**Data Base Management System Lab (310247)**

**Problem Statements**

**Statement -1: View**

* **Create table Employees and Departments as shown below:**

(Use emp\_id as primary key for employees table and dept\_id as a primary key for departments table)

|  |  |  |  |
| --- | --- | --- | --- |
| emp\_id | emp\_name | salary | dept\_id |
| 1 | Ethan Hunt | 5000 | 4 |
| 2 | Tony Montana | 6500 | 1 |
| 3 | Sarah Connor | 8000 | 5 |
| 4 | Rick Deckard | 7200 | 3 |
| 5 | Martin Blank | 5600 | NULL |

|  |  |
| --- | --- |
| dept\_id | Dept\_name |
| 1 | Administration |
| 2 | Customer Service |
| 3 | Finance |
| 4 | Human Resources |
| 5 | Sales |

* **Execute the following queries in MySQL:**

1. Retrieve the id and name of the employees along with their department name (Use Left Join)

SELECT t1.emp\_id, t1.emp\_name, t2.dept\_name

FROM employees AS t1 LEFT JOIN departments AS t2

ON t1.dept\_id = t2.dept\_id;

1. Create View for the above query.
2. Retrieve records from the above created view.
3. Replace the above view to show emp\_id, .emp\_name, dept\_name salary also.
4. Create another view using salary, emp\_id and dept\_id.
5. Insert 3 more records in above created view.
6. Update the above view to set salary =6000 for the employee having emp\_id 1.
7. Retrieve the record from view where dept\_id of employees is NULL.
8. Delete those records from view where employee salary is 8000.
9. Drop the above created view.

**Statement -2: Triggers**

Write a database 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.

**Statement -1: Joins**

* **Create following tables**

Pack\_grades(grade\_id, grade\_name ,min\_price,max\_price)

Sectors(sector\_id,sector\_name)

Packages(pack\_id,speed,strt\_date,monthly\_payment,sector\_id)

Customers(cust\_id, name, start\_date,,city, phone\_num, , monthly\_discount, pack\_id)

**Inner Join**

1. Customers and internet packages (*Customers* & *Packages* tables) –
   1. Write a query to display name, package id and internet speed for all customers with similar package id.
   2. Write a query to display name, package id and internet speed for all customers whose package id equals 22 or 27. Order the query in ascending order by name.
2. Internet packages and sectors –
   1. Display the package id, internet speed, monthly payment and sector name for all packages with similar sector id. (*Packages* and *Sectors* tables).
   2. Display the customer name, package id, internet speed, monthly payment and sector name for all customers with similar package id and sector id (*Customers*, *Packages* and *Sectors* tables).
   3. Display the customer name, package id, internet speed, monthly payment and sector name for all customers in the business sector with similar package id and sector id (*Customers*, *Packages* and *Sectors* tables).

**Outer Join**

Customers and internet packages (Customers and Packages tables)

a. Modify 1(a) query to display all customers, including those without any internet package.

c. Modify 1(a) query to display all packages, including those without any customers.

d. Modify 1(a) query to display all packages and all customers.

**Non Equi Join**

Display the package number, internet speed, monthly payment and package grade for all packages whose monthly payment is between min\_price and max\_price(*Packages* and *Pack\_Grades* tables).

**Statement -1: Use of stored procedure and trigger**

**Unnamed PL/SQL code block: Use of Control structure and Exception handling is mandatory.** Write a PL/SQL block of code for the following requirements:- Schema:

1. Borrower(Roll\_no, Name, DateofIssue, NameofBook, Status)
2. Fine(Roll\_no, Date, Amount)

* Accept roll\_no & name of book from user
* Check the number of days (from date of issue); if days are between 15 to 30 then fine amounts will be Rs 5per day
* If no. of days>30, per day fine will be Rs 50 per day & for days less than 30, Rs. 5 per day.
* After submitting the book, status will change from I to R
* If condition of fine is true, then details will be stored into fine table

**Statement-2: JSON Encoding and Decoding**

Encode and Decode JSON Objects using Java/PHP/Python

**Statement- 1: Map-Reduce**

1. Create collection Bank

2. Insert following documents into it.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sr. No.** | **Cust\_Id** | **Cust\_Name** | **Bank** | **Balance** | **Location** |
|  | C001 | Amit | SBI | 3000 | Pune |
|  | C002 | Amol | AXIS | 4000 | Pune |
|  | C002 | Amit | AXIS | 5000 | Mumbai |
|  | C003 | Amit | AXIS | 5000 | Nagpur |
|  | C004 | Amay | AXIS | 6000 | Nashik |
|  | C005 | Amar | HDFC | 7000 | Pune |
|  | C006 | Vijay | HDFC | 8000 | Mumbai |

3**.** Perform aggregation using Map-Reduce to display bank wise balance

4. Display the result of new collection bankMR

5. Display the balance of HDFC bank only from bankMR collection

6. Display the balance of SBI bank only from bankMR collection

7. Display the balance of AXIS bank only from bankMR collection

**Statement- 2:**

Encode and Decode JSON Objects using Java//PHP/Python

**Statement -1: Nested Cursor**

Write a PL/SQL block of code using Nested Cursor, that will merge the data available in the newly created table N\_RollCall 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.

**Statement-2: MongoDB Aggregation**

1. Create Collection Employee and insert following documents into it
2. Insert following records into it.

|  |  |  |
| --- | --- | --- |
| custID | Amount | status |
| A123 | 500 | A |
| A123 | 250 | A |
| B212 | 200 | A |
| A123 | 300 | D |

1. Find out maximum amount for individual customer ID having status ‘A’
2. Find out minimum amount for individual customer ID having status ‘A’
3. Find out average amount for individual customer ID having status ‘A’
4. Find out total amount for individual customer ID having status ‘A’
5. Find out amount of first record for individual customer ID having status ‘A’
6. Find out amount of last record for individual customer ID having status ‘A’
7. Create array of amount for individual customer ID having status ‘A’
8. After sorting record find out total amount for individual customer ID

**Statement -1: Use of stored procedure**

Create Procedure and use cursor to check stock of items whose quantity is less than particular number and display result in temporary table 'stock check' and drop temp table after display.

**Statement-2: MongoDB Aggregation**

1. Create Collection Employee and insert following documents into it
2. Insert following records into it.

|  |  |  |
| --- | --- | --- |
| custID | Amount | Status |
| A123 | 500 | A |
| A123 | 250 | A |
| B212 | 200 | A |
| A123 | 300 | D |

1. After sorting record find out average amount for individual customer ID
2. After sorting record find out minimum amount for individual customer ID
3. After sorting record Find out maximum amount for individual customer ID
4. After sorting record find out amount of First record for individual customer ID
5. After sorting record find out amount of last record for individual customer ID
6. After sorting record create array of amount for individual customer ID
7. Without considering first record Find out total amount for individual customer ID
8. Without considering first record Find out average amount for individual customer ID

**Statement -1: PL/SQL block**

Write a PL/SQL code block that will accept an account number from the user. Check if the users balance is less than the minimum balance, only then deduct Rs. 100/- from the balance. The process is fired on the Account Master table.

(Use conditional control- If-Else If-Else-End If)

**Statement-2: MongoDB Aggregation**

1. Create Collection Employee and insert following documents into it
2. Insert following records into it.

|  |  |  |
| --- | --- | --- |
| custID | Amount | Status |
| A123 | 500 | A |
| A123 | 250 | A |
| B212 | 200 | A |
| A123 | 300 | D |

1. Without considering first record create array of amount for individual customer ID
2. Without considering first record Find out amount of First record for individual customer ID
3. Without considering first record Find out amount of last record for individual customer ID
4. Just by considering first 2 records Find out total amount for individual customer ID
5. Just by considering first 2 records Find out Average amount for individual customer ID
6. Just by considering first 2 records Find out minimum amount for individual customer ID
7. Just by considering first 2 records Find out maximum amount for individual
8. Just by considering first 2 records Find out amount of First record for individual customer ID
9. Just by considering first 2 records Find out amount of last record for individual customer ID
10. Just by considering first 2 records create array of amount for individual customer ID

**Statement -1: PL/SQL block**

Write a PL/SQL code block to calculate the area of a circle for a value of radius varying from 3 to 7, consisting of two columns Radius & Area.(Use Iterative control- the While loop).

**Statement- 2: Map-Reduce**

1. Create collection Bank

2. Insert following documents into it.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sr. No.** | **Cust\_Id** | **Cust\_Name** | **Bank** | **Balance** | **Location** |
|  | C001 | Amit | SBI | 3000 | Pune |
|  | C002 | Amol | AXIS | 4000 | Pune |
|  | C002 | Amit | AXIS | 5000 | Mumbai |
|  | C003 | Amit | AXIS | 5000 | Nagpur' |
|  | C004 | Amay | AXIS | 6000 | Nashik |
|  | C005 | Amar | HDFC | 7000 | Pune |
|  | C006 | Vijay | HDFC | 8000 | ’Mumbai |

3**.** Perform aggregation using Map-Reduce to display bank wise balance

4. Display the result of new collection bankMR

5. Display the balance of HDFC bank only from bankMR collection

6. Display the balance of SBI bank only from bankMR collection

7. Display the balance of AXIS bank only from bankMR collection

**Statement -1: PL/SQL block for bank**

Write a PL/SQL block of code to achieve the following:

If there are no transactions taken place in the last 365 days then mark the account status as inactive, and then record the account number, the opening date & the type of account in the Inactive Acct\_Master table.

**Statement- 2: SQL Queries**

* **Create Employee table, Project table and add rows shown below**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Eid** | **EName** | **Address** | **Salary** | **Commission** | **PrNo** | **Addr** |
| **1** | Amit | Pune | 35000 | 5000 | 10 | Mumbai |
| **2** | Sneha | Pune | 25000 |  | 20 | Pune |
| **3** | Savita | Nasik | 28000 | 2000 | 30 | Jalgoan |
| **4** | Pooja | Mumbai | 19000 |  | 40 | Nagpur |
| **5** | Sagar | Mumbai | 25000 | 3000 | 50 | Delhi |
| **6** | Rohit | Jaipur | 40000 |  | 60 | Kochi |
| **7** | Poonam | Patana | 45000 | 2000 | 70 | Pune |
| **8** | Arjun | Delhi | 20000 | 900 | 80 | Nasik |
| **9** | Rahul | Nagpur | 60000 | 5000 |
| **10** | Dulquer | Kochi | 30000 | 1000 |

* **Execute the following queries in MySQL:**

1. What are maximum, minimum salary, average salary and sum of all salaries?
2. Display the content of employee table according to the ascending order of salary amount.
3. Find the name of employee who lived in Nasik or Pune city.
4. Find the name of employees who does not get commission.
5. Change the city of Amit to Nashik.
6. Find the count of staff from each city
7. Find the address from where employees are belonging as well as where projects are going on. (Use union operator)
8. Find city wise maximum salary having maximum salary greater than 26000
9. Delete the employee who is having salary greater than 30,000.
10. Display all the employees from Pune alphabetically.

**Statement -1:**

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 category is first class, if marks 899 and 825 category is Higher Second Class.

Write a PL/SQL block for using procedure created with above requirement.

**Stud\_Marks(name, total\_marks)**

**Result(Roll,Name, Class)**

**Statement- 2:**

1. Consider a collection “Inventory” with the following documents:
   1. "\_id" : 1, "item" : "f1", type: "food", quantity: 500
   2. "\_id" : 2, "item" : "f2", type: "food", quantity: 100
   3. "\_id" : 3, "item" : "p1", type: "paper", quantity: 200
   4. "\_id" : 4, "item" : "p2", type: "paper", quantity: 150
   5. "\_id" : 5, "item" : "f3", type: "snacks", quantity: 300
   6. "\_id" : 6, "item" : "t1", type: "toys", quantity: 500
   7. "\_id" : 7, "item" : "a1", type: "apparel", quantity: 250
   8. "\_id" : 8, "item" : "a2", type: "apparel", quantity: 400
   9. "\_id" : 9, "item" : "t2", type: "toys", quantity: 50
   10. "\_id" : 10, "item" : "f4", type: "snacks", quantity: 75
2. Find all documents where the type field has the value snacks
3. Update a document
4. Remove a single document where type is toys and qty is 50
5. All documents where the type field has the value 'food' and the value of the price field is less than 9.9
6. All documents in the collection where the field qty has a value greater than 100 or the value of the price field is less than 9.95:
7. Find all documents in the collection where the value of the type field is 'food' and either the qty has a value greater than 100 or the value of the price field is less than 9.95:
8. Return food only with item and qty fields from the documents.
9. Sort documents on qty in descending order
10. Create Index on item field
11. Remove Index on item field

**Statement -1: Triggers**

Write a database 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.

**Statement- 2:**

1. Crete a collection having name “student” with following documents:

{

“roll\_no" : 1,

"name" : "amit",

"addr" : "Loni"

}

{

"roll\_no" : 2,

"name" : "shashir",

"addr" : {

"At" : "Loni",

"Tal" : "Rahata",

"Dist" : "Ahemadnagar"

}

}

{

“roll\_no" : 11,

"name" : "sachin",

"percent\_marks" : 60.23

"addr" : "Pune"

}

{

"name" : "Rahul",

"Address”: "Kolhar"

}

1. Insert new documents into a collection named “student”.
2. Retrieve all students’ details.
3. Display only name field from student collection.
4. Display first 5 student details.
5. Fetch the remaining documents after first 5 documents.
6. Find the details of student whose roll\_no is between 5 and 10.
7. Display the student details whose address is “Loni” or “Pune”.
8. Sort the documents of student collection in ascending order of roll\_no.
9. Remove all those documents having roll\_no less than 8 and greater than 3.
10. Remove all those documents whose address is “Loni”.
11. Add these details to a document where \_id=10 (Use save() )

Name =”Ram” Roll \_no = 60

Address =”Pune”

1. Update the address of roll\_no 1. Change the address as

At: “Sangamner” Tal:”Sangamner” Dist: “Ahemadnagar”

1. Create Index on roll\_no field
2. Remove collection “student”.

**Statement -1:**

Implement MYSQL database connectivity with PHP/ Python/Java

Implement Database navigation operations (Insert, Delete, Update, Display) using ODBC/JDBC.

**Example: Student Registration System**

**Statement -1:**

Implement MYSQL database connectivity with PHP/ Python/Java

Implement Database navigation operations (Insert, Delete, Update, Display) using ODBC/JDBC.

**Example: Library Information System**

**Statement -1:**

Implement MYSQL database connectivity with PHP/ Python/Java

Implement Database navigation operations (Insert, Delete, Update, Display) using ODBC/JDBC.

**Example: Employee Information System**

**Statement -1:**

Write a program to implement MongoDB database connectivity with PHP/ Python/Java Implement Database navigation operations (Insert, Delete, Update, Display) using ODBC/JDBC.

**Example: Library Information System**

**Statement -1:**

Write a program to implement MongoDB database connectivity with PHP/ Python/Java Implement Database navigation operations (Insert, Delete, Update, Display) using ODBC/JDBC.

**Example: Bank Information System**

**Statement -1:**

Write a program to implement MongoDB database connectivity with PHP/ Python/Java Implement Database navigation operations (Insert, Delete, Update, Display) using ODBC/JDBC.

**Example: Bus Reservation System**

**Statement -1:**

Write a program to implement MongoDB database connectivity with PHP/ Python/Java Implement Database navigation operations (Insert, Delete, Update, Display) using ODBC/JDBC.

**Example: Hospital Management System**

**Statement -1: Joins**

* **Create following tables**

Pack\_grades(grade\_id, grade\_name ,min\_price,max\_price)

Sectors(sector\_id,sector\_name)

Packages(pack\_id,speed,strt\_date,monthly\_payment,sector\_id)

Customers(cust\_id, name, start\_date,,city, phone\_num, , monthly\_discount, pack\_id)

**Inner Join**

1. Customers and internet packages (*Customers* & *Packages* tables) –
   1. Write a query to display name, package id and internet speed for all customers with similar package id.
   2. Write a query to display name, package id and internet speed for all customers whose package id equals 22 or 27. Order the query in ascending order by name.
2. Internet packages and sectors –
   1. Display the package id, internet speed, monthly payment and sector name for all packages with similar sector id. (*Packages* and *Sectors* tables).
   2. Display the customer name, package id, internet speed, monthly payment and sector name for all customers with similar package id and sector id (*Customers*, *Packages* and *Sectors* tables).
   3. Display the customer name, package id, internet speed, monthly payment and sector name for all customers in the business sector with similar package id and sector id (*Customers*, *Packages* and *Sectors* tables).

**Outer Join**

Customers and internet packages (Customers and Packages tables)

a. Modify 1(a) query to display all customers, including those without any internet package.

c. Modify 1(a) query to display all packages, including those without any customers.

d. Modify 1(a) query to display all packages and all customers.

**Non Equi Join**

Display the package number, internet speed, monthly payment and package grade for all packages whose monthly payment is between min\_price and max\_price(*Packages* and *Pack\_Grades* tables).

**Statement -1:**

Write a PL/SQL code block to calculate the area of a circle for a value of radius varying from 3 to 7, consisting of two columns Radius & Area.(Use Iterative control- the While loop).

**Statement- 2:**

1. Crete a collection having name “student” with following documents:

{

“roll\_no" : 1,

"name" : "amit",

"addr" : "Loni"

}

{

"roll\_no" : 2,

"name" : "shashir",

"addr" : {

"At" : "Loni",

"Tal" : "Rahata",

"Dist" : "Ahemadnagar"

}

}

{

“roll\_no" : 11,

"name" : "sachin",

"percent\_marks" : 60.23

"addr" : "Pune"

}

{

"name" : "Rahul",

"Address”: "Kolhar"

}

1. Insert new documents into a collection named “student”.
2. Retrieve all students’ details.
3. Display only name field from student collection.
4. Display first 5 student details.
5. Fetch the remaining documents after first 5 documents.
6. Find the details of student whose roll\_no is between 5 and 10.
7. Display the student details whose address is “Loni” or “Pune”
8. Sort the documents of student collection in ascending order of roll\_no.
9. Remove all those documents having roll\_no less than 8 and greater than 3.
10. Remove all those documents whose address is “Loni”.
11. Add these details to a document where \_id=10 (Use save() )

Name =”Ram” Roll \_no = 60

1. Update the address of roll\_no 1. Change the address as

At: “Sangamner” Tal:”Sangamner” Dist: “Ahemadnagar”

Address =”Pune”

1. Create Index on roll\_no field
2. Remove collection “student”.

**Statement -1:**

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 category is first class, if marks 899 and 825 category is Higher Second Class.

Write a PL/SQL block for using procedure created with above requirement.

**Statement- 2:**

1. Consider a collection “Inventory” with the following documents:
   1. "\_id" : 1, "item" : "f1", type: "food", quantity: 500
   2. "\_id" : 2, "item" : "f2", type: "food", quantity: 100
   3. "\_id" : 3, "item" : "p1", type: "paper", quantity: 200
   4. "\_id" : 4, "item" : "p2", type: "paper", quantity: 150
   5. "\_id" : 5, "item" : "f3", type: "snacks", quantity: 300
   6. "\_id" : 6, "item" : "t1", type: "toys", quantity: 500
   7. "\_id" : 7, "item" : "a1", type: "apparel", quantity: 250
   8. "\_id" : 8, "item" : "a2", type: "apparel", quantity: 400
   9. "\_id" : 9, "item" : "t2", type: "toys", quantity: 50
   10. "\_id" : 10, "item" : "f4", type: "snacks", quantity: 75
2. Find all documents where the type field has the value snacks
3. Update a document
4. Remove a single document where type is toys and qty is 50
5. All documents where the type field has the value 'food' and the value of the price field is less than 9.9
6. All documents in the collection where the field qty has a value greater than 100 or the value of the price field is less than 9.95:
7. Find all documents in the collection where the value of the type field is 'food' and either the qty has a value greater than 100 or the value of the price field is less than 9.95:
8. Return food only with item and qty fields from the documents.
9. Sort documents on qty in descending order
10. Create Index on item field
11. Remove Index on item field

**Statement -1: View**

* **Create table Employees and Departments as shown below:**

(Use emp\_id as primary key for employees table and dept\_id as a primary key for departments table)

|  |  |  |  |
| --- | --- | --- | --- |
| emp\_id | emp\_name | salary | dept\_id |
| 1 | Ethan Hunt | 5000 | 4 |
| 2 | Tony Montana | 6500 | 1 |
| 3 | Sarah Connor | 8000 | 5 |
| 4 | Rick Deckard | 7200 | 3 |
| 5 | Martin Blank | 5600 | NULL |

|  |  |
| --- | --- |
| dept\_id | Dept\_name |
| 1 | Administration |
| 2 | Customer Service |
| 3 | Finance |
| 4 | Human Resources |
| 5 | Sales |

* **Execute the following queries in MySQL:**

1. Retrieve the id and name of the employees along with their department name (Use Left Join)

SELECT t1.emp\_id, t1.emp\_name, t2.dept\_name

FROM employees AS t1 LEFT JOIN departments AS t2

ON t1.dept\_id = t2.dept\_id;

1. Create View for the above query.
2. Retrieve records from the above created view.
3. Replace the above view to show emp\_id, .emp\_name, dept\_name salary also.
4. Create another view using salary, emp\_id and dept\_id.
5. Insert 3 more records in above created view.
6. Update the above view to set salary =6000 for the employee having emp\_id 1.
7. Retrieve the record from view where dept\_id of employees is NULL.
8. Delete those records from view where employee salary is 8000.
9. Drop the above created view.

**Statement -2: JSON**

Create simple objects and array objects using JSON