Experiment No : 1

raisoni EDUCATION

* Aim :

Draw E-R diagram and convert entities and relationship to relation table for given scenario.
Two assignments shall be carried out i.e consider two different scenarios (eg bank, college, Employee, Hotel etc)

* Theory :

ER diagrams are the visual tools that helpful to represent the ER model. Peter Chen proposed ER-Diagram in 1971 to create a uniform convention that can be used for relational databases and networks. He aimed to use an ER model as a conceptual modeling approach.
An entity-relationship diagram (ERD) shows the relationships of entity sets stored in a database. An entity in this context is an object, a component of data. An entity set is a collection of similar entities can have attributes that define their properties. By defining the entities and their attributes and showing the relationship between them, an ER diagram illustrates the logical structure of databases.

* Facts about ER diagram are as follows :

- ER model allows you to draw Database Design
- It is an easy to use graphical tool for modeling data
- Widely used in Database Design
- It is GUI representation of the logical structure of Databas

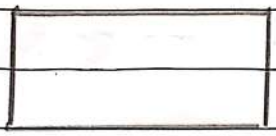
raisoni
EDUCATION

* ER diagrams Symbols & Notations :

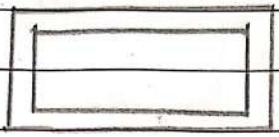
Entity Relationship Diagram Symbols & Notations are mainly contains three basic symbols which are rectangle, oval and diamond to represent relationships between the elements, entities and attributes. There are some sub-elements which are based on main elements in ERD Diagram. ER Diagram is a visual representation of data that describes how data is related to each other using different ERD Symbols and Notations.

* Following are the main components and its symbols in ER Diagram

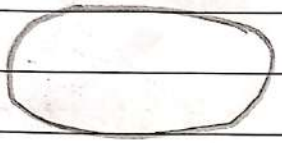
- Rectangles :- This Entity Relationship Diagram symbol represents entity types
- Ellipses :- Symbol represent attributes
- Diamonds :- This symbol represents relationship types.
- Lines :- It links attributes to entity types and entity types with other relationship types.
- Primary key :- attributes are underlined
- Double Ellipses :- Represent multi-valued attributes.



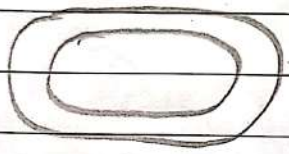
Entity or Strong Entity



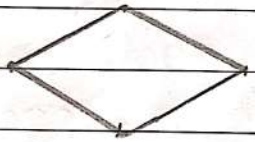
Weak Entity



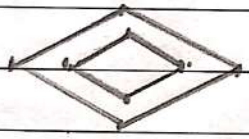
Attribute



Multivalued Attribute



Relationship



Weak Relationship

* Example :- ER Diagram

A simple model of a banking system :-

1.) Entities :

Customer :- A person who has a bank account. The attributes of the customer are :

- First Name
- Last Name
- Gender
- ID Number
- Birth Date

- Bank Account :-

Represents the account owned by a customer. The attribute of bank account are :

- Balance
- Account Type

- Bank :- The entity representing the bank where accounts are held. Its attribute is :

- Name (the bank's name)

- Transactions :- Represents the action taken on the bank account, like deposits and withdrawals. Its attributes are :-

- Time Stamp
- Deposit
- Withdrawal
- Currency
- Amount

2.) Relationships :

- A customer "owns" a Bank Amount.
- A Bank Account is "attached to" Transactions.
- A Bank is associated with a Bank Account

* Conclusion :-

In this experiment, we have studied history, facts about ER Diagram, necessity of ER diagram along with Entity Relationship diagram notations and their use to draw the design of a database system.

Experiment No : 2

raisoni EDUCATION

* Aim :-

a) Creating a database
b) Creating Tables (With and Without Constraints)
c) Inserting Record in table

* Theory

• Constraints :-

Integrity Constraints are a mechanism to prevent invalid data entry into the table to maintain data consistency. The whole purpose of constraints is to maintain the data integrity during the various transactions like update / delete / insert on a table.

• Types of constraints :

- NOT NULL
- UNIQUE
- DEFAULT
- CHECK
- Key Constraints
  - PRIMARY KEY
  - FOREIGN KEY

→ NOT NULL :-

NOT NULL contraint makes sure that a column does not hold NULL value. When we don't provide value for a

particular column while inserting a record into a table, it takes NULL value by default. By specifying the NULL constraint, we can be sure that particular column (s) cannot have NULL values.

→ UNIQUE :-

UNIQUE Constraint enforces a column or set of columns to have unique values. If a column has a unique constraint, it means that a particular column cannot have duplicate values in a table.

→ DEFAULT :-

The DEFAULT constraint provides a default value to a column when there is no value provided inserting a record into table.

→ CHECK :-

This constraint is used for specifying a range of values for particular column of a table. When this constraint is being set of column, it ensures that the specified column must have the value falling in the specified range.

• Key Constraint :-

→ PRIMARY KEY :-

The primary key uniquely identifies each record in a table. It must have unique values and cannot contain nulls. In the below example the ROLL-NO field is marked as the primary key, which means the ROLL-NO field cannot have duplicate and null values.

FOREIGN KEY

Foreign key are the column of a table that points to the primary key of another table. They act as a cross-reference between tables.

Conclusion :-

In this experiment, we have studied the concept of constraint and executed all constraints such as NULL, NOT NULL, PRIMARY KEY, UNIQUE, CHECK, DEFAULT, and REFERENCES. Also created tables with and without constraint and inserted records into it.

Experiment No : 3

raisoni EDUCATION

* Aim :

To Perform the following :- Viewing all databases, Viewing all the tables in the Database, Updating / Deleting the Records in table.

* Theory :-

- View :-

→ A database view is a virtual table or logical table which is defined as SQL SELECT query with joins. Because a database view is similar to a database table management systems, including MySQL, allows you to update data in the underlying tables through the database view with some prerequisites.

→ A database view is dynamic because it is not related to the physical schema. The database system stores database views as SQL SELECT statement with joins. When the data of the tables changes, the view reflects that changes as well

- Advantages of database view :-

a) A database views allows you to simplify complex queries : a database view is defined by an SQL statement that is associates with underlying tables. You can use a database view to hide the complexity of underlying tables to the end-users and external applications.

b) A database view helps limits data access to specific users :- You may not want a subset of sentitive data that can be queryable by all users.

c) A database view provides an extra security layer. Security is a vital part of any relational database management system. Database views provide extra security for a database management system.

d) A database view enables computed columns. A database table should not have calculated columns however a database view should.

- Disadvantages of database view :-

Besides the advantages above, there are several disadvantages of using database views

a) Performance :- Querying data from a database view can be slow especially if the view is created based on others views.

b) Tables Dependency :- you create a view based on underlying tables of the database. Whenever you change the structure of those tables that view associates with, you have to change the view as well.

* DDL COMMAND ON VIEW :-

1. CREATE :- Syntax :- CREATE VIEW view_name AS SELECT colum_nam FROM table_name WHERE condition ;

ALTER :- Once view is defined, you can modify it by using the ALTER VIEW statement.

Syntax :- ALTER VIEW view_name AS SELECT column_name(s) FROM table_name WHERE condition ;

DROP :-

Syntax :- DROP VIEW [IF EXISTS] [database_name].[view-nam

Example :- DROP VIEW EMP.emp_view ;

TRUNCATE :-

We can't do truncate on view

DML Commands :-

DML commands are most frequently used SQL command and are used to query and manipulate the existing database. Some of the commands are Insert, Select, Update and Delete.

Insert Command is used to add one or more rows to the table.

Select Command :- Select Command is used to retrieve information from the table.

Update Command :- Update Command is used to alter the column values in a table. A single column may be updated or more than one column could be updated.

Delete command :- Delete Command after inserting a row in a table we can also delete them if required The delete command consists of a from clause followed by an optional where clause

* Conclusion :-

In this experiment, we have studied the concept of Data Manipulation Language and implemented Insert, Select, Update Delete. Also implemented queries to View all databases, all Tables in a Database.

Experiment No :- 4

raisoni EDUCATION

* Aim :-

To perform the following SQL query on database :
Altering a Table , Dropping / Truncating / Renaming Tables , Backing up / Restoring a Database

* Theory :-

Structured Query Language (SQL) as we all know is the database language by the use of which we can perform certain operation on the existing database and also we can use this language to create a database . SQL uses certain commands like Create , Drop , Insert , etc .

Data Definition Language actually consist of the SQL commands that can be used to define the database schema . It simply deals with descriptions of the database schema and is used to create and modify the structure of database object in the database . DDL is a set of SQL command used to create modify , and delete database structure but not data . These commands are normally not used by a general user , who should be accessing the database via an application .

* List of DDL commands

- CREATE :- This command is used to create the database or its objects (like table , index , function , views , store procedure , and triggers)
- DROP :- This command is used to delete objects from the database .

ALTER : This is used to alter the structure of the database

TRUNCATE : This is used to remove all records from a table including all spaces allocated for the records are removed.

RENAME : This is used to rename an object exiting in the database.

The SQL CREATE TABLE Statement :

The CREATE TABLE statement is used to create a new table in a database.

Syntax :

```
CREATE TABLE table_name (

column 1 datatype,
column 2 datatype,
column 3 datatype,
.....
);
```

The SQL DROP Table Statement

The Drop TABLE statement is used to drop an existing table in a database.

Syntax :

```
DROP TABLE table_name ;
```

* SQL ALTER TABLE Statements

The ALTER TABLE statement is used to add, delete, or modify columns in an existing table.

The ALTER TABLE statement is also used to add and drop various constraints on an existing table.

To add a column in a table, use the following syntax

Syntax :-
```
ALTER TABLE table_name
ADD column_name datatype ;
```

* ALTER TABLE - DROP COLUMN

To delete a column in a table, use the following syntax

Syntax :-
```
ALTER TABLE table_name
DROP COLUMN column_name ;
```

- DCL (Data Control Language)

DCL includes commands as GRANT and REVOKE which mainly deal with the rights, permissions, and other controls of the database system.

- List of DCL commands :-

GRANT :- This command gives users access privileges to the database.

REVOKE :- This command withdraws the user's access privileges given by using the GRANT command.

List of TCL commands

COMMIT :- Commits a Transaction

ROLLBACK :- Rollbacks a transaction in case of any error occurs.

SAVEPOINT :- Set a savepoint within a transaction

The SQL BACKUP DATABASE Statement

The BACKUP DATABASE statement is used in SQL Server to create a full back up of an existing SQL database.

Syntax :- BACKUP DATABASE database name
TO DISK = 'filepath',

Conclusion :-

In this experiment we studied Data Definition Languge all structure related queries such as create, alter with add and modify option, rename, drop

Experiment No :- 5

raisoni EDUCATION

* AIm :-

For given set of relation schemes, create tables and perform the following Simple Queries. Simple Queries with Aggregate functions, Queries with Aggregate functions (group by having clause), Queries involving - Data Functions, string Function Math Functions.

* Theory :-

Aggregate Functions :-

Aggregate functions perform a calculation on a set of values and returns a single value. There are different types of aggregate functions such as min, max, sum, avg count, etc.

Why use aggregate functions :-

From a business perspective, different organization levels have different information requirements. Top levels managers are usually interested in knowing whole figures and not necessary the individual details.

Aggregate functions allow us to easily produce summarized data from our database.

* Aggregate Functions are all about :-

- Performing calculations on multiple rows
- Of a single column of a table
- And returning a single value.

- Simple Queries with Aggregate Functions :

Aggregate functions perform calculations on a set of values and return a single value. Common aggregate functions inclu

- SUM() : Calculate the total sum of a numeric column.
- AVG() : Returns the average value
- COUNT() : Counts the number of rows
- MAX() : Returns the maximum value
- MIN() : Returns the minimum value

- Queries with Aggregate Functions (GROUP BY and HAVING clause)

The GROUP BY clause groups rows that have the same values i specified columns into summary rows. It is often used with aggregate functions.

The HAVING clause is similar to the WHERE clause, but it is used to filter groups after the grouping is done

- Queries Involving Date Functions :

Date Functions allow manipulation and extraction of data related to dates. Common date function include :

- NOW() : Return the current date and time
- DATEDIFF() : Returns the difference between two dates.
- DATEADD() : Adds a specified time interval to a date.

- Queries Involving String Functions :

String Function are used to manipulate text strings. Common string function include :

- CONCAT() : Concatenates two or more strings
- SUBSTRING() : Extracts a substring from a string.
- LENGTH() : Returns the length of a string.

- Queries Involving Math Functions :

Math Functions allow for numerical calculations. Common math functions include

- ROUND() : Rounds a number to a specified number of decimal places.

- ABS() : Returns the absolute value of a number
- POWER() : Raises a number to the power of another number.

* Conclusion :

In this experiment, we have studied aggregate functions and Transection Control Language and implemented Count, Sum, Max Min, Avg, Commit, Rollback, and Savepoint commands. Simple Queries with Aggregate Functions, Quries with Aggregate functions (group by and having clause, Queries involving - Date Functions String Functions, Math Function.

Experiment No : 6

raisoni EDUCATION

* Aim :

To perform SQL query that demonstrate Join Queries - Inner Join . Outer Join , Left join , Right Join .

* Theory :

SQL JOIN : A JOIN clause is used to combine rows from two or more tables , based on a related column between them . Following are different types of join

1. Equi - join / Inner join
2. Non - equi - join
3. Self - join
4. Outer join

• Equi - join :

A join that is based on equalities is called equal - join . '=' operator is used in equi - join comparsion . It retrieves rows from tables having a common column . It also called simple join .

• Non - equi - join :

A join that specifies the relationship between column belonging to different tables by making use of the relational operators (< , > , <= , >= , !=) other than the '=' operator is called as non - equi - join .

Self - join

Joining a table to itself is known as self - join i.e. it joins one row in a table to another. It can compare each row of the table to itself and with other rows of the same table

Outer join :

An outer join returns all the rows returned by simple join or equi join as well as those rows from one table that do not match any two row from the other table.

The symbol (+) represents outer join.

Example :

Consider three tables

1) Employees Table : This table stores information about employees including their name, department, salary and manger
2) Department Table : This table contain details about different departments in the organization.
3) Salaries Table : This table defines salary ranges for employees.

1) Equi - join | Inner Join

```
SELECT Employees.Name, Departments.DeptName
FROM Employees
INNER JOIN Departments ON Employees.DeptID = Departments.DeptID
```

This query returns the employee names and their corresponding

department names where the DeptID matches in both tables

2) Non - equi join :-

```
SELECT Employees.Name, Salaries.SalaryRange
FROM Employees
JOIN Salaries ON Employees.Salary < Salaries.MaxSalary
```

This query select employees whose salary is less than the maximum salary range from the Salaries table.

3) Self - join :-

```
SELECT A.Name AS Employee, B.Name AS Manager
FROM Employees A, Employees B
WHERE A.ManagerID = B.EmployeeID;
```

This query shows each employees and their manager by joining the Employees table with itself.

4) Outer - join :-

```
SELECT Employees.Name, Departments.DeptName
FROM Employees
LEFT JOIN Departments ON Employees.DeptID = Departments.DeptI
```

This query returns all employees and department names, but if an employee has no department, the DeptName will be NULL.

* Conclusion :- In this experiment, we have studied SQL queries for suitable database application using SQL

Experiment No :- 7

raisoni EDUCATION

* Aim :-

To perform SQL query that demonstrate following : Search condition, Summary queries, Sub-queries, Subqueries With IN clause, With EXISTS clause.

* Theory :-

In SQL a Subquery can be simply defined as a query with another query. In other words we can say that a Subquery is a query that is embedded in WHERE clause of another SQL query. Important rules for Subqueries :-

- You can place the Subquery in a number of SQL clauses : WHERE clause, HAVING clause, FROM clause. Subqueries can be used with SELECT, UPDATE, INSERT, DELETE statement along with expression operator. It could be equality operation or comparison operator such as =, >, =, <= and Like operator.

- A subquery is a query within another query. The outer query is called as main query and inner query is called as sub-query.

- The subquery generally executes first when the subquery doesn't have any co-relation with the main query, when there is a co-relation the parser takes the decision on the fly on which query to execute on precedence and uses the output of the subquery accordingly.

- Subquery must be enclosed in parentheses.
- Subqueries are on the right side of the comparsion operator
- ORDER BY command cannot be used in a Subquery. GROUPBY command can be used to perform same function as ORDER BY command.
- Use single-row operators with single row Subqueries. Use multiple-row operators with multiple-row Subqueries.

Syntax :- There is not any general syntax for Subqueries. However, Subqueries are seen to be used most frequently with SELECT statement as shown below.

```
SELECT column_name
FROM table_name
WHERE column_name expression operator
```

- Sample Table :- Database

NAME	ROLL-NO	LOCATION	PHONE-NUMBER
Ram	101	Chennai	9988775566
Raj	102	Coimbatore	8877665544
Sasi	103	Madurai	7766553344
Ravi	104	Salem	8989898989
Sumathi	105	Kanchipuram	89985668 68

- Student :

Name	ROLL-NO	SECTION
Ravi	104	A
Sumathi	105	B
Raj	102	A

- Sample Queries :

- To display NAME , LOCATION , PHONE-NUMBER of the students from DATABASE table whose section A

```
Select NAME , LOCATION , PHONE-NUMBER from DATABASE

WHERE ROLL-NO IN

( SELECT ROLL-NO from STUDENT where SECTION = 'A' );
```

- Explanation : First subquery executes " SELECT ROLL-NO from STUDENT where SECTION = 'A' Return ROLL-NO from STUDENT table whose SECTION is 'A' . Then outer-query executes it and return the NAME , LOCATION , PHONE-NUMBER from the DATABASE table of the student whose ROLL-NO is returned from inner subquery .

- Output :

NAME	ROLL-NO	LOCATION	PHONE_NUMBER
Ravi	104	Salem	8989898989
Raj	102	Coimbatore	8877665544

* Conclusion :-

In this experiment concept of subquery single row subquery multiple row subquery and multiple column subqueries studied and implemented.

Experiment No : 8

raisoni EDUCATION

* Aim :

To perform SQL query for extracting data from more than one table using SQL concept.

* Theory :

- To extract data from more than one table in SQL, we can use JOINs, which combines rows from two or more tables based on a related column. The most common types are :

1) INNER JOIN :

Returns records that having (have) matching values in both tables.

2) LEFT JOIN (LOUTER JOIN) :

Returns all records from the left table and the matched records from the right table. If there is no match, NULL values are returned for columns from the right table.

3) RIGHT JOIN (RIGHT OUTER JOIN) :

Returns all records from the right table and the matched records from left table. If there is no match, NULL values are returned for columns from the left table.

raisoni EDUCATION

4) FULL JOIN (FULL OUTER JOIN)

Return all records where (is) there is a match in either left or right table. If there is no match, NULL values are returned.

* Example :- Consider two tables

- Students :-

student_id	name
1	John
2	Jay
3	Ajay

- Courses :-

course_id	student_id	course_name
101	1	Math
102	2	Science
103	3	History

To extract data showing each student and their enrolled course, we would use.

```
SELECT Student.name, Courses.course_name
FROM Students
INNER JOIN Courses ON Students.student_id = Courses.student_id;
```

This query joins the Students and Courses tables based on the student_id column and retrieves the student names along with their corresponding course names.

- We also use subqueries to extract data from multiple tables. A subquery is a query within another query and it can be used as an alternative to joins in some situations. In context of extracting data from multiple tables, subqueries can be used in the SELECT, WHERE, or FROM clauses to filter or manipulate data from one table based on the result of another query.

Example using Subquery :-

Using the same Student and Courses tables :-

```
SELECT name,
        (SELECT course_name
         FROM Courses
         WHERE Courses.student_id = Students.student_id ) ...
         AS course_name
FROM Students;
```

Here, the subquery inside the SELECT statement retrieves the course_name for each student by matching the student_id from the Courses table with the Students table. This produces same result as using an INNER JOIN in this case.
But it might less efficient compared to JOINs when workin with large datasets

* Conclusion :-

In this experiment we implemented SQL query for extracting data from more than one table using SQL concept.

Experiment No :- 9

raisoni EDUCATION

* Aim :-

To perform SQL query to understand the concepts. Transaction ROLL BACK, COMMIT, and CHECK POINTS.

* Theory :-

Transaction Control Language (TCL) is a critical component of SQL used to manage transaction and ensure data integrity in relational databases. By using TCL commands, we can control how changes to the database are commited or reverted, maintaining consistency arcross multiple operations.

* TCL Commands :-

TCL includes the following commands :

1) COMMIT

- The COMMIT command is used to save all the transactions to the database that have been performed during the current transaction.
- Once a transaction is committed, it becomes permanent and connot be undone.
- This command is typically used at the end of the end of a series of SQL statements to ensure that all change made during the transaction are saved.

Syntax :- COMMIT ;

Example :-

Consider the following Table Student :

Name	Marks
John	79
Jolly	65
Shuzan	70

```
UPDATE STUDENT
SET NAME = 'Sherlock'
WHERE NAME = 'Jolly'
COMMIT ;
```

Now after commit :-

Name	Marks
John	79
Sherlock	65
Shuzan	70

2) ROLLBACK

- The ROLLBACK command is used to undo all the transaction that have been performed during the during the current transaction but yet been committed.
- This command is useful for reverting the database to its

pervious state in case an error occurs or if the changes made are not desired.

Syntax :

```
ROLLBACK;
```

Example :-

Now if the ROLLBACK is performed on the above table :

```
ROLLBACK;
```

After rollback :-

Name	Marks
John	79
Jolly	65
Shuzan	70

3) SAVEPOINT

- The SAVEPOINT command is used to set a point within a transaction to which we can later roll back.
- This command allows for partial rollbacks within a transaction providing more control over which part of a transaction to undo

Syntax :- SAVEPOINT savepoint_name ;

Example :- if on the above table savepoint is performed

```
INSERT into STUDENT
VALUES ('Jack', 95);
COMMIT ;
UPDATE NAME
SET NAME = 'Rossic'
WHERE marks = 70;
SAVEPOINT A ;

INSERT INTO STUDENT
VALUES ('Zack', 76)
SAVEPOINT B ;

INSERT INTO STUDENT
VALUES ('Bruno', 85);
SAVEPOINT C ;
```

Rollback to Savepoint B

```
ROLLBACK to B ;
```

Name	Marks
John	79
Jolly	65
Rossic	70
Jack	95
Zack	76

Rollback to Savepoint A

```
ROLLBACK to A ;
```

Name	Marks
John	79
Jolly	65
Rossic	70
Jack	95

* Conclusion :- In this experiment performed Transaction control Language such commit , rollback , savepoint.

Experiment No : 10

raisoni EDUCATION

* Aim :

Design and Develop Mongo DB Qurics using CRUD operations.

* Theory :

Mongo DB CRUD Operations

- Create Operations
- Read Operations
- Update Operations
- Delete Operations

- Create Operations

Create or insert operation and add new documents to a collectio
does not currently exist, insert operations will create the collectio

Mongo DB, insert operation target a single collection. All write operations in Mongo DB are atomic on level of single document.

- Mongo DB provides the following methods to insert documents into a collection

- db.collection.insertOne
- db.collection.insertMany()

- Read Operations :

Read operation retrieves document from a collection, i.e queries a collection for documents MongoDB provides the following methods to read documents from a collection

- db.collection.find()

You can specify query filters or criteria that identify the documents to return.

- Update Operations

Update operation modify existing document in a collection. MongoDB provide the following methods to update documents of a collection.

- db.collection.updateOne()
- db.collection.updateMony()
- db.collection.replaceOne()

In MongoDB, update operations target a single collection. All write operations in MongoDB are atomic on the level of single document.

You can specify criteria, or filters that identify the documents to update. These filters use the same syntax as reading operations

For Examples, see Update Documents.

- Delete Operations :-

Delete operations remove documents from a collection. MongoDB provides the following methods to delete document of a collection :-

- db.collection.deleteOne()
- db.collection.deliteMany()

In MongoDB, delete operation target a single collection. All write operations in MongoDB are atomic on the level of a single document.

You can specify criteria, or filters, that identify the documents to remove. These filters as the use same syntax as reading operations.

- The Use Command :- MongoDB uses DATABASE_NAME is used to create a database. The command will create a new database; if it doesn't exist otherwise it will return the existing database.

Syntax :- Use DATABASE_NAME ;

- MongoDB Drop Database :-

The Drop Database : MongoDB db.dropDatabase() command is used to drop an existing database.

Syntax :

db.dropDatabase()

3) Basic Operations with the Shell

We can use the four basic operations, create, read, update and delete to manipulate and view data in the shell.

Create :- The insert function adds a document to a collection.

Update :- If we would like to modify our post, we can use use an update. The update takes two parameters : the first is the criteria to find which document to update, and the second is new document.

Read :- Find and FindOne can be used to query a collection. If we just want to see one document from a collection we can use findOne.

Delete :- Remove permanently deletes document from the database. Called with no parameters, it removes all documents from a collection.

* Conclusion :-

In this experiment, we have studied NoSQL and implemented CRUD operations for MongoDB.