**S1**

Consider following Relation

**Account (Acc_no, branch_name,balance)**
**Branch(branch_name,branch_city,assets)**
**Customer(cust_name,cust_street,cust_city)**
**Depositor(cust_name,acc_no)**
**Loan(loan_no,branch_name,amount)**
**Borrower(cust_name,loan_no)**

Create above tables with appropriate constraints like primary key, foreign key, not null etc.

1. Find the names of all branches in loan relation.
2. Find all loan numbers for loans made at 'Wadia College' Branch with loan amount > 12000.
3. Find all customers who have a loan from bank. Find their names,loan_no and loan amount.
4. List all customers in alphabetical order who have loan from 'Wadia College' branch.
5. Display distinct cities of branch.

**S2**

Consider following Relation

**Account (Acc_no, branch_name,balance)**
**Branch(branch_name,branch_city,assets)**
**Customer(cust_name,cust_street,cust_city)**
**Depositor(cust_name,acc_no)**
**Loan(loan_no,branch_name,amount)**
**Borrower(cust_name,loan_no)**

Create above tables with appropriate constraints like primary key, foreign key, not null etc.

1. Find all customers who have both account and loan at bank.
2. Find all customers who have an account or loan or both at bank.
3. Find all customers who have account but no loan at the bank.
4. Find average account balance at 'Wadia College' branch.
5. Find no. of depositors at each branch

**P10**

**Trigger**: Write a after trigger for Insert, update and delete event considering following requirement:

Emp(Emp_no, Emp_name, Emp_salary)

a) Trigger should be initiated when salary tried to be inserted is less than Rs.50,000/-
b) Trigger should be initiated when salary tried to be updated for value less than Rs. 50,000/-

Also the new values expected to be inserted will be stored in new table Tracking(Emp_no,Emp_salary).

| | |
|---|---|
| **S3** | Consider following Relation<br>**Account (Acc_no, branch_name,balance)**<br>**Branch(branch_name,branch_city,assets)**<br>**Customer(cust_name,cust_street,cust_city)**<br>**Depositor(cust_name,acc_no)**<br>**Loan(loan_no,branch_name,amount)**<br>**Borrower(cust_name,loan_no)**<br>Create above tables with appropriate constraints like primary key, foreign key, not null etc.<br>1. Find the branches where average account balance > 15000.<br>2. Find number of tuples in customer relation.<br>3. Calculate total loan amount given by bank.<br>4. Delete all loans with loan amount between 1300 and 1500.<br>5. Find the average account balance at each branch<br>6. Find name of Customer and city where customer name starts with Letter P. |
| | |
| **S4** | **SQL Queries:**<br>Create following tables with suitable constraints (primary key, foreign key, not null etc).<br>Insert record and solve the following queries:<br>**Create table Cust_Master(Cust_no, Cust_name, Cust_addr)**<br>**Create table Order(Order_no, Cust_no, Order_date, Qty_Ordered)**<br>**Create Product (Product_no, Product_name, Order_no)**<br>1. List names of customers having 'A' as second letter in their name.<br>2. Display order from Customer no C1002, C1005, C1007 and C1008<br>3. List Clients who stay in either 'Banglore or 'Manglore'<br>4. Display name of customers& the product_name they have purchase<br>5. Create view View1 consisting of Cust_name, Product_name.<br>6. Disply product_name and quantity purchase by each customer<br>7. Perform different joint operation. |
| | |
| **P1** | Write a **PL/SQL code** block to calculate the area of a circle for a value of radius varying from 5 to 9. Store the radius and the corresponding values of calculated area in an empty table named areas, consisting of two columns, radius and area. |
| | |
| **P3** | Write a **PL/SQL block** of code using Cursor that will merge the data available in the newly created table N_Roll Call with the data available in the table O_RollCall. If the data in the first table already exist in the second table, then that data should be skipped. |

| | |
|---|---|
| **S5** | Consider following Relation<br>**Employee(emp_id,employee_name,street,city)**<br>**Works(employee_name,company_name,salary)**<br>**Company(company_name,city)**<br>**Manages(employee_name,manager_name)**<br>Create above tables with appropriate constraints like primary key, foreign key, not null etc.<br>1. Find the names of all employees who work for 'TCS'.<br>2. Find the names and company names of all employees sorted in ascending order of company name and descending order of employee names of that company.<br>3. Change the city of employee working with InfoSys to 'Bangalore'<br>4. Find the names, street address, and cities of residence for all employees who work for 'TechM' and earn more than $10,000.<br>5. Add Column Asset to Company table. |
| | |
| **S6** | Consider following Relation<br>**Employee(emp_id,employee_name,street,city)**<br>**Works(employee_name,company_name,salary)**<br>**Company(company_name,city)**<br>**Manages(employee_name,manager_name)**<br>Create above tables with appropriate constraints like primary key, foreign key, not null etc.<br>1. Change the city of employee working with InfoSys to 'Bangalore'<br>2. Find the names of all employees who earn more than the average salary of all employees of their company. Assume that all people work for at most one company.<br>3. Find the names, street address, and cities of residence for all employees who work for 'TechM' and earn more than $10,000.<br>4. Change name of table Manages to Management.<br>5. Create Simple and Unique index on employee table.<br>6. Display index Information |
| | |
| **P8** | Write a **Row Level Before and After Trigger** on Library table. The System should keep track of the records that are being updated or deleted. The old value of updated or deleted records should be added in Library_Audit table. |
| | |
| **P9** | **Trigger:** Create a row level trigger for the CUSTOMERS table that would fire INSERT or UPDATE or DELETE operations performed on the CUSTOMERS table. This trigger will display the salary difference between the old values and new values. |

|  |  |
|---|---|
| **S7** | Consider following Relation<br>**Account (Acc_no, branch_name,balance)**<br>**Branch(branch_name,branch_city,assets)**<br>**Customer(cust_name,cust_street,cust_city)**<br>**Depositor(cust_name,acc_no)**<br>**Loan(loan_no,branch_name,amount)**<br>**Borrower(cust_name,loan_no)**<br>Execute the following query:<br>1. Create a View1 to display List all customers in alphabetical order who have loan from Pune_Station branch.<br>2. Create View2 on branch table by selecting any two columns and perform insert update delete operations.<br>3. Create View3 on borrower and depositor table by selecting any one column from each table perform insert update delete operations.<br>4. Create Union of left and right joint for all customers who have an account or loan or both at bank<br>5. Create Simple and Unique index.<br>6. Display index Information.<br>7. |
|  |  |
| **S8** | Consider following Relation:<br>**Companies (comp_id, name, cost, year)**<br>**Orders (comp_id, domain, quantity)**<br>Execute the following query:<br>1. Find names, costs, domains and quantities for companies using inner join.<br>2. Find names, costs, domains and quantities for companies using left outer join.<br>3. Find names, costs, domains and quantities for companies using right outer join.<br>4. Find names, costs, domains and quantities for companies using Union operator.<br>5. Create View View1 by selecting both tables to show company name and quantities.<br>6. Create View View2 by selecting any two columns and perform insert update delete operations.<br>7. Display content of View1, View2. |
|  |  |
| **P5** | Write a **PL/SQL Block** to increase the salary of employees by 10% of existing salary, who are having salary less than average salary of organization, whenever such salary updates take place, a record for same is maintained in the increment_salary table.<br>emp(emp_no, salary)<br>increment_salary(emp_no, salary) |

**S9**

**SQL Queries**

Create following tables with suitable constraints. Insert data and solve the following queries:

**CUSTOMERS(CNo, Cname, Ccity, CMobile)**
**ITEMS(INo, Iname, Itype, Iprice, Icount)**
**PURCHASE(PNo, Pdate, Pquantity, Cno, INo)**

1. List all stationary items with price between 400/- to 1000/-
2. Change the mobile number of customer "Gopal"
3. Display the item with maximum price
4. Display all purchases sorted from the most recent to the oldest
5. Count the number of customers in every city
6. Display all purchased quantity of Customer Maya
7. Create view which shows Iname, Price and Count of all stationary items in descending order of price.

**M1**

**Design and Develop MongoDB Queries using CRUD operations:**

Create Employee collection by considering following Fields:

i. Name: Embedded Doc (FName, LName)
ii. Company Name: String
iii. Salary: Number
iv. Designation: String
v. Age: Number
vi. Expertise: Array
vii. DOB: String or Date
viii. Email id: String
ix. Contact: String
x. Address: Array of Embedded Doc (PAddr, LAddr)

Insert at least 5 documents in collection by considering above attribute and execute following queries:

1. Select all documents where the Designation field has the value "Programmer" and the value of the salary field is greater than 30000.
2. Creates a new document if no document in the employee collection contains
   {Designation: "Tester", Company_name: "TCS", Age: 25}
3. Increase salary of each Employee working with "Infosys" 10000.
4. Finds all employees working with "TCS" and reduce their salary by 5000.
5. Return documents where Designation is not equal to "Tester".
6. Find all employee with Exact Match on an Array having Expertise: ['Mongodb','Mysql','Cassandra']

| | |
|---|---|
| **M2** | **Design and Develop MongoDB Queries using CRUD operations:**<br>Create Employee collection by considering following Fields:<br>i. Name: Embedded Doc (FName, LName)<br>ii. Company Name: String<br>iii. Salary: Number<br>iv. Designation: String<br>v. Age: Number<br>vi. Expertise: Array<br>vii. DOB: String or Date<br>viii. Email id: String<br>ix. Contact: String<br>x. Address: Array of Embedded Doc (PAddr, LAddr)<br>Insert at least 5 documents in collection by considering above attribute and execute following queries:<br>1. Final name of Employee where age is less than 30 and salary more than 50000.<br>2. Creates a new document if no document in the employee collection contains<br>{Designation: "Tester", Company_name: "TCS", Age: 25}<br>3. Selects all documents in the collection where the field age has a value less than 30 or the value of the salary field is greater than 40000.<br>4. Find documents where Designation is not equal to "Developer".<br>5. Find _id, Designation, Address and Name from all documents where Company_name is "Infosys".<br>6. Display only FName and LName of all Employees |
| | |
| **P4** | Write a **PL/SQL block** for following requirements and handle the exceptions. Roll no. of students will be entered by the user. Attendance of roll no. entered by user will be checked in the Stud table. If attendance is less than 75% then display the message “Term not granted” and set the status in stud table as “Detained”. Otherwise display message “Term granted” and set the status in stud table as “Not Detained”. **Student (Roll, Name, Attendance, Status)** |
| | |
| **P7** | Create a **stored function** titled '**Age_calc**'.<br>Accept the date of birth of a person as a parameter.<br>Calculate the age of the person in years, months and days e.g. 3 years, 2months, 10 days.<br>Return the age in years directly (with the help of Return statement). The months and days are to be returned indirectly in the form of OUT parameters. |

**M3**

**Design and Develop MongoDB Queries using CRUD operations:**

Create Employee collection by considering following Fields:

i. Emp_id : Number
ii. Name:  Embedded Doc (FName, LName)
iii. Company Name: String
iv. Salary: Number
v. Designation: String
vi. Age: Number
vii. Expertise: Array
viii. DOB: String or Date
ix. Email id: String
x. Contact: String
xi. Address: Array of Embedded Doc (PAddr, LAddr)

Insert at least 5 documents in collection by considering above attribute and execute following queries:

1. Creates a new document if no document in the employee collection contains
   {Designation: "Tester", Company_name: "TCS", Age: 25}
2. Finds all employees working with Company_name: "TCS" and increase their salary by 2000.
3. Matches all documents where the value of the field Address is an embedded document that contains only the field city with the value "Pune" and the field Pin_code with the value "411001".
4. Find employee details who are working as "Developer" or "Tester".
5. Drop Single documents where designation="Developer".
6. Count number of documents in employee collection.

**M6**

**Design MongoDB database and perform following Map reduce operation:**

Create Employee collection by considering following Fields:

i. Name:  Embedded Doc (FName, LName)
ii. Company Name: String
iii. Salary: Number
iv. Designation: String
v. Age: Number
vi. Expertise: Array
vii. DOB: String or Date
viii. Email id: String
ix. Contact: String
x. Address: Array of Embedded Doc (PAddr, LAddr)

Execute the following query:

1. Display the total salary of per company
2. Display the total salary of company Name:"TCS"
3. Return the average salary of company whose address is “Pune".
4. Display total count for “City=Pune”
5. Return count for city pune and age greater than 40.

| | |
|---|---|
| **M4** | **Design and Develop MongoDB Queries using Aggregation operations:**<br>Create Employee collection by considering following Fields:<br>i. Emp_id : Number<br>ii. Name: Embedded Doc (FName, LName)<br>iii. Company Name: String<br>iv. Salary: Number<br>v. Designation: String<br>vi. Age: Number<br>vii. Expertise: Array<br>viii. DOB: String or Date<br>ix. Email id: String<br>x. Contact: String<br>xi. Address: Array of Embedded Doc (PAddr, LAddr)<br>Insert at least 5 documents in collection by considering above attribute and execute following:<br>1. Using aggregation Return Designation with Total Salary is Above 200000.<br>2. Using Aggregate method returns names and _id in upper case and in alphabetical order.<br>3. Using aggregation method find Employee with Total Salary for Each City with Designation="DBA".<br>4. Create Single Field Indexes on Designation field of employee collection<br>5. To Create Multikey Indexes on Expertise field of employee collection.<br>6. Create an Index on Emp_id field, compare the time require to search Emp_id before and after creating an index. (Hint Add at least 10000 Documents)<br>7. Return a List of Indexes on created on employee Collection. |
| | |
| **P6** | Write a **Stored Procedure** namely **proc_Grade** for the categorization of student. If marks scored by students in examination is <=1500 and marks>=990 then student will be placed in distinction category if marks scored are between 989 and 900 categories is first class, if marks 899 and 825 category is Higher Second Class.<br>Write a PL/SQL block for using procedure created with above requirement.<br>Stud_Marks(name, total_marks),<br>Result (Roll,Name, Class) |
| | |
| **C1** | Write a program to implement **MongoDB database connectivity** with PHP /python /Java Implement Database navigation CRUD operations (add, delete, edit etc.) |
| | |

| | |
|---|---|
| **C2** | Implement **MYSQL/Oracle database connectivity** with PHP /python /Java Implement Database navigation operations (add, delete, edit,). |
| | |
| **P2** | Write an **Unnamed PL/SQL** of code for the following requirements: -<br>Schema:<br>Borrower (Rollin, Name, DateofIssue, NameofBook, Status)<br>Fine (Roll_no,Date,Amt)<br>Accept roll_no & name of book from user.<br>Check the number of days (from date of issue).<br>1. If days are between 15 to 30 then fine amounts will be Rs 5 per day.<br>2. If no. of days>30, per day fine will be Rs 50 per day & for days less than 30, Rs. 5 per day.<br>3. After submitting the book, status will change from I to R.<br>4. If condition of fine is true, then details will be stored into fine table. |
| | |
| **M5** | **Design and Develop MongoDB Queries using Aggregation operations:**<br>Create Employee collection by considering following Fields:<br>i. Emp_id : Number<br>ii. Name: Embedded Doc (FName, LName)<br>iii. Company Name: String<br>iv. Salary: Number<br>v. Designation: String<br>vi. Age: Number<br>vii. Expertise: Array<br>viii. DOB: String or Date<br>ix. Email id: String<br>x. Contact: String<br>xi. Address: Array of Embedded Doc (PAddr, LAddr)<br>Insert at least 5 documents in collection by considering above attribute and execute following:<br>1. Using aggregation Return separates value in the Expertise array and return sum of each element of array.<br>2. Using Aggregate method return Max and Min Salary for each company.<br>3. Using Aggregate method find Employee with Total Salary for Each City with Designation="DBA".<br>4. Using aggregation method Return separates value in the Expertise array for employee name where Swapnil Jadhav<br>5. To Create Compound Indexes on Name: 1, Age: -1<br>6. Create an Index on Emp_id field, compare the time require to search Emp_id before and after creating an index. (Hint Add at least 10000 Documents)<br>7. Return a List of Indexes on created on employee Collection. |