Lab 1 **Perform basic SQL commands like Create, Insert, Select**

Database Name: Course_Rollno (Example: MCA_101)
<u>Note: Create all the tables under above database.</u>
Create following tables and insert the data into tables using query as shown below.

DEPOSIT

Column_Name	DataType
ACTNO	INT
CNAME	VARCHAR(50)
BNAME	VARCHAR(50)
AMOUNT	DECIMAL(8,2)
ADATE	DATETIME

ACTNO	CNAME	BNAME	AMOUNT	ADATE
101	ANIL	VRCE	1000.00	1-3-95
102	SUNIL	AJNI	5000.00	4-1-96
103	MEHUL	KAROLBAGH	3500.00	17-11-95
104	MADHURI	CHANDI	1200.00	17-12-95
105	PRAMOD	M.G. ROAD	3000.00	27-3-96
106	SANDIP	ANDHERI	2000.00	31-3-96
107	SHIVANI	VIRAR	1000.00	5-9-95
108	KRANTI	NEHRU PLACE	5000.00	2-7-95
109	MINU	POWAI	7000.00	10-8-95

BRANCH

Column_Name	DataType
BNAME	VARCHAR(50)
CITY	VARCHAR(50)

BNAME	CITY
VRCE	NAGPUR
AJNI	NAGPUR
KAROLBAGH	DELHI
CHANDI	DELHI
DHARAMPETH	NAGPUR
M.G. ROAD	BANGLORE
ANDHERI	BOMBAY
VIRAR	BOMBAY
NEHRU PLACE	DELHI
POWAI	BOMBAY

CUSTOMERS

Column_Name	DataType
CNAME	VARCHAR(50)
CITY	VARCHAR(50)

CNAME	CITY
ANIL	CALCUTTA
SUNIL	DELHI
MEHUL	BARODA
MANDAR	PATNA
MADHURI	NAGPUR
PRAMOD	NAGPUR
SANDIP	SURAT
SHIVANI	BOMBAY
KRANTI	BOMBAY
NAREN	BOMBAY

BORROW

Column_Name	DataType
LOANNO	INT
CNAME	VARCHAR(50)
BNAME	VARCHAR(50)
AMOUNT	DECIMAL(8,2)

LOANNO	CNAME	BNAME	AMOUNT
201	ANIL	VRCE	1000.00
206	MEHUL	AJNI	5000.00
311	SUNIL	DHARAMPETH	3000.00
321	MADHURI	ANDHERI	2000.00
375	PRAMOD	VIRAR	8000.00
481	KRANTI	NEHRU PLACE	3000.00

<u>SELECT Operation</u>
Part – A:
From the above given tables perform the following queries:

1. Retrieve all data from table DEPOSIT.

2. Retrieve all data from table BORROW.
3. Display Account No, Customer Name & Amount from DEPOSIT.
4. Display Loan No, Amount from BORROW.
5. Display loan details of all customers who belongs to ‘ANDHERI’ branch from borrow table.
6. Give account no and amount of depositor, whose account no is equals to 106 from deposit table.
7. Give name of borrowers having amount greater than 5000 from borrow table.
8. Give name of customers who opened account after date '1-12-95' from deposit table.
9. Display name of customers whose account no is less than 105 from deposit table.
10. Display name of customer who belongs to either ‘NAGPUR’ or ‘DELHI’ from customer table. (USE OR & IN)
11. Display name of customers with branch whose amount is greater than 4000 and account no is less than 105 from deposit table.
12. Find all borrowers whose amount is greater than equals to 3000 & less than equals to 8000 from borrow table. (USE AND & BETWEEN)
13. Find all depositors who do not belongs to ‘ANDHERI’ branch from deposit table.
14. Display Account No, Customer Name & Amount of such customers who belongs to ‘AJNI’, ‘KAROLBAGH’ Or ‘M.G. ROAD’ and Account No is less than 104 from deposit table.
15. Display all loan no, customer from borrow table does not belong to ‘VIRAR’ or ‘AJNI’ branch. (use NOT IN)
16. Display all the customer’s name other than ‘MINU’ from deposit table (Use: NOT, <>, !=)
17. Display customer name from deposit table whose branch name is not available. (NULL)
18. Retrieve all unique branches using DISTINCT. (Use Branch Table)
19. Retrieve first 50% record from borrow table.
20. Retrieve first five account number from deposit table.

Part – B:

1. Display all the details of first five customers from deposit table.
2. Display all the details of first three depositors from deposit table whose amount is greater than 1000.
3. Display Loan No, Customer Name of first five borrowers whose branch name does not belongs to ‘ANDHERI’ from borrow table.
4. Select all details with account numbers not in the range 105 to 109 in deposit table.
5. Select all records from BORROW where the amount is greater than 1000 and less than or equal to 7000, and the loan number is between 250 and 600

Part – C:

1. Display all the detail of customer who deposited more than 5000 without using * from deposit table.
2. Retrieve all unique customer names with city. (Use Customer table)
3. Retrieve records from the BORROW table where the loan amount is greater than 3000 and the loan number is not a multiple of 3.
4. Retrieve records from the DEPOSIT table where amount is greater than 2000 also account number is between 100 and 110 and date is after '1-MAR-1995' or before '27-MAR-1996'.
5. Retrieve all odd/even value loan number from Borrow table

Lab 2 **Perform SQL queries for Update, Alter, Rename, Delete, Truncate, Drop**

Update Operation

Part – A:

From the above given tables perform the following queries (UPDATE Operation):

1. Update deposit amount of all customers from 3000 to 5000. (Use **Deposit Table**)
2. Change branch name of ANIL from VRCE to C.G. ROAD. (Use **Borrow Table**)
3. Update Account No of SANDIP to 111 & Amount to 5000. (Use **Deposit Table**)
4. Update amount of KRANTI to 7000. (Use **Deposit Table**)

5. Update branch name from ANDHERI to ANDHERI WEST. (Use **Branch Table**)
6. Update branch name of MEHUL to NEHRU PALACE. (Use **Deposit Table**)
7. Update deposit amount of all depositors to 5000 whose account no between 103 & 107. (Use **Deposit Table**)
8. Update ADATE of ANIL to 1-4-95. (Use **Deposit Table**)
9. Update the amount of MINU to 10000. (Use **Deposit Table**)
10. Update deposit amount of PRAMOD to 5000 and ADATE to 1-4-96 (Use **Deposit Table**)

Part – B:

1. Give 10% Increment in Loan Amount. (Use **Borrow Table**)
2. Customer deposits additional 20% amount to their account, update the same. (Use **Deposit Table**)
3. Increase Amount by 1000 in all the account. (Use **Deposit Table**)
4. Update the BORROW table to set the amount to 7000 and the branch name to 'CENTRAL' where the customer name is 'MEHUL' and the loan number is even.
5. Update the DEPOSIT table to set the date to '2022-05-15' and the amount to 2500 for all accounts in 'VRCE' and with an account number less than 105.

Part – C:

1. Update amount of loan no 321 to *NULL.* (Use **Borrow Table**)
2. Update branch name of KRANTI to NULL (Use **Borrow Table**)
3. Display the name of borrowers whose Loan number is *NULL*. (Use **Borrow Table**)
4. Display the Borrowers whose having branch. (Use **Borrow Table**)
5. Update the Loan Amount to 5000, Branch to VRCE & Customer Name to Darshan whose loan no is 481. (Use **Borrow Table**)
6. Update the Deposit table and set the date to 01-01-2021 for all the depositor whose amount is less than 2000.
7. Update the Deposit table and set the date to NULL & Branch name to 'ANDHERI whose Account No is 110.

Alter, Rename Operation

Part – A:

Use Deposit table of lab-1.

DEPOSIT	
Column_Name	**DataType**
ACTNO	INT
CNAME	VARCHAR(50)
BNAME	VARCHAR(50)
AMOUNT	DECIMAL(8,2)
ADATE	DATETIME

From the above given tables perform the following queries (ALTER, RENAME Operation):

1. Add two more columns City VARCHAR (20) and Pincode INT.
2. Add column state VARCHAR(20).
3. Change the size of CNAME column from VARCHAR (50) to VARCHAR (35).
4. Change the data type DECIMAL to INT in amount Column.
5. Delete Column City from the DEPOSIT table.
6. Rename Column ActNo to ANO.
7. Change name of table DEPOSIT to DEPOSIT_DETAIL.

Part – B:

1. Rename Column ADATE to AOPENDATE OF DEPOSIT_DETAIL table.
2. Delete Column AOPENDATE from the DEPOSIT_DETAIL table.
3. Rename Column CNAME to CustomerName.
4. Add Column country.

Part – C:

Create following table using query according to the definition.

STUDENT_DETAIL	
Column_Name	**DataType**
Enrollment_No	VARCHAR(20)
Name	VARCHAR(25)
CPI	DECIMAL(5,2)
Birthdate	DATETIME

From the above given tables perform the following queries (ALTER, RENAME Operation):

1. Add two more columns City VARCHAR (20) (Not null) and Backlog INT (Null).
2. Add column department VARCHAR (20) Not Null.
3. Change the size of NAME column of student_detail from VARCHAR (25) to VARCHAR (35).
4. Change the data type DECIMAL to INT in CPI Column.
5. Delete Column City from the student_detail table.
6. Rename Column Enrollment_No to ENO.
7. Change name of table student_detail to STUDENT_MASTER.

DELETE, Truncate, Drop Operation

Part – A:

Use Deposit_Detail table (Altered table of DEPOSIT)

DEPOSIT_DETAIL	
Column_Name	**DataType**
ANO	INT
CustomerName	VARCHAR(35)
BNAME	VARCHAR(50)
AMOUNT	INT
PINCODE	INT

1. Delete all the records of DEPOSIT_DETAIL table having amount less than and equals to 4000.
2. Delete all the accounts CHANDI BRANCH.
3. Delete all the accounts having account number (ANO) is greater than 102 and less than 105.
4. Delete all the accounts whose branch is ‘AJNI’ or ‘POWAI’
5. Delete all the accounts whose account number is NULL.
6. Delete all the remaining records using Delete command.
7. Delete all the records of Deposit_Detail table. (Use **Truncate**)
8. Remove Deposit_Detail table. (Use **Drop**)

Part – B:

Create following table using query according to the definition.

EMPLOYEE_MASTER	
Column_Name	**DataType**
EmpNo	INT
EmpName	VARCHAR(25)
JoiningDate	DATETIME
Salary	DECIMAL (8,2)
City	VARCHAR(20)

Insert the following records in the EMPLOYEE_MASTER table.

EmpNo	EmpName	JoiningDate	Salary	City
101	Keyur	5-1-02	12000.00	Rajkot
102	Hardik	15-2-04	14000.00	Ahmedabad
103	Kajal	14-3-06	15000.00	Baroda

104	Bhoomi	23-6-05	12500.00	Ahmedabad
105	Harmit	15-2-04	14000.00	Rajkot
106	Mitesh	25-9-01	5000.00	Jamnagar
107	Meera	Null	7000.00	Morbi
108	Kishan	6-2-03	10000.00	NULL

From the above given tables perform the following queries (DELETE Operation):

1. Delete all the records of Employee_MASTER table having salary greater than and equals to 14000.
2. Delete all the Employees who belongs to 'RAJKOT' city.
3. Delete all the Employees who joined after 1-1-2007.
4. Delete the records of Employees whose joining date is null and Name is not null.
5. Delete the records of Employees whose salary is 50% of 20000.
6. Delete the records of Employees whose City Name is not empty.
7. Delete all the records of Employee_MASTER table. (Use **Truncate**)
8. Remove Employee_MASTER table. (Use **Drop**)

Part – C:

1. Summarize Delete, Truncate and Drop

Lab 3 **Perform SQL queries for Like operator**

Part – A:

Create following table using query according to the definition.

STUDENT	
Column_Name	**DataType**
StuID	INT
FirstName	VARCHAR(25)
LastName	VARCHAR(25)
Website	VARCHAR(50)
City	VARCHAR(25)
Address	VARCHAR(100)

Insert the following records in the STUDENT table.

StuID	FirstName	LastName	Website	City	Address
1011	Keyur	Patel	techonthenet.com	Rajkot	A-303 'Vasant Kunj', Rajkot
1022	Hardik	Shah	digminecraft.com	Ahmedabad	"Ram Krupa", Raiya Road
1033	Kajal	Trivedi	bigactivities.com	Baroda	Raj bhavan plot, near garden
1044	Bhoomi	Gajera	checkyourmath.com	Ahmedabad	"Jig's Home", Narol
1055	Harmit	Mitel	@me.darshan.com	Rajkot	B-55, Raj Residency
1066	Ashok	Jani	NULL	Baroda	A502, Club House Building

From the above given tables perform the following queries (LIKE Operation):

1. Display the name of students whose name starts with 'k'.
2. Display the name of students whose name consists of five characters.
3. Retrieve the first name & last name of students whose city name ends with 'a' & contains six characters.
4. Display all the students whose last name ends with 'tel'.
5. Display all the students whose first name starts with 'ha' & ends with't'.
6. Display all the students whose first name starts with 'k' and third character is 'y'.
7. Display the name of students having no website and name consists of five characters.
8. Display all the students whose last name consist of 'jer'.
9. Display all the students whose city name starts with either 'r' or 'b'.
10. Display all the name students having websites.
11. Display all the students whose name starts from alphabet A to H.

12. Display all the students whose name’s second character is vowel.
13. Display the name of students having no website and name consists of minimum five characters.
14. Display all the students whose last name starts with ‘Pat’.
15. Display all the students whose city name does not starts with ‘b’.
16. Display all the students whose student ID ends with digit.
17. Display all the students whose address does not contain any digit.
18. Find students whose first name starts with 'B', last name ends with 'A', and their website contains either 'math' or 'science'. Ensure that their city does not start with 'B'.
19. Retrieve students who have 'Shah' in their last name and whose city ends with 'd'. Additionally, their website should be either null or contain 'com'.
20. Select students whose first and second character is a vowel. Their city should start with 'R' and they must have a website containing '.com'.

Part – B:

1. Display all the students whose name’s second character is vowel and of and start with H.
2. Display all the students whose last name does not ends with ‘a’.
3. Display all the students whose first name starts with consonant.
4. Retrieve student details whose first name starts with 'K', last name ends with 'tel', and either their website contains 'tech' or they live in a city starting with 'R'.
5. Retrieve students whose address contains a hyphen '-' and whose city starts with either 'R' or 'B'. They must have a website that ends with '.com' and their first name should not start with 'A'.

Part – C:

1. Display all the students whose address contains single quote or double quote.
2. Find students whose city does not contain the letter 'S' and their address contains either single or double quotes. Their last name should start with 'P' and they must have a website that contains 'on'.

Lab 4 **Perform SQL queries for Aggerate function and group by (without having)**

Part – A:
Create table and inset records as per below.

EMP

EID	EName	Department	Salary	JoiningDate	City	Gender
101	Rahul	Admin	56000	1-Jan-90	Rajkot	Male
102	Hardik	IT	18000	25-Sep-90	Ahmedabad	Male
103	Bhavin	HR	25000	14-May-91	Baroda	Male
104	Bhoomi	Admin	39000	8-Feb-91	Rajkot	Female
105	Rohit	IT	17000	23-Jul-90	Jamnagar	Male
106	Priya	IT	9000	18-Oct-90	Ahmedabad	Female
107	Bhoomi	HR	34000	25-Dec-91	Rajkot	Female

1. Display the Highest, Lowest, Label the columns Maximum, Minimum respectively.
2. Display Total, and Average salary of all employees. Label the columns Total_Sal and Average_Sal, respectively.
3. Find total number of employees of EMPLOYEE table.
4. Find highest salary from Rajkot city.
5. Give maximum salary from IT department.
6. Count employee whose joining date is after 8-feb-91.
7. Display average salary of Admin department.
8. Display total salary of HR department.
9. Count total number of cities of employee without duplication.
10. Count unique departments.
11. Give minimum salary of employee who belongs to Ahmedabad.

12. Find city wise highest salary.
13. Find department wise lowest salary.
14. Display city with the total number of employees belonging to each city.
15. Give total salary of each department of EMP table.
16. Give average salary of each department of EMP table without displaying the respective department name.
17. Count the number of employees for each department in every city.
18. Calculate the total salary distributed to male and female employees.
19. Give city wise maximum and minimum salary of female employees.
20. Calculate department, city, and gender wise average salary.

Part – B:

1. Count the number of employees living in Rajkot.
2. Display the difference between the highest and lowest salaries. Label the column DIFFERENCE.
3. Display the total number of employees hired before 1st January, 1991.

Part – C:

1. Count the number of employees living in Rajkot or Baroda.
2. Display the total number of employees hired before 1st January, 1991 in IT department.
3. Find the Joining Date wise Total Salaries.
4. Find the Maximum salary department & city wise in which city name starts with ‘R’.

Lab 5 **Perform SQL queries for Group by with having and Order by**

Table: SALES_DATA

Region	Product	Sales_Amount	Year
North America	Watch	1500	2023
Europe	Mobile	1200	2023
Asia	Watch	1800	2023
North America	TV	900	2024
Europe	Watch	2000	2024
Asia	Mobile	1000	2024
North America	Mobile	1600	2023
Europe	TV	1500	2023
Asia	TV	1100	2024
North America	Watch	1700	2024

Part – A:

1. Display Total Sales Amount by Region.
2. Display Average Sales Amount by Product
3. Display Maximum Sales Amount by Year
4. Display Minimum Sales Amount by Region and Year
5. Count of Products Sold by Region
6. Display Sales Amount by Year and Product
7. Display Regions with Total Sales Greater Than 5000
8. Display Products with Average Sales Less Than 10000
9. Display Years with Maximum Sales Exceeding 500
10. Display Regions with at Least 3 Distinct Products Sold.
11. Display Years with Minimum Sales Less Than 1000
12. Display Total Sales Amount by Region for Year 2023, Sorted by Total Amount
13. Find the Region Where 'Mobile' Had the Lowest Total Sales Across All Years.
14. Find the Product with the Highest Sales Across All Regions in 2023.
15. Find Regions Where 'TV' Sales in 2023 Were Greater Than 1000.

Part – B:

1. Display Count of Orders by Year and Region, Sorted by Year and Region
2. Display Regions with Maximum Sales Amount Exceeding 1000 in Any Year, Sorted by Region
3. Display Years with Total Sales Amount Less Than 10000, Sorted by Year Descending
4. Display Top 3 Regions by Total Sales Amount in Year 2024
5. Find the Year with the Lowest Total Sales Across All Regions.

Part – C:

1. Display Products with Average Sales Amount Between 1000 and 2000, Ordered by Product Name
2. Display Years with More Than 1 Orders from Each Region
3. Display Regions with Average Sales Amount Above 1500 in Year 2023 sort by amount in descending.
4. Find out region wise duplicate product.
5. Find out year wise duplicate product.

Lab 6 **Implement SQL In-built functions (Math, String, and Date Functions)**

Math functions

Part – A:

1. Display the result of 5 multiply by 30.
2. Find out the absolute value of -25, 25, -50 and 50.
3. Find smallest integer value that is greater than or equal to 25.2, 25.7 and -25.2.
4. Find largest integer value that is smaller than or equal to 25.2, 25.7 and -25.2.
5. Find out remainder of 5 divided 2 and 5 divided by 3.
6. Find out value of 3 raised to 2nd power and 4 raised 3rd power.
7. Find out the square root of 25, 30 and 50.
8. Find out the square of 5, 15, and 25.
9. Find out the value of PI.
10. Find out round value of 157.732 for 2, 0 and -2 decimal points.
11. Find out exponential value of 2 and 3.
12. Find out logarithm having base e of 10 and 2.
13. Find out logarithm having base b having value 10 of 5 and 100.
14. Find sine, cosine and tangent of 3.1415.
15. Find sign of -25, 0 and 25.
16. Generate random number using function.

Part – B:

Create and insert the following records in the EMP_MASTER table.

EmpNo	EmpName	JoiningDate	Salary	Commission	City	Dept Code
101	Keyur	5-1-02	12000.00	4500	Rajkot	3@g
102	Hardik	15-2-04	14000.00	2500	Ahmedabad	3@
103	Kajal	14-3-06	15000.00	3000	Baroda	3-GD
104	Bhoomi	23-6-05	12500.00	1000	Ahmedabad	1A3D
105	Harmit	15-2-04	14000.00	2000	Rajkot	312A

1. Display the result of Salary plus Commission.
2. Find smallest integer value that is greater than or equal to 55.2, 35.7 and -55.2.
3. Find largest integer value that is smaller than or equal to 55.2, 35.7 and -55.2.
4. Find out remainder of 55 divided 2 and 55 divided by 3.
5. Find out value of 23 raised to 2nd power and 14 raised 3rd power.

Part – C:

1. Retrieve the details of employees whose total earnings (Salary + Commission) are greater than 15000.
2. Find the details of employees whose commission is more than 25% of their salary.

3. List the employees who joined before 2005 and whose total earnings (Salary + Commission) are greater than 15000.
4. Find employees whose total earnings (Salary + Commission) are at least double their salary.

String functions

Part – A:

1. Find the length of following. (I) NULL (II) ‘ hello ’ (III) Blank
2. Display your name in lower & upper case.
3. Display first three characters of your name.
4. Display 3rd to 10th character of your name.
5. Write a query to convert ‘abc123efg’ to ‘abcXYZefg’ & ‘abcabcabc’ to ‘ab5ab5ab5’ using REPLACE.
6. Write a query to display ASCII code for ‘a’,’A’,’z’,’Z’, 0, 9.
7. Write a query to display character based on number 97, 65,122,90,48,57.
8. Write a query to remove spaces from left of a given string ‘hello world ‘.
9. Write a query to remove spaces from right of a given string ‘ hello world ‘.
10. Write a query to display first 4 & Last 5 characters of ‘SQL Server’.
11. Write a query to convert a string ‘1234.56’ to number (Use cast and convert function).
12. Write a query to convert a float 10.58 to integer (Use cast and convert function).
13. Put 10 space before your name using function.
14. Combine two strings using + sign as well as CONCAT ().
15. Find reverse of “Darshan”.
16. Repeat your name 3 times.

Part – B: Perform following queries on EMP_MASTER table.

1. Find the length of EMP Name and City columns.
2. Display EMP Name and City columns in lower & upper case.
3. Display first three characters of EMP Name column.
4. Display 3rd to 10th character of city column.
5. Write a query to display first 4 & Last 5 characters of EMP Name column.

Part – C: Perform following queries on EMP_MASTER table.

1. Put 10 space before EMP Name using function.
2. Combine EMP Name and city columns using + sign as well as CONCAT ().
3. Combine all columns using + sign as well as CONCAT ().
4. Combine the result as < EMP Name > Lives in <City>.
5. Combine the result as ‘EMP no of < EMP Name> is <EmpNo> .
6. Retrieve the names of all employee where the third character of the Name is a vowel.
7. Concatenate the name and city of students who have a name that ends with the letter 'r' and a city that starts with 'R'.

Date Functions

Part – A:

1. Write a query to display the current date & time. Label the column Today_Date.
2. Write a query to find new date after 365 day with reference to today.
3. Display the current date in a format that appears as may 5 1994 12:00AM.
4. Display the current date in a format that appears as 03 Jan 1995.
5. Display the current date in a format that appears as Jan 04, 96.
6. Write a query to find out total number of months between 31-Dec-08 and 31-Mar-09.
7. Write a query to find out total number of hours between 25-Jan-12 7:00 and 26-Jan-12 10:30.
8. Write a query to extract Day, Month, Year from given date 12-May-16.

9. Write a query that adds 5 years to current date.
10. Write a query to subtract 2 months from current date.
11. Extract month from current date using datename () and datepart () function.
12. Write a query to find out last date of current month.
13. Calculate your age in years and months.

Part – B: Perform following queries on EMP_MASTER table.

1. Write a query to find new date after 365 days with reference to JoiningDate.
2. Write a query to find out total number of months between JoiningDate and 31-Mar-09.
3. Write a query to find out total number of years between JoiningDate and 14-Sep-10.

Part – C: Perform following queries on EMP_MASTER table.

1. Write a query to extract Day, Month, Year from JoiningDate.
2. Write a query that adds 5 years to JoiningDate.
3. Write a query to subtract 2 months from JoiningDate.
4. Extract month from JoiningDate using datename () and datepart () function.
5. Select employee who joined between the 1st and 15th of any month in any year.
6. Find employee whose JoiningDate is the last day of any month.
7. List employee whose JoiningDate is during a leap year.

Lab 7 **Implement SQL Joins**

Create below tables as per following data

STU_INFO

Rno(PK)	Name	Branch
101	Raju	CE
102	Amit	CE
103	Sanjay	ME
104	Neha	EC
105	Meera	EE
106	Mahesh	ME

RESULT

Rno(FK)	SPI
101	8.8
102	9.2
103	7.6
104	8.2
105	7.0
107	8.9

EMPLOYEE_MASTER

EmployeeNo	Name	ManagerNo
E01	Tarun	NULL
E02	Rohan	E02
E03	Priya	E01
E04	Milan	E03
E05	Jay	E01
E06	Anjana	E04

Part – A:

1. Combine information from student and result table using cross join or Cartesian product.
2. Perform inner join on Student and Result tables.
3. Perform the left outer join on Student and Result tables.
4. Perform the right outer join on Student and Result tables.
5. Perform the full outer join on Student and Result tables.
6. Display Rno, Name, Branch and SPI of all students.
7. Display Rno, Name, Branch and SPI of CE branch's student only.
8. Display Rno, Name, Branch and SPI of other than EC branch's student only.
9. Display average result of each branch.
10. Display average result of CE and ME branch.
11. Display Maximum and Minimum SPI of each branch.
12. Display branch wise student's count in descending order.

Part – B:

1. Display average result of each branch and sort them in ascending order by SPI.
2. Display highest SPI from each branch and sort them in descending order.

Part – C:

1. Retrieve the names of employee along with their manager's name from the Employee table.

Lab 8 **Implement Complex Joins**

Create following table (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

DEPT

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

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

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. Combine information from Person and Department table using cross join or Cartesian product.
2. Find all persons with their department name
3. Find all persons with their department name & code.
4. Find all persons with their department code and location.
5. Find the detail of the person who belongs to Mechanical department.
6. Final person’s name, department code and salary who lives in Ahmedabad city.
7. Find the person's name whose department is in C-Block.
8. Retrieve person name, salary & department name who belongs to Jamnagar city.
9. Retrieve person’s detail who joined the Civil department after 1-Aug-2001.
10. Display all the person's name with the department whose joining date difference with the current date is more than 365 days.
11. Find department wise person counts.
12. Give department wise maximum & minimum salary with department name.
13. Find city wise total, average, maximum and minimum salary.

14. Find the average salary of a person who belongs to Ahmedabad city.
15. Produce Output Like: <PersonName> lives in <City> and works in <DepartmentName> Department. (In single column)

Part – B:

1. Produce Output Like: <PersonName> earns <Salary> from <DepartmentName> department monthly. (In single column)
2. Find city & department wise total, average & maximum salaries.
3. Find all persons who do not belong to any department.
4. Find all departments whose total salary is exceeding 100000.

Part – C:

1. List all departments who have no person.
2. List out department names in which more than two persons are working.
3. Give a 10% increment in the computer department employee's salary. (Use Update)

Lab 9 **Implement Advanced level Joins**

Create following table (Using Design Mode)

Author

Column Name	Data Type	Constraints
AuthorID	INT	Primary Key
AuthorName	VARCHAR(100)	NOT NULL
Country	VARCHAR(50)	NULL

Publisher

Column Name	Data Type	Constraints
PublisherID	INT	Primary Key
PublisherName	VARCHAR(100)	NOT NULL, UNIQUE
City	VARCHAR(50)	NOT NULL

Book

Column Name	Data Type	Constraints
BookID	INT	Primary Key
Title	VARCHAR(200)	NOT NULL
AuthorID	INT	Foreign Key, AUTHOR(AuthorID), NOT NULL
PublisherID	INT	Foreign Key, PUBLISHER(PublisherID), NOT NULL
Price	DECIMAL(8,2)	NOT NULL
PublicationYear	INT	NOT NULL

AuthorID	AuthorName	Country
1	Chetan Bhagat	India
2	Arundhati Roy	India
3	Amish Tripathi	India
4	Ruskin Bond	India
5	Jhumpa Lahiri	India
6	Paulo Coelho	Brazil
7	Sudha Murty	India

PublisherID	PublisherName	City
1	Rupa Publications	New Delhi
2	Penguin India	Gurugram
3	HarperCollins India	Noida
4	Aleph Book Company	New Delhi

BookID	Title	AuthorID	PublisherID	Price	PublicationYear
101	Five Point Someone	1	1	250.00	2004
102	The God of Small Things	2	2	350.00	1997

103	Immortals of Meluha	3	3	300.00	2010
104	The Blue Umbrella	4	1	180.00	1980
105	The Lowland	5	2	400.00	2013
106	Revolution 2020	1	1	275.00	2011
107	Sita: Warrior of Mithila	3	3	320.00	2017
108	The Room on the Roof	4	4	200.00	1956

From the above given table perform the following queries:

Part – A:

1. List all books with their authors.
2. List all books with their publishers.
3. List all books with their authors and publishers.
4. List all books published after 2010 with their authors and publisher and price.
5. List all authors and the number of books they have written.
6. List all publishers and the total price of books they have published.
7. List authors who have not written any books.
8. Display total number of Books and Average Price of every Author.
9. Lists each publisher along with the total number of books they have published, sorted from highest to lowest.
10. Display number of books published each year.

Part – B:

1. List the publishers whose total book prices exceed 500, ordered by the total price.
2. List most expensive book for each author, sort it with the highest price.

Part – C: Create table as per following schema with proper validation and try to insert data which violate your validation.

1. Emp_info(Eid, Ename, Did, Cid, Salary, Experience)
   Dept_info(Did, Dname)
   City_info(Cid, Cname, Did))
   District(Did, Dname, Sid)
   State(Sid, Sname, Cid)
   Country(Cid, Cname)
2. Insert 5 records in each table.
3. Display employeename, departmentname, Salary, Experience, City, District, State and country of all employees

Lab 10 **Perform SQL queries for Subqueries**

STUDENT_DATA

Rno	Name	City	DID
101	Raju	Rajkot	10
102	Amit	Ahmedabad	20
103	Sanjay	Baroda	40
104	Neha	Rajkot	20
105	Meera	Ahmedabad	30
106	Mahesh	Baroda	10

DEPARTMENT

DID	DName
10	Computer
20	Electrical
30	Mechanical
40	Civil

ACADEMIC

RNO	SPI	BKLOG
101	8.8	0
102	9.2	2
103	7.6	1
104	8.2	4
105	7.0	2
106	8.9	3

Part – A:

1. Display details of students who are from computer department.

2. Displays name of students whose SPI is more than 8.
3. Display details of students of computer department who belongs to Rajkot city.
4. Find total number of students of electrical department.
5. Display name of student who is having maximum SPI.
6. Display details of students having more than 1 backlog.

Part – B:

1. Display name of students who are either from computer department or from mechanical department.
2. Display name of students who are in same department as 102 studying in.

Part – C:

1. Display name of students whose SPI is more than 9 and who is from electrical department.
2. Display name of student who is having second highest SPI.
3. Display city names whose students SPI is 9.2
4. Find the names of students who have more than the average number of backlogs across all students.
5. Display the names of students who are in the same department as the student with the highest SPI.

Lab 11 **Implement Stored Procedure**

Person

Column_Name	DataType	Constraints
PersonID	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

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

PersonID	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

DepartmentID	DepartmentName
1	Admin
2	IT

DesignationID	DesignationName
11	Jobber
12	Welder

3	HR
4	Account

13	Clerk
14	Manager
15	CEO

From the above given tables create Stored Procedures:

Part – A

1. Department, Designation & Person Table's INSERT, UPDATE & DELETE Procedures.
2. Department, Designation & Person Table's SELECTBYPRIMARYKEY
3. Department, Designation & Person Table's (If foreign key is available then do write join and take columns on select list)
4. Create a Procedure that shows details of the first 3 persons.

Part – B

1. Create a Proc that takes the dept name as input and returns a table with all workers working in that dept.
2. Create Procedure that takes department name & designation name as input and returns a table with worker's first name, salary, joining date & department name.
3. 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.
4. Create Procedure which displays department wise maximum, minimum & total salaries.
5. Create Procedure which displays designation wise average & total salaries.

Part – C

1. Create Procedure that Accepts Department Name and Returns Person Count.
2. Create a procedure that takes a salary value as input and returns all workers with a salary greater than input salary value along with their department and designation details.
3. Create a procedure to find the department(s) with the highest total salary among all departments.
4. Create a procedure that takes a designation name as input and returns a list of all workers under that designation who joined within the last 10 years, along with their department.
5. Create a procedure to list the number of workers in each department who do not have a designation assigned.
6. Create a procedure to retrieve the details of workers in departments where the average salary is above 12000.

Lab 12 **Implement Advanced Stored Procedure**

Departments			
ColumnName	**DataType**	**AN | NN**	**Remarks**
DepartmentID	Int	NN	Primary Key
DepartmentName	Varchar(100)	NN	
ManagerID	Int	NN	
Location	Varchar(100)	NN	

Employee			
Column Name	**Data Type**	**AN | NN**	**Remarks**
EmployeeID	Int	NN	Primary Key
FirstName	Varchar(100)	NN	
LastName	Varchar(100)	NN	
DoB	Datetime	NN	
Gender	Varchar(50)	NN	
HireDate	Datetime	NN	
DeptID	Int	NN	Foreign Key
Salary	Decimal(10,2)	NN	

Projects			
ColumnName	**DataType**	**AN | NN**	**Remarks**
ProjectID	Int	NN	Primary Key
ProjectName	Varchar(100)	NN	
StartDate	Datetime	NN	
EndDate	Datetime	NN	
DepartmentID	Int	NN	Foreign Key

DepartmentID	DepartmentName	ManagerID	Location
1	IT	101	New York
2	HR	102	San Francisco
3	Finance	103	Los Angeles
4	Admin	104	Chicago
5	Marketing	105	Miami

EmployeeID	FirstName	LastName	DoB	Gender	HireDate	DeptID	Salary
101	John	Doe	1985-04-12	Male	2010-06-15	1	75000.00
102	Jane	Smith	1990-08-24	Female	2015-03-10	2	60000.00
103	Robert	Brown	1982-12-05	Male	2008-09-25	3	82000.00
104	Emily	Davis	1988-11-11	Female	2012-07-18	4	58000.00
105	Michael	Wilson	1992-02-02	Male	2018-11-30	5	67000.00

ProjectID	ProjectName	StartDate	EndDate	DepartmentID
201	Project Alpha	2022-01-01	2022-12-31	1
202	Project Beta	2023-03-15	2024-03-14	2
203	Project Gamma	2021-06-01	2022-05-31	3
204	Project Delta	2020-10-10	2021-10-09	4
205	Project Epsilon	2024-04-01	2025-03-31	5

Part – A

1. Create Stored Procedure for Employee table As User enters either First Name or Last Name and based on this you must give EmployeeID, DOB, Gender & Hiredate.
2. Create a Procedure that will accept Department Name and based on that gives employees list who belongs to that department.
3. Create a Procedure that accepts Project Name & Department Name and based on that you must give all the project related details.
4. Create a procedure that will accepts any integer and if salary is between provided integer, then those employee list comes in output.
5. Create a Procedure that will accepts a date and gives all the employees who all are hired on that date.

Part – B

1. Create a Procedure that accepts Gender's first letter only and based on that employee details will be served.
2. Create a Procedure that accepts First Name or Department Name as input and based on that employee data will come.
3. Create a procedure that will accepts location, if user enters a location any characters, then he/she will get all the departments with all data.

	Part – C 1. Create a procedure that will accepts From Date & To Date and based on that he/she will retrieve Project related data. 2. Create a procedure in which user will enter project name & location and based on that you must provide all data with Department Name, Manager Name with Project Name & Starting Ending Dates.
Lab 13	**Implement UDF**

Note: for Table valued function use tables of Lab-2

Part – A

1. Write a function to print "hello world".
2. Write a function which returns addition of two numbers.
3. Write a function to check whether the given number is ODD or EVEN.
4. Write a function which returns a table with details of a person whose first name starts with B.
5. Write a function which returns a table with unique first names from the person table.
6. Write a function to print number from 1 to N. (Using while loop)
7. Write a function to find the factorial of a given integer.

Part – B

1. Write a function to compare two integers and return the comparison result. (Using Case statement)
2. Write a function to print the sum of even numbers between 1 to 20.
3. Write a function that checks if a given string is a palindrome

Part – C

1. Write a function to check whether a given number is prime or not.
2. Write a function which accepts two parameters start date & end date, and returns a difference in days.
3. Write a function which accepts two parameters year & month in integer and returns total days each year.
4. Write a function which accepts departmentID as a parameter & returns a detail of the persons.
5. Write a function that returns a table with details of all persons who joined after 1-1-1991.

Lab 14	**Implement Cursor**

Products

Column_Name	DataType	Constraints
Product_id	Int	Primary Key
Product_Name	Varchar (250)	Not Null
Price	Decimal (10,2)	Not Null

Products

Product_id	Product_Name	Price
1	Smatphone	35000
2	Laptop	65000
3	Headphones	5500
4	Television	85000
5	Gaming Console	32000

From the above given tables perform the following queries:

Part - A

1. Create a cursor Product_Cursor to fetch all the rows from a products table.
2. Create a cursor Product_Cursor_Fetch to fetch the records in form of ProductID_ProductName. (Example: 1_Smartphone)
3. Create a Cursor to Find and Display Products Above Price 30,000.
4. Create a cursor Product_CursorDelete that deletes all the data from the Products table.

Part – B

1. Create a cursor Product_CursorUpdate that retrieves all the data from the products table and increases the price by 10%.
2. Create a Cursor to Rounds the price of each product to the nearest whole number.

Part – C

1. Create a cursor to insert details of Products into the NewProducts table if the product is "Laptop" (Note: Create NewProducts table first with same fields as Products table)
2. Create a Cursor to Archive High-Price Products in a New Table (ArchivedProducts), Moves products with a price above 50000 to an archive table, removing them from the original Products table.

Lab 15 **Implement Trigger**

Create following table (Using Design Mode)

PersonInfo

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

PersonLog

Column_Name	DataType	Constraints
PLogID	Int	Primary Key, Auto increment
PersonID	Int	Not Null
PersonName	Varchar (250)	Not Null
Operation	Varchar (50)	Not Null
UpdateDate	Datetime	Not Null

From the above given tables perform the following queries:

Part – A

1. Create a trigger that fires on INSERT, UPDATE and DELETE operation on the PersonInfo table to display a message "Record is Affected."
2. Create a trigger that fires on INSERT, UPDATE and DELETE operation on the PersonInfo table. For that, log all operations performed on the person table into PersonLog.
3. Create an INSTEAD OF trigger that fires on INSERT, UPDATE and DELETE operation on the PersonInfo table. For that, log all operations performed on the person table into PersonLog.
4. Create a trigger that fires on INSERT operation on the PersonInfo table to convert person name into uppercase whenever the record is inserted.
5. Create trigger that prevent duplicate entries of person name on PersonInfo table.
6. Create trigger that prevent Age below 18 years.

Part – B

1. Create a trigger that fires on INSERT operation on person table, which calculates the age and update that age in Person table.

2. Create a Trigger to Limit Salary Decrease by a 10%.

Part – C

1. Create Trigger to Automatically Update JoiningDate to Current Date on INSERT if JoiningDate is NULL during an INSERT.
2. Create DELETE trigger on PersonLog table, when we delete any record of PersonLog table it prints 'Record deleted successfully from PersonLog'.

Lab 16 **Implement Advanced Trigger**

AFTER Trigger

EmployeeDetails

Column_Name	**DataType**	**Constraints**
EmployeeID	Int	Primary Key
EmployeeName	Varchar(100)	Not Null
ContactNo	Varchar(100)	Not Null
Department	Varchar(100)	Not Null
Salary	Decimal(10,2)	Not Null
Joining Date	DateTime	Allow Null

EmployeeLogs

Column_Name	**DataType**	**Constraints**
LogID	Int	Primary Key, Autoincrement
EmployeeID	Int	Not Null
EmployeeName	Varchar(100)	Not Null
ActionPerformed	Varchar(100)	Not Null
ActionDate	DateTime	Not Null

Part – A

1. Create a trigger that fires AFTER INSERT, UPDATE, and DELETE operations on the EmployeeDetails table to display the message "Employee record affected."
2. Create a trigger that fires AFTER INSERT, UPDATE, and DELETE operations on the EmployeeDetails table to log all operations into the EmployeeLog table.
3. Create a trigger that fires AFTER INSERT to automatically calculate the joining bonus (10% of the salary) for new employees and update a bonus column in the EmployeeDetails table.

Part – B

1. Create a trigger to ensure that the JoiningDate is automatically set to the current date if it is NULL.

Part – C

1. Create a trigger that ensure that ContactNo is valid during insert and update (ContactNo length is 10)

Instead of Trigger

Movies

Column_Name	**DataType**	**Constraints**
MovieID	INT	PRIMARY KEY
MovieTitle	VARCHAR(255)	NOT NULL
ReleaseYear	INT	NOT NULL
Genre	VARCHAR(100)	NOT NULL
Rating	DECIMAL(3,1)	NOT NULL
Duration	INT	NOT NULL – (In minutes)

MoviesLog

Column_Name	**DataType**	**Constraints**
LogID	INT	PRIMARY KEY,AUTO INCREMENT

MovieID	INT	NOT NULL
MovieTitle	VARCHAR(255)	NOT NULL
ActionPerformed	VARCHAR(100)	NOT NULL
ActionDate	DATETIME	NOT NULL

From the above given tables perform the following queries:

Part – A

1. Create an INSTEAD OF trigger that fires on INSERT, UPDATE and DELETE operation on the Movies table. For that, log all operations performed on the Movies table into MoviesLog.
2. Create a trigger that only allows to insert movies for which Rating is greater than 5.5 .
3. Create trigger that prevent duplicate 'MovieTitle' of Movies table and log details of it in MoviesLog table.

Part – B

1. Create trigger that prevents to insert pre-release movies.

Part – C

1. Develop a trigger to ensure that the Duration of a movie cannot be updated to a value greater than 120 minutes (2 hours) to prevent unrealistic entries.

Lab 17 **Perform Exception Handling**

Customers

Column_Name	**DataType**	**Constraints**
Customer_id	Int	Primary Key
Customer _Name	Varchar (250)	Not Null
Email	Varchar (50)	Unique

Orders

Column_Name	**DataType**	**Constraints**
Order_id	Int	Primary Key
Customer_id	Int	Foreign Key
Order_date	date	Not Null

From the above given tables perform the following queries:

Part – A

1. Handle Divide by Zero Error and Print message like: Error occurs that is - Divide by zero error.
2. Try to convert string to integer and handle the error using try…catch block.
3. Create a procedure that prints the sum of two numbers: take both numbers as integer & handle exception with all error functions if any one enters string value in numbers otherwise print result.
4. Handle a Primary Key Violation while inserting data into customers table and print the error details such as the error message, error number, severity, and state.
5. Throw custom exception using stored procedure which accepts Customer_id as input & that throws Error like no Customer_id is available in database.

Part – B

1. Handle a Foreign Key Violation while inserting data into Orders table and print appropriate error message.
2. Throw custom exception that throws error if the data is invalid.
3. Create a Procedure to Update Customer’s Email with Error Handling

Part – C

1. Create a procedure which prints the error message that “The Customer_id is already taken. Try another one”.
2. Handle Duplicate Email Insertion in Customers Table.

Lab 18 **Perform SQL queries to implement constraints**

Part – A:

Create below table with following constraints

1. Do not allow SPI more than 10
2. Do not allow Bklog less than 0.
3. Enter the default value as 'General' in branch to all new records IF no other value is specified.
4. Try to update SPI of Raju from 8.80 to 12.
5. Try to update Bklog of Neha from 0 to -1

STU_MASTER

Rno(PK)	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	0
105	Meera	EE	5.52	2
106	Mahesh		4.50	3

Part – B: Create table as per following schema with proper validation and try to insert data which violate your validation.

1. Emp_details(Eid, Ename, Did, Cid, Salary, Experience)
   Dept_details(Did, Dname)
   City_details(Cid, Cname)

Part – C: Create table as per following schema with proper validation and try to insert data which violate your validation.

1. Emp_info(Eid, Ename, Did, Cid, Salary, Experience)
   Dept_info(Did, Dname)
   City_info(Cid, Cname, Did))
   District(Did, Dname, Sid)
   State(Sid, Sname, Cid)
   Country(Cid, Cname)
2. Insert 5 records in each table.
3. Display employeename, departmentname, Salary, Experience, City, District, State and country of all employees.

Lab 19

Perform following queries using use, drop, createcollection, dropcollection, insertOne and insertMany method.
Part - A

1. Create a new database named "Darshan".
2. Create another new database named "DIET".
3. List all databases.
4. Check the current database.
5. Drop "DIET" database.
6. Create a collection named "Student" in the "Darshan" database.
7. Create a collection named "Department" in the "Darshan" database.
8. List all collections in the "Darshan" database.
9. Insert a single document using insertOne into "Department" collection. (Dname:'CE', HOD:'Patel')
10. Insert two document using insertMany into "Department" collection. (Dname:'IT' and Dname:'ICT')
11. Drop a collection named "Department" from the "Darshan" database.
12. Insert a single document using insertOne into "Student" collection.
    (Fields are Name, City, Branch, Semester, Age) Insert your own data.
13. Insert three documents using insertMany into "Student" collection.

(Fields are Name, City, Branch, Semester, Age) Insert your three friend's data.

14. Check whether "Student" collection exists or not.
15. Check the stats of "Student" collection.
16. Drop the "Student" collection.
17. Create a collection named "Deposit".
18. Insert following data in to "Deposit" collection.

Deposit

ACTNO	CNAME	BNAME	AMOUNT	CITY
101	ANIL	VRCE	1000.00	RAJKOT
102	SUNIL	AJNI	5000.00	SURAT
103	MEHUL	KAROLBAGH	3500.00	BARODA
104	MADHURI	CHANDI	1200.00	AHMEDABAD
105	PRMOD	M.G. ROAD	3000.00	SURAT
106	SANDIP	ANDHERI	2000.00	RAJKOT
107	SHIVANI	VIRAR	1000.00	SURAT
108	KRANTI	NEHRU PLACE	5000.00	RAJKOT

19. Display all the documents of "Deposit" collection.
20. Drop the "Deposit" collection.

Part – B

1. Create a new database named "Computer".
2. Create a collection named "Faculty" in the "Computer" database.
3. Insert a below document using insertOne into "Faculty" collection.

Faculty

FID	FNAME	BNAME	SALARY	JDATE
1	ANIL	CE	10000	1-3-95

4. Insert below documents using insertMany into "Faculty" collection.

Faculty

FID	FNAME	BNAME	SALARY	JDATE
2	SUNIL	CE	50000	4-1-96
3	MEHUL	IT	35000	17-11-95
4	MADHURI	IT	12000	17-12-95
5	PRMOD	CE	30000	27-3-96
6	SANDIP	CE	20000	31-3-96
7	SHIVANI	CE	10000	5-9-95
8	KRANTI	IT	50000	2-7-95

5. Display all the documents of "Faculty" collection.
6. Drop the "Faculty" collection.
7. Drop the "Computer" database.

Part – C (Perform following operation using UI)

1. Create a new database named "Computer".
2. Create a collection named "Faculty" in the "Computer" database.
3. Insert a below documents into "Faculty" collection.

Faculty

FID	FNAME	BNAME	SALARY	JDATE
1	ANIL	CE	10000	1-3-95
2	SUNIL	CE	50000	4-1-96
3	MEHUL	IT	35000	17-11-95
4	MADHURI	IT	12000	17-12-95

5	PRMOD	CE	30000	27-3-96
6	SANDIP	CE	20000	31-3-96
7	SHIVANI	CE	10000	5-9-95
8	KRANTI	IT	50000	2-7-95

4. Display all the documents of "Faculty" collection.
5. Drop the "Faculty" collection.
6. Drop the "Computer" database.

Lab 20 **Perform following queries using find, limit, skip and sort method.**
Create and Select Database Named: "**BANK_INFO**"

Deposit (Collection name)

ACTNO	CNAME	BNAME	AMOUNT	ADATE
101	ANIL	VRCE	1000	1-3-95
102	SUNIL	AJNI	5000	4-1-96
103	MEHUL	KAROLBAGH	3500	17-11-95
104	MADHURI	CHANDI	1200	17-12-95
105	PRMOD	M.G. ROAD	3000	27-3-96
106	SANDIP	ANDHERI	2000	31-3-96
107	SHIVANI	VIRAR	1000	5-9-95
108	KRANTI	NEHRU PLACE	5000	2-7-95

From the above given collection perform the following queries using find, limit, skip and sort method:

Part - A

1. Retrieve/Display every document of Deposit collection.
2. Display only one document of Deposit collection. (Use: findOne())
3. Insert following document into Deposit collection. (Use: insertOne())

109	KIRTI	VIRAR	3000	3-5-97

4. Insert following documents into Deposit collection. (Use: insertMany())

110	MITALI	ANDHERI	4500	4-9-95
111	RAJIV	NEHRU PLACE	7000	2-10-98

5. Display all the documents of 'VIRAR' branch from Deposit collection.
6. Display all the documents of Deposit collection whose amount is between 3000 and 5000.
7. Display all the documents of Deposit collection whose amount is greater than 2000 and branch is VIRAR.
8. Display all the documents with CNAME, BNAME and AMOUNT fields from Deposit collection.
9. Display all the documents of Deposit collection on ascending order by CNAME.
10. Display all the documents of Deposit collection on descending order by BNAME.
11. Display all the documents of Deposit collection on ascending order by ACTNO and descending order by AMOUNT.
12. Display only two documents of Deposit collection.
13. Display 3rd document of Deposit collection.
14. Display 6th and 7th documents of Deposit collection.
15. Display the count of documents in Deposit collection.

Part- B

1. Insert following documents into "Student" collection. (Use: insertMany())
   { "_id": 1, "name": "John", "age": 30, "city": "New York", "isActive": true }
   { "_id": 2, "name": "Jane", "age": 25, "city": "Los Angeles", "isActive": false }
   { "_id": 3, "name": "Tom", "age": 35, "city": "Chicago", "isActive": true }
   { "_id": 4, "name": "Lucy", "age": 28, "city": "San Francisco", "isActive": true }
   { "_id": 5, "name": "David", "age": 40, "city": "Miami", "isActive": false }

	{ "_id": 6, "name": "Eva", "age": 23, "city": "Boston", "isActive": true } { "_id": 7, "name": "Nick", "age": 38, "city": "Seattle", "isActive": false } { "_id": 8, "name": "Sophia", "age": 27, "city": "New York", "isActive": true } { "_id": 9, "name": "Liam", "age": 32, "city": "Los Angeles", "isActive": false } { "_id": 10, "name": "Olivia", "age": 29, "city": "San Diego", "isActive": true } 2. Display all documents of "Student" collection. 3. Display all documents of "Student" collection whose age is 30. 4. Display all documents of "Student" collection whose age is greater than 25. 5. Display all documents of "Student" collection whose name is "John" and age is 30. 6. Display all documents of "Student" collection whose age is not equal to 25. 7. Display all documents of "Student" collection whose age is equal to 25 or 30 or 35. (using $or as well as using $in). 8. Display all documents of "Student" collection whose name is "John" or age is 30. 9. Display all documents of "Student" collection whose name is "John" and city is New York. 10. Display name and age of students from "Student" collection whose name is "John" and city is New York. **Part – C** 1. Display name of students from "Student" collection whose age is between to 25 and 35 and sort output by age in ascending order. 2. Display all documents of "Student" collection and sort all the documents by name in ascending order and then by age in descending. 3. Display first five documents of "Student" collection. 4. Display fourth and fifth documents of "Student" collection. 5. Display the name of oldest student from "Student" collection. 6. Display all documents of "Student" collection in such a way that skip the first 2 documents and return the rest documents.
Lab 21	**Perform the following queries using update, delete, rename and createcollection method:** **Part – A (Use collection "Student" created in Lab-2)** 1. Update the age of John's to 31. 2. Update the city of all students from 'New York' to 'New Jersey'. 3. Set isActive to false for every student older than 35. 4. Increment the age of all students by 1 year. 5. Set the city of 'Eva' to 'Cambridge'. 6. Update 'Sophia's isActive status to false. 7. Update the city field of student aged below 30 to 'San Diego'. 8. Rename the age field to years for all documents. 9. Update 'Nick' to make him active (isActive = true). 10. Update all documents to add a new field country with the value 'USA'. 11. Update 'David's city to 'Orlando'. 12. Multiply the age of all students by 2. 13. Unset (remove) the city field for 'Tom'. 14. Add a new field premiumUser and to true for users older than 30. 15. Set isActive to true for 'Jane'. 16. Update isActive field of 'Lucy' to false. 17. Delete a document of 'Nick' from the collection. 18. Delete all students who are inactive (isActive = false). 19. Delete all students who live in 'New York'.

20. Delete all the students aged above 35.
21. Delete a student named ''Olivia'' from the collection.
22. Delete all the students whose age is below 25.
23. Delete the first student whose isActive field is true.
24. Delete all students from 'Los Angeles'.
25. Delete all students who have city field missing.
26. Rename 'city' field to 'location' for all documents.
27. Rename the name field to FullName for 'John'.
28. Rename the isActive field to status for all documents.
29. Rename age to yearsOld for everyone from 'San Francisco' student only.
30. Create a Capped Collection named "Employee" as per follows:
    a. Ecode and Ename are compulsory fields
    b. Datatype of EID is int, Ename is string, Age is int and City is string

    Insert following documents into above "Employee" collection.
    {"Ecode": 1, "Ename": "John"}
    {"Ecode ": 2, "Ename": "Jane", "age": 25, "city": "Los Angeles"}
    {"Ecode ": 3, "Ename": "Tom", "age": 35}
    {"Ecode ": 4, "Ename": "Lucy", "age": 28, "city": "San Francisco", "isActive": true}
    {"Ename": "Dino"}

Part – B Create collection named "Student_data" and insert following 10 documents into it.

Student_data						
ROLLNO	**SNAME**	**DEPARTMENT**	**FEES**	**SEM**	**GENDER**	**CITY**
101	Vina	CE	15000	3	Female	Rajkot
102	Krisha	EC	8000	5	Female	Ahmedabad
103	Priti	Civil	12000	7	Female	Baroda
104	Mitul	CE	15000	3	Male	Rajkot
105	Keshav	CE	15000	3	Male	Jamnagar
106	Zarna	Civil	12000	5	Female	Ahmedabad
107	Nima	EE	9000	5	Female	Rajkot
108	Dhruv	Mechanical	10000	5	Male	Rajkot
109	Krish	Mechanical	10000	7	Male	Baroda
110	Zeel	EE	9000	3	Female	Jamnagar

From the above given "Student_data" collection perform the following queries:

1. Display Female students and belong to Rajkot city.
2. Display students not studying in 3rd sem.
3. Display students whose city is Jamnagar or Baroda. (use: IN)
4. Display first 2 students names who lives in Baroda.
5. Display Male students who studying in 3rd sem.
6. Display sname and city and fees of those students whose roll no is less than 105.
7. Update City of all students from 'Jamnagar' City and Department as 'CE' to 'Surat'.
8. Increase Fees by 500 where the Gender is not 'Female'. (Use: Not)
9. Set the Department of all students from 'EE' and in Sem 3 to 'Electrical'.
10. Update the Fees of students in 'Rajkot' who are male.
11. Change City to 'Vadodara' for students in Sem 5 and with fees less than 10000.
12. Delete all students where the City is 'Ahmedabad' or GENDER is 'Male'.
13. Delete students whose Rollno is not in the list [101, 105, 110].
14. Delete students from the 'Civil' department who are in Sem 5 or Sem 7.

15. Delete all students who are not in the cities 'Rajkot', 'Baroda', or 'Jamnagar'.
16. Delete students whose Rollno is between 105 and 108.
17. Rename the City field to LOCATION for all students.
18. Rename the Department field to Branch where the Fees is less than 10000.
19. Rename Sname to Fullname for students with Rollno in [106, 107, 108].
20. Rename Fees to Tuition_Fees for all students with Fees greater than 9000.
21. Rename Department to Major where the Fees is less than 15000 and Gender is 'Female'.
22. Rename City to Hometown for all students whose SEM is 3 and Department is not 'Mechanical'.

Part – C

1. Create a capped collection named" logs" with a maximum size of 100 KB and a maximum of 10 documents.
2. Insert below 12 log entries into the "logs" collection. Each entry should contain a message, level (e.g., "info", "warning", "error"), and a timestamp field. Use the insertMany() method.
   { message: "System started", level: "info", timestamp: new Date() }
   { message: "Disk space low", level: "warning", timestamp: new Date() }
   { message: "User login", level: "info", timestamp: new Date() }
   { message: "System reboot", level: "info", timestamp: new Date() }
   { message: "Error in module", level: "error", timestamp: new Date() }
   { message: "Memory usage high", level: "warning", timestamp: new Date() }
   { message: "User logout", level: "info", timestamp: new Date() }
   { message: "File uploaded", level: "info", timestamp: new Date() }
   { message: "Network error", level: "error", timestamp: new Date() }
   { message: "Backup completed", level: "info", timestamp: new Date() }
   { message: "Database error", level: "error", timestamp: new Date() }
   { message: "Service started", level: "info", timestamp: new Date() }
3. Perform find method on "logs" collection to ensure only the **last 10 documents** are retained (even though you inserted 12).
4. Insert below 5 more documents and check if the oldest ones are automatically removed.
   { message: "New log entry 1", level: "info", timestamp: new Date() }
   { message: "New log entry 2", level: "info", timestamp: new Date() }
   { message: "New log entry 3", level: "info", timestamp: new Date() }
   { message: "New log entry 4", level: "warning", timestamp: new Date() }
   { message: "New log entry 5", level: "error", timestamp: new Date() }

Lab 22

Perform the following queries using Regex:

Part – A Create collection named "Employee" and insert following 10 documents into it.

employee

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

1. Find employees whose name start with E.

2. Find employees whose name ends with n.
3. Find employees whose name starts with S or M in your collection.
4. Find employees where city starts with A to M in your collection.
5. Find employees where city name ends in 'ney'.
6. Display employee info whose name contains n. (Both uppercase(N) and lowercase(n))
7. Display employee info whose name starts with E and having 5 characters.
8. Display employee whose name start with S and ends in a.
9. Display EID, ENAME, CITY and SALARY whose name starts with 'Phi'.
10. Display ENAME, JOININGDATE and CITY whose city contains 'dne' as three letters in city name.
11. Display ENAME, JOININGDATE and CITY who does not belongs to city London or Sydney.
12. Find employees whose names start with 'J'.
13. Find employees whose names end with 'y'.
14. Find employees whose names contain the letter 'a'.
15. Find employees whose names contain either 'a' or 'e'.
16. Find employees whose names start with 'J' and end with 'n'.
17. Find employees whose CITY starts with 'New'.
18. Find employees whose CITY does not start with 'L'
19. Find employees whose CITY contains the word 'York'.
20. Find employees whose names have two consecutive vowels (a, e, i, o, u).
21. Find employees whose names have three or more letters.
22. Find employees whose names have exactly 4 letters.
23. Find employees whose names start with either 'S' or 'M'.
24. Find employees whose names contain 'il' anywhere.
25. Find employees whose names do not contain 'a'.
26. Find employees whose names contain any digit.
27. Find employees whose names contain exactly one vowel.
28. Find employees whose names start with any uppercase letter followed by any lowercase letter.

Part – B Create collection named "Student" and insert following 10 documents into it.

Student

ROLLNO	SNAME	DEPARTMENT	FEES	SEM	GENDER	CITY
101	Vina	CE	15000	3	Female	Rajkot
102	Krisha	EC	8000	5	Female	Ahmedabad
103	Priti	Civil	12000	7	Female	Baroda
104	Mitul	CE	15000	3	Male	Rajkot
105	Keshav	CE	15000	3	Male	Jamnagar
106	Zarna	Civil	12000	5	Female	Ahmedabad
107	Nima	EE	9000	5	Female	Rajkot
108	Dhruv	Mechanical	10000	5	Male	Rajkot
109	Krish	Mechanical	10000	7	Male	Baroda
110	Zeel	EE	9000	3	Female	Jamnagar

1. Display documents where sname start with K.
2. Display documents where sname starts with Z or D.
3. Display documents where city starts with A to R.
4. Display students' info whose name start with P and ends with i.
5. Display students' info whose department name starts with 'C'.

6. Display name, sem, fees, and department whose city contains 'med' as three letters somewhere in city name.
7. Display name, sem, fees, and department who does not belongs to city Rajkot or Baroda.
8. Find students whose names start with 'K' and are followed by any character.
9. Find students whose names end with 'a'.
10. Find students whose names contain 'ri'. (case-insensitive)

Part – C

1. Find students whose names start with a vowel (A, E, I, O, U).
2. Find students whose CITY ends with 'pur' or 'bad'.
3. Find students whose FEES starts with '1'.
4. Find students whose SNAME starts with 'K' or 'V'.
5. Find students whose CITY contains exactly five characters.
6. Find students whose names do not contain the letter 'e'.
7. Find students whose CITY starts with 'Ra' and ends with 'ot'.
8. Find students whose names contain exactly one vowel.
9. Find students whose names start and end with the same letter.
10. Find students whose DEPARTMENT starts with either 'C' or 'E'.
11. Find students whose SNAME has exactly 5 characters.
12. Find students whose GENDER is Female and CITY starts with 'A'.

Lab 23

Perform the following queries using Regex:

Part – A Create collection named "Customer" and insert following 10 documents into it.

CID	CNAME	EMAIL	GENDER	CITY	BALANCE
1	Rajesh	rajesh@gmail.com	Male	Surat	5000
2	Meena	meena@yahoo.com	Female	Ahmedabad	8000
3	Anil	anil@outlook.com	Male	Vadodara	6000
4	Sneha	sneha@hotmail.com	Female	Rajkot	50000
5	Nikhil	nikhil@gmail.com	Male	Surat	4000
6	Rachna	rachna@yahoo.com	Female	Baroda	9000
7	Gaurav	gaurav@outlook.com	Male	Jamnagar	10000
8	Tanya	tanya@hotmail.com	Female	Mehsana	5500
9	Pushti	pushti@gmail.com	Female	Rajkot	50000
10	Avni	avni@yahoo.com	Female	Morbi	10000

1. Find customers whose CITY does not start with 'S'
2. Find customers whose CITY contains the word 'bad'
3. Find customers whose CNAME has two consecutive vowels.
4. Find customers whose CNAME has three or more letters.
5. Find customers whose EMAIL ends with 'gmail.com'.
6. Find customers whose EMAIL contains 'outlook'
7. Find customers whose name starts with any uppercase letter.
8. Find customers not from Surat or Rajkot.
9. Find customers whose EMAIL starts with 't' and ends with '.com'.
10. Find customers whose CNAME does not contain 'a' or 'e'
11. Find customers whose EMAIL contains exactly one digit.

Part – B Create collection named "Company" and insert following 10 documents into it.

CID	CNAME	EMAIL	INDUSTRY	CITY	EMPLOYEES
1	TechNova	info@technova.com	IT	Bangalore	250

2	GreenWorld	support@greenw.com	Agriculture	Ahmedabad	120
3	SkyHigh Ltd	contact@skyhigh.in	Aviation	Mumbai	300
4	UrbanBuild	info@urbanbuild.org	Construction	Surat	180
5	MediCore	hello@medicore.net	Healthcare	Pune	90
6	FinEdge	info@finedge.co	Finance	Kolkata	200
7	AutoSphere	sales@autos.com	Automotive	Chennai	400
8	EduQuest	info@eduquest.edu	Education	Rajkot	75
9	FoodiesHub	contact@foodies.org	Food	Baroda	60
10	BioPure	info@biopure.bio	Pharma	Hyderabad	150

1. Find companies whose name starts with 'B' or 'F'
2. Find companies located in cities ending with 'pur'
3. Find companies whose name contains the word "Core" Find companies with email addresses starting with "info"
4. Find companies whose INDUSTRY starts with a capital letter followed by 4 lowercase letters
5. Find companies whose CNAME ends with a capital letter
6. Find companies whose CITY starts with any letter from A to K
7. Find companies whose INDUSTRY name has more than 8 letters
8. Find companies whose EMAIL has a number in it
9. Find companies whose name starts and ends with vowels
10. Find companies with CITY names that contain the same letter twice in a row

Part – C

1. Find companies whose email starts with any two letters followed by digits
2. Find companies whose EMAIL includes an underscore _
3. Find companies whose EMAIL domain (after @) starts with 'g' or 'h'
4. Find companies whose CNAME contains a repeating pattern like "ana", "ele", etc.
5. Find companies whose CNAME contains at least 3 vowels
6. Find companies whose EMAIL domain ends with '.com' and starts with 'out'
7. Find companies whose INDUSTRY does not contain any vowels
8. Find companies whose CNAME contains two or more consecutive consonants
9. Find companies whose CNAME has alternating vowels and consonants (at least 4 characters)

Find companies where CITY starts with two same letters (e.g., "Mehsana" doesn't match, but "AAhmedabad" would)

Lab 24

Perform the following queries using Aggregate:

Part – A Create collection named "Student" and insert following 10 documents into it.

Student

ROLLNO	SNAME	DEPARTMENT	FEES	SEM	GENDER	CITY
101	Vina	CE	15000	3	Female	Rajkot
102	Krisha	EC	8000	5	Female	Ahmedabad
103	Priti	Civil	12000	7	Female	Baroda
104	Mitul	CE	15000	3	Male	Rajkot
105	Keshav	CE	15000	3	Male	Jamnagar
106	Zarna	Civil	12000	5	Female	Ahmedabad
107	Nima	EE	9000	5	Female	Rajkot
108	Dhruv	Mechanical	10000	5	Male	Rajkot
109	Krish	Mechanical	10000	7	Male	Baroda
110	Zeel	EE	9000	3	Female	Jamnagar

1. Display distinct city.

2. Display city wise count of number of students.
3. Display sum of salary in your collection.
4. Display average of salary in your document.
5. Display maximum and minimum salary of your document.
6. Display city wise total salary in your collection.
7. Display gender wise maximum salary in your collection.
8. Display city wise maximum and minimum salary.
9. Display count of persons lives in Sydney city in your collection.
10. Display average salary of New York city.
11. Count the number of male and female students in each Department
12. Find the total Fees collected from each Department.
13. Find the minimum Fees paid by male and female students in each City.
14. Sort students by Fees in descending order and return the top 5.
15. Group students by City and calculate the average Fees for each city, only including cities with more than 1 student.
16. Filter students from CE or Mechanical department, then calculate the total Fees.
17. Count the number of male and female students in each Department.
18. Filter students from Rajkot, then group by Department and find the average Fees for each department.
19. Group by Sem and calculate both the total and average Fees, then sort by total fees in descending order.
20. Find the top 3 cities with the highest total Fees collected by summing up all students' fees in those cities.

Part – B

1. Create a collection named" Stock."
2. Insert below 9 documents into the "Stock" collection.

```
{  "_id": 1,
   "company": "Company-A",
   "sector": "Technology",
   "eps": 5.2,
   "pe": 15.3,
   "roe": 12.8,
   "sales": 300000,
   "profit": 25000
}
{   "_id": 2,
   "company": "Company-B",
   "sector": "Finance",
   "eps": 7.1,
   "pe": 12.4,
   "roe": 10.9,
   "sales": 500000,
   "profit": 55000
}
{   "_id": 3,
   "company": "Company-C",
   "sector": "Retail",
   "eps": 3.8,
   "pe": 22.1,
   "roe": 9.5,
   "sales": 200000,
   "profit": 15000
}
{   "_id": 4,
   "company": "Company-D",
   "sector": "Technology",
   "eps": 5.2,
   "pe": 15.3,
   "roe": 12.8,
   "sales": 300000,
   "profit": 25000
}
{   "_id": 5,
   "company": "Company-E",
   "sector": "Finance",
   "eps": 7.1,
   "pe": 12.4,
   "roe": 10.9,
   "sales": 450000,
   "profit": 40000
}
{   "_id": 6,
   "company": "Company-F",
   "sector": "Healthcare",
   "eps": 3.8,
   "pe": 18.9,
   "roe": 9.5,
   "sales": 500000,
   "profit": 35000
}
{   "_id": 7,
   "company": "Company-G",
   "sector": "Retail",
   "eps": 4.3,
   "pe": 22.1,
   "roe": 14.2,
   "sales": 600000,
   "profit": 45000
}
{
   "_id": 8,
   "company": "Company-H",
   "sector": "Energy",
   "eps": 6.5,
   "pe": 10.5,
   "roe": 16.4,
   "sales": 550000,
       "profit": 50000
     }
     {
       "_id": 9,
       "company": "Company-I",
       "sector": "Consumer Goods",
       "eps": 2.9,
       "pe": 25.3,
       "roe": 7.8,
       "sales": 350000,
       "profit": 20000
   }
```

3. Calculate the total sales of all companies.
4. Find the average profit for companies in each sector.
5. Get the count of companies in each sector/
6. Find the company with the highest PE ratio.
7. Filter companies with PE ratio greater than 20.(Use: Aggregate)
8. Calculate the total profit of companies with sales greater than 250,000.
9. Project only the company name and profit fields.(Use: Aggregate)
10. Find companies where EPS is greater than the average EPS.
11. Group companies by sector and get the maximum sales in each sector.
12. Calculate the total sales and total profit of companies in each sector.
13. Sort companies by profit in descending order.(Use: Aggregate)
14. Find the average ROE across all companies.
15. Group companies by sector and calculate both the minimum and maximum EPS.

Part – C

1. Count the number of companies with profit greater than 30,000.
2. Get the total profit by sector and sort by descending total profit.
3. Find the top 3 companies with the highest sales.
4. Calculate the average PE ratio of companies grouped by sector.
5. Get the sum of sales and profit for each company.
6. Find companies with sales less than 400,000 and sort them by sales.
7. Group companies by sector and find the total number of companies in each sector.
8. Get the average ROE for companies with sales greater than 200,000.
9. Find the maximum profit in each sector.
10. Get the total sales and count of companies in each sector.
11. Project fields where profit is more than 20,000 and only show company and profit.
12. Find companies with the lowest ROE and sort them in ascending order.(Use: Aggregate)

Lab 25

Perform the following queries using Aggregate:

Part – A Create collection named "Faculty" and insert following 10 documents into it.

Faculty						
FID	**FNAME**	**DEPARTMENT**	**SALARY**	**EXPERIENCE**	**GENDER**	**CITY**
201	Aarti	CE	50000	5	Female	Rajkot
202	Bhavesh	EC	45000	7	Male	Ahmedabad
203	Chitra	Civil	40000	6	Female	Baroda
204	Deepak	CE	50000	8	Male	Rajkot
205	Ekta	CE	52000	4	Female	Jamnagar

206	Faizan	Civil	42000	7	Male	Ahmedabad
207	Gita	EE	46000	3	Female	Rajkot
208	Harsh	Mechanical	48000	4	Male	Rajkot
209	Dhruv	EC	45000	5	Male	Morbi
210	Tina	EE	47000	3	Female	Surat

1. Display all distinct departments.
2. Count of faculty in each department.
3. Average salary of each department (without department name)
4. Get faculty with salary between 45000 and 50000
5. Departments having more than 2 faculty members
6. Total salary per department where city is 'Rajkot' and salary > 45000
7. List faculties from departments CE or EC having more than 5 years of experience.
8. Display department and faculty name with highest experience
9. Find second lowest salary
10. City-wise average experience of male faculty
11. Sort by experience descending.
12. Department-wise average experience.
13. Top 3 highest paid faculty.
14. Total salary of faculty in 'Rajkot'.
15. Count of faculty per gender per city.

Part – B

1. Find faculty with second highest experience in 'Rajkot'
2. Faculty with experience more than 6 years and salary > 45000.
3. Find departments where total experience > total salary.
4. List of departments with more than 2 faculty.
5. City-wise total and average salary.
6. Find the highest paid faculty per department.
7. Find department with the most faculty.
8. Find the department with the highest average salary.
9. Find the gender distribution in each city.
10. Get cities where average experience is more than 6 years.

Part – C

1. Find second highest paid faculty.
2. Find department with most experienced faculty member.
3. Calculate the overall experience-to-salary ratio (E/S ratio)
4. Compare salary differences by gender
5. Show top 3 cities with highest average salary.
6. City-wise faculty count and average salary for each gender.

Lab 26

Perform the following queries using Aggregate:

Part – A Create collection named "Shopping_Order" and insert following 10 documents into it.

Shopping_Order

OID	CUSTOMER	PRODUCT	CATEGORY	PRICE	QTY	Total	Date	Status
101	Riya Shah	iPhone 13	Electronics	70000	1	70000	2025-05-10	Delivered
102	Amit Patel	T-shirt	Fashion	500	3	1500	2025-05-12	Shipped
103	Nidhi Desai	Microwave	Appliances	9500	1	9500	2025-05-15	Cancelled

104	Raj Mehta	Running Shoes	Footwear	3000	2	6000	2025-05-16	Delivered
105	Zarna Modi	Book Set	Stationery	1200	1	1200	2025-05-18	Delivered
106	Mehul Joshi	Bluetooth Speaker	Electronics	2500	1	2500	2025-05-20	Shipped
107	Sneha Rana	Handbag	Fashion	1800	2	3600	2025-05-22	Delivered
103	Nidhi Desai	Laptop	Electronics	65000	1	65000	2025-05-19	Delivered
101	Riya Shah	iPhone 13	Electronics	70000	1	70000	2025-05-10	Delivered
102	Amit Patel	T-shirt	Fashion	500	3	1500	2025-05-12	Shipped

1. Count unique customers as per category.
2. Find the highest order value of each Customer.
3. Determine average order per product category.
4. Group by customer and product,count total units ordered.
5. Group by product, get average quantity and total revenue.
6. As per day get total products sold.
7. Filter only those Customers with total spend > 5000.
8. Average order value per status.
9. Find Top 3 Highest sold Product(revenue).
10. How many quantity of Product's order is cancelled.
11. Count how many products each customer has bought across all categories.
12. Get the Number of orders placed per Customer per status (Delivered,Shipped,etc..)

Part – B

1. Find the Customer who has placed the most number of high-value orders(value>5000).
2. List customers who bought from multiple categories.
3. Most frequently ordered product per Customer.
4. List customers who made multiple orders on same date.
5. Find category with secnd lowest revenue(total price).
6. Customer with highest average spend per order.
7. Find customer with highest total quantity of Fashion iteams.
8. Find top 3 customers who spent the most in Electronics.

Part – C

1. Calculate average price per unit sold for each category (price weighted by qty).
2. Which category is delivered in 5 th Month.
3. For each customer show the Number of Products and total amount for delivered orders only.
4. Get daily total revenue and Number of orders.
5. Find customers who have repeated orders of the same product (Product-wise repeat buyers).
6. Compute the total sales and number of products sold per day and sort by highest sales day.
7. Find the highest quantity of product which is delivered at Customer.

Lab 27 **Perform the following queries using Aggregate:**

Part – A Create collections as per below and insert documents into it.

Project

PROJECT_ID	PROJECT_NAME	DEPARTMENT	BUDGET	MANAGER_ID	STATUS	START_DATE
201	Alpha Launch	Marketing	120000	101	Active	2023-01-15
202	Beta Upgrade	IT	80000	102	Completed	2022-05-01
203	Gamma Research	R&D	150000	103	Active	2023-06-10
204	Delta Expansion	Marketing	90000	101	Active	2023-03-20

205	Epsilon Launch	IT	110000	104	Active	2023-04-01
206	Zeta Project	R&D	130000	103	Completed	2022-12-10
207	Eta Upgrade	Finance	70000	105	Active	2023-02-15

Manager

MANAGER_ID	NAME	EMAIL	PHONE
101	John Smith	john@example.com	123-456-7890
102	Alice Brown	alice@example.com	234-567-8901
103	Carol White	carol@example.com	345-678-9012
104	David Green	david@example.com	456-789-0123
105	Emma Black	emma@example.com	567-890-1234

Department

DEPARTMENT	HEAD_MANAGER_ID
Marketing	101
IT	102
R&D	103
Finance	105
HR	106

1. List all projects with manager details.
2. Show project name and manager name.
3. Find all active projects with manager info.
4. List all completed projects with manager name
5. Show project budget and manager email
6. Get all manager names who manage projects.
7. List manager phone numbers for all projects
8. List project names that started after Jan 1, 2023.
9. Show project name and the department’s head manager ID.
10. List all departments and the count of managers heading them.

Part – B

1. Count how many projects each manager handles.
2. List project and manager where budget > 100000.
3. Show project names with their department head's ID.
4. List managers with total budget handled.
5. Get project name, department, and manager name.
6. Show manager with highest total project budget.
7. List all manager names working in Marketing or IT
8. Find projects handled by managers whose name starts with 'A'.
9. Get total budget of projects grouped by manager's name.
10. List all managers and the count of projects they are managing.
11. Find managers who do not manage any project.
12. Show the earliest project each manager has started.
13. List all projects with the department and department head name.
14. List each manager and the total budget of their active projects.
15. Count of projects handled by each manager by status
16. List managers who are not assigned to any project.

Part – C

	1. List all projects with manager and department head names. 2. List of managers who are also department heads 3. Count of projects under each department head 4. Department-wise average budget and department head name 5. Show top 3 managers with most budget across departments
Lab 28	**Perform the following queries on Index, Cursor, Schema Validation, Embedded and Multivalued Documents:** **Part – A (Use collection "Stock" created in Lab-24)** 1. Create an index on the company field in the stocks collection. 2. Create a compound index on the sector and sales fields in the stocks collection. 3. List all the indexes created on the stocks collection. 4. Drop an existing index on the company field from the stocks collection. 5. Use a cursor to retrieve and iterate over documents in the stocks collection, displaying each document. 6. Limit the number of documents returned by a cursor to the first 3 documents in the stocks collection. 7. Sort the documents returned by a cursor in descending order based on the sales field. 8. Skip the first 2 documents in the result set and return the next documents using the cursor. 9. Convert the cursor to an array and return all documents from the stocks collection. 10. Create a collection named "Companies" with schema validation to ensure that each document must contains a company field (string) and a sector field (string). **Part – B** 1. Create a collection named "Scripts" with validation for fields like eps, pe, and roe to ensure that they are numbers and required/compulsory fields. 2. Create a collection named "Products" where each product has an embedded document for manufacturer details and a multivalued field for categories that stores an array of category names the product belongs to. • manufacturer details: The manufacturer will be an embedded document with fields like name, country, and establishedYear. • categories: The categories will be an array field that holds multiple values. (i.e. Electronics, Mobile, Smart Devices). **Part – C** 1. Create a collection named "financial_Reports" that requires revenue (a positive number) but allows optional fields like expenses and netIncome (if provided, they should also be numbers). 2. Create a collection named "Student" where each student has name and address are embedded document and mobilenumber and emailaddress are multivalued field that stores an array of values.
Lab 29	**Revision Lab** **CUSTOMER:** • customer_id: Unique ID for each customer. • name: Full name of the customer. • email: Email address of the customer. • phone: Contact number. • location: City where the customer is based. **PRODUCT:** • product_id: Unique ID for each product. • name: Name of the product. • category: Category of the product (e.g., Electronics, Clothing, etc.). • price: Selling price of the product. **SALES:** • sale_id: Unique ID for each sale transaction. • customer_id: References the customer making the purchase.

	• product_id: References the product being purchased. • sale_date: Date of the sale. • quantity: Number of units sold. • total_amount: Total cost of the sale (quantity * price). **Consider above table schema and solve following queries:** 1. Display customers who purchased TV worth price 20,000. 2. Display highest selling item. 3. Create Stored procedure to update the customer's contact information 4. Create stored procedure to retrieve all sales transactions done by "Kairavi". 5. Create UDF to calculate the total revenue generated from a sale of television. 6. Create Trigger to prevent deleting a customer if they have made a purchase. 7. Create Cursor to fetch and display all sales along with customer and product details. 8. Update product price with error handling (price cannot be negative) 9. Regex (MongoDB) 1. Find all customers whose names start with 'J'. 2. Find all products whose names consist of minimum 5 characters. 3. Find all customer whose names start and ends with vowels. 4. Find all customer emails ending with '@gmail.com' 5. Find all product names containing exactly two words. 10. Aggregation (MongoDB) 1. Count the total number of products whose price between 1000 and 5000. 2. Find the category-wise cheapest product. 3. Display the name of customers having same name. 4. Count the product whose name starts with vowel. 5. Display category-wise products count whose name starts with vowels and sort on category in descending order.
Lab 30	**Draw an E-R Diagram of following system:** 1. Library Management System 2. Hospital Management System 3. Online Shopping System 4. University Database 5. Hotel Reservation System 6. Banking System 7. Airline Reservation System 8. Social Networking Site 9. Cinema Ticket Booking System 10. School Attendance System **Design a database of following system:** 1. Library Management System 2. Hospital Management System 3. Online Shopping System 4. University Database 5. Hotel Reservation System 6. Banking System 7. Airline Reservation System 8. Social Networking Site 9. Cinema Ticket Booking System 10. School Attendance System