-10-1990	2	*NULL*
107	Neha	Trivedi	18000	20-02-2014	3	15

Part – A

1. All the tables Insert, Update Delete.
2. Create a stored procedure that takes department name as an input and returns a table with all workers working in that department.

Part – B

3. Create procedure that takes department name & designation name as input and returns a table with worker's first name, salary, joining date & department name.
4. Create a Procedure that takes the first name as an input parameter and display all the details of the worker with their department & designation name.

Part – C

5. Create Procedure which displays department wise maximum, minimum & total salaries.
6. Create Procedure which displays designation wise maximum, minimum & total salaries.

Lab-11

Stored Procedures

Consider the following tables and solve the given queries:

Employees		
Column_Name	**DataType**	**Constraints**
Emp_ID	Int	Primary Key
Emp_Name	Varchar (50)	Not Null
Emp_Salary	Decimal (8,2)	Not Null
Department	Varchar (50)	Not Null
Hire_Date	Date	Not Null

Emp_ID	Emp_Name	Emp_Salary	Department	Hire_Date
1	John	50000.00	Sales	2022-01-15
2	Jane	60000.00	Marketing	2021-05-10
3	Mike	75000.00	IT	2020-09-20
4	Emily	45000.00	Finance	2023-02-28
5	David	80000.00	IT	2021-11-05

Part – A

1. Create a Stored Procedure to Retrieve Employee Information.

2. Create a Stored Procedure to Update Employee Salary.
3. Create a Stored Procedure to Calculate Average Salary in a Department.
4. Create a Stored Procedure to Delete Inactive Employees.
5. Create a Stored Procedure to Generate Monthly Salary Report.

Part – B

6. Create a Stored Procedure to Get Highest Paid Employee.
7. Create a Stored Procedure to Get Employees Hired in a Specific Year.
8. Create a Stored Procedure to Calculate Total Salary Expense.
9. Create a Stored Procedure to Update Employee Department.
10. Create a Stored Procedure to Get Employees with a Specific Salary Range.

Part – C

11. Create a Stored Procedure to Get Employees by Name.
12. Create a Stored Procedure to Get Employees with the Longest Tenure.
13. Create a Stored Procedure to Calculate the Total Number of Employees in Each Department.
14. Create a Stored Procedure to Delete Employees by Salary Range.
15. Create a Stored Procedure to Calculate the Average Salary for Each Department.

Lab-12

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

Part – A

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.

Part – B

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.

Part – C

6. Create a function which returns minimum salary of female employee.
7. Create a function which count unique city from employee table.

8. Create a Scalar-valued function that returns the name combined with salary of an employee based on their employee id and displayed output like 'Roy having 3500 salary'.

Table Valued Functions

Part – A

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.

Part – B

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.

Part – C

6. Create a function to get employees from a given city.
7. Create a function that displays employees with a salary greater than a specified amount.
8. Create a function to get employees who joined after a given specified date.

Lab-13

User Defined Functions (UDF)

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

Employee	
Column_Name	**DataType**
Employee_ID	Int, Primary Key
First_Name	Varchar (100)
Last_Name	Varchar (50)
Age	Int
Departemt	Varchar (50)

Employee_ID	First_Name	Last_Name	Age	Department
1	John	Doe	30	HR
2	Jane	Smith	25	Finance
3	Michael	Johnson	35	IT
4	Emily	Williams	28	Marketing
5	Robert	Brown	22	IT

From the above given table perform the following queries:

Part – A

1. Create UDF to get the full name of an employee.
2. Create UDF to calculate the age of an employee based on the birth year.
3. Create UDF to get the number of employees in a specific department.
4. Create UDF to concatenate the first name and last name with a custom separator.
5. Create UDF to check if an employee is part of the IT department.

Part – B

6. Create UDF to convert age into a friendly message.
7. Create UDF to get the first initial of an employee's first name.
8. Create UDF to find the average age of employees in a department.
9. Create UDF to check if an employee exists in the table.
10. Create UDF to get the last name in uppercase.

Part – C

11. Create UDF to get the full name and department of an employee
12. Create UDF to check if an employee is older than a specific age.
13. Create UDF to get the number of employees older than a specific age.
14. Create UDF to check if an employee's first name starts with a specific letter.

15. Create UDF to calculate the years of experience based on the current year and an employee's starting year.

Lab-14 **Practice on View, Stored Procedures and User Defined Functions (UDFs)**

Create following tables under **VIEW_SP_UDF_SQL** database as per following data.

Department		
Column_Name	**DataType**	**Constraints**
DepartmentID	Int	Primary Key
DepartmentName	Varchar (100)	Not Null, Unique

DepartmentID	DepartmentName
1	HR
2	IT
3	Finance
4	Marketing

Employee		
Column_Name	**DataType**	**Constraints**
EmployeeID	Int	Primary Key
FirstName	Varchar (50)	Not Null
LastName	Varchar (50)	Not Null
Age	Int	Not Null
DepartmentID	Int	Foreign Key,Null

EmployeeID	FirstName	LastName	Age	DepartmentID
1	Ram	Charan	30	1
2	Laxman	Prasad	25	2
3	Siya	Gupta	28	1
4	Ankit	Trivedi	32	2

From the above given tables perform the following queries:

1. Create a simple view BasicEmpInfo that displays information of employees including EmployeeID, FirstName, LastName and Age.
2. Create a complex view that shows the department wise employee count.
3. Create a stored procedure that retrieves employee information based on the department name.
4. Create a scalar-valued function that calculates the average age of employees in a specific department.
5. Create a table-valued function that retrieves all employees from a specific department.
6. Insert one record in BasicEmpInfo view (5, 'Pavan', 'Kumar', 45, 3).
7. Update age 31 to employee Ram in BasicEmpInfo view.
8. Delete data from BasicEmpInfo view whose age is between 20 to 25.
9. Drop BasicEmpInfo view from the database.
10. Create a complex view that shows the employees information along with their age whose department is IT and age is more than 25.
11. Create a stored procedure that displays information of top 3 employee with their department name.
12. Create scalar-valued function that counts total number of employees works in IT department.
13. Create table-valued function that shows employees whose name start with A to R and department name is HR.
14. Create a complex view that displays employee count having no department.
15. Create a stored procedure that displays department information having no employees.