(For Internal Circulation and reference only)

Visvesvaraya Technological University

"Jnana Sangama", Belagavi-590 018

****

Fourth Semester B.E.

[As per Choice Based Credit System (CBCS) scheme]

"Database Management System Laboratory (BCS403)" Manual

Name	
USN	
Section	
Lab Batch	
Day / Time	

****

Kalpataru Institute of Technology, Tiptur - 572 201

Department of Computer Science and Engineering

AY: 2024-2025

Kalpataru Institute of Technology, Tiptur-572 201
(Affiliated to Visvesvaraya Technological University, Belagavi & Recognised by AICTE, New Delhi)
Department of Computer Science & Engineering
NBA Accredited from 2022-2025

LAB COMPLETION DETAILS FOR EVEN SEMESTER AY: 2024-2025

Sl.NO	Experiments	DATE AND TIME	SIGNATURE
1	Create a table called Employee & execute the following. **Employee (EMPNO, ENAME, JOB, MANAGER_NO, SAL, COMMISSION)** Create a user and grant all permissions to the user. Insert the any three records in the employee table contains attributes EMPNO, ENAME JOB, MANAGER_NO, SAL, COMMISSION and use rollback. Check the result. Add primary key constraint and not null constraint to the employee table. Insert null values to the employee table and verify the result.		
2	Create a table called Employee that contain attributes EMPNO, ENAME, JOB, MGR, SAL & execute the following. Add a column commission with domain to the Employee table. Insert any five records into the table. Update the column details of job Rename the column of Employ table using alter command. Delete the employee whose Empno is 105.		
3	Queries using aggregate functions (COUNT, AVG, MIN, MAX, SUM), Group by, Order by.		

	Employee (E_id, E_name, Age, Salary) Create Employee table containing all Records E_id, E_name, Age, Salary. Count number of employee names from employee table Find the Maximum age from employee table. Find the Minimum age from employee table. Find salaries of employee in Ascending Order. Find grouped salaries of employees.		
4	Create a row level trigger for the customers table that would fire for INSERT or UPDATE orDELETE operations performed on the CUSTOMERS table. This trigger will display the salary difference between the old & new Salary. **CUSTOMERS(ID,NAME,AGE,ADDRESS,SALARY)**		
5	Create cursor for Employee table & extract the values from the table. Declare the variables, Open the cursor & extract the values from the cursor. Close the cursor. **Employee(E_id, E_name, Age, Salary)**		
6	Write a PL/SQL block of code using parameterized 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.		
7	Install an Open Source NoSQL Data base MangoDB & perform basic CRUD (Create, Read, Update & Delete) operations. Execute MangoDB basic Queries using CRUD operations.		

Kalpataru Institute of Technology, Tiptur-572 201
(Affiliated to Visvesvaraya Technological University, Belagavi & Recognised by AICTE, New Delhi)
Department of Computer Science & Engineering
NBA Accredited from 2022-2025

SYLLABUS FOR EVEN SEMESTER AY: 2024-2025

SEMESTER:	**4TH**	**SECTION:**	**A**	**BATCH:**	**2023**
COURSE:	**DATABASE MANAGEMENT SYSTEMS**		**COURSE CODE**		**IPCC BCS403**
CIE	**50**	**SEE Marks:**	**50**	**Exam Hours**	**03**
Number of Contact Hours/Week (L:T:P:S)	**3:0:2:0**	**Total Number of Contact Hours**	**50**	**CREDITS**	**4**
FACULTY NAME:	**SHASHIDHARA MS**				

PRACTICAL COMPONENT OF IPCC *(May cover all / major modules)*

Sl.NO	Experiments
1	Create a table called Employee & execute the following. **Employee (EMPNO, ENAME, JOB, MANAGER_NO, SAL, COMMISSION)** Create a user and grant all permissions to the user. Insert the any three records in the employee table contains attributes EMPNO, ENAME JOB, MANAGER_NO, SAL, COMMISSION and use rollback. Check the result. Add primary key constraint and not null constraint to the employee table. Insert null values to the employee table and verify the result.
2	Create a table called Employee that contain attributes EMPNO, ENAME, JOB, MGR, SAL &execute the following. Add a column commission with domain to the Employee table. Insert any five records into the table. Update the column details of job Rename the column of Employ table using alter command. Delete the employee whose Empno is 105.
3	Queries using aggregate functions (COUNT, AVG, MIN, MAX, SUM), Group by, Order by. **Employee (E_id, E_name, Age, Salary)** Create Employee table containing all Records E_id, E_name, Age, Salary. Count number of employee names from employee table Find the Maximum age from employee table.

	Find the Minimum age from employee table. Find salaries of employee in Ascending Order. Find grouped salaries of employees.
4	Create a row level trigger for the customers table that would fire for INSERT or UPDATE orDELETE operations performed on the CUSTOMERS table. This trigger will display the salary difference between the old & new Salary. **CUSTOMERS(ID,NAME,AGE,ADDRESS,SALARY)**
5	Create cursor for Employee table & extract the values from the table. Declare the variables, Open the cursor & extract the values from the cursor. Close the cursor. **Employee(E_id, E_name, Age, Salary)**
6	Write a PL/SQL block of code using parameterized Cursor, that will merge the data availablein 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.
7	Install an Open Source NoSQL Data base MangoDB & perform basic CRUD (Create, Read, Update & Delete) operations. Execute MangoDB basic Queries using CRUD operations.

Pre Commands for setup Oracle database

1. Create a user command

Syntax: create user username identified by password;

Example: create user example identified by example;

2. Grant connect to user

Syntax: grant connect to userName identified by password;

Example: grant connect to example identified by example;

3. Grant all privileges to user

Syntax: grant all privileges to userName identified by password;

Example: grant all privileges to example identified by example;

EXPERIMENT 1:

AIM:

Create a table called Employee & execute the following.

Employee (EMPNO, ENAME, JOB, MANAGER_NO, SAL, COMMISSION)

1. Create a user and grant all permissions to the user.
2. Insert the any three records in the employee table contains attributes EMPNO, ENAME JOB, MANAGER_NO, SAL, COMMISSION and use rollback. Check the result.
3. Add primary key constraint and not null constraint to the employee table.
4. Insert null values to the employee table and verify the result.

DESIGN:

TABLE NAME: employee

Fields/Column name	Domain/Data Format (SQL keyword)	Size	Syntax
EMPNO	Number (number)	32-bit integer	empno number(10)
ENAME	Character (VARCHAR2)	10 Characters	ename varchar2(10)
JOB	Character (VARCHAR2)	10 Characters	job varchar2(10)
MANAGER_NO	Number (number)	32-bit integer	manager_no number(10)
SAL	Number (number)	32-bit integer	sal number(10)
COMMISSION	Number (number)	32-bit integer	commission number(10)

PROBLEMS AND SOLUTION:

P1. Create a table called Employee & execute the following.

Employee (EMPNO, ENAME, JOB, MANAGER_NO, SAL, COMMISSION)

SOLUTION:

SQL> create table employee (empno number(10), ename varchar2(10), job varchar2(10), manager_no number(10), sal number(10), commission number(10));

Results Explain Describe Saved SQL History

```
Table created.
```

0.03 seconds

SQL> desc employee;

Object Type TABLE Object EMPLOYEE

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
EMPLOYEE	EMPNO	Number	-	10	0	-	✓	-	-
	ENAME	Varchar2	10	-	-	-	✓	-	-
	JOB	Varchar2	10	-	-	-	✓	-	-
	MANAGER_NO	Number	-	10	0	-	✓	-	-
	SAL	Number	-	10	0	-	✓	-	-
	COMMISSION	Number	-	10	0	-	✓	-	-

1 - 6

Practice Query:

Drop the table created

SQL> `DROP TABLE employee;`

Results Explain Describe Saved SQL History

```
Table dropped.
```

0.34 seconds

P2. Insert the any three records in the employee table contains attributes EMPNO, ENAME JOB, MANAGER_NO, SAL, COMMISSION and use rollback. Check the result.

SOLUTION:

SYNTAX:

INSERT INTO table (column1, column2, ... column_n) VALUES (expression1, expression2, ... expression_n);

SQL>

1. insert into employee (empno, ename, job, manager_no, sal, commission)
values(1,'Abhi','Manager',1,20000,1000);

Results Explain Describe Saved SQL History

```
1 row(s) inserted.

0.00 seconds
```

2. insert into employee (empno, ename, job, manager_no, sal, commission)
values(2,'Rohan','Analyst',1,10000,100);

3. insert into employee (empno, ename, job, manager_no, sal, commission)
values(3,'David','Clerk',1,9000,50);

Insert command with rollback option

```
BEGIN
    insert into employee (empno, ename, job, manager_no, sal, commission)
    values(4,'Karan','Manager',1,20000,1000);
    ROLLBACK;
END;
```

Practice Query:

Check the result of insertion of rows into the employee table.

SQL> select * from employee;

Results Explain Describe Saved SQL History

EMPNO	ENAME	JOB	MANAGER_NO	SAL	COMMISSION
3	David	Clerk	1	9000	50
1	Abhi	Manager	1	20000	1000
2	Rohan	Analyst	1	10000	100

3 rows returned in 0.00 seconds CSV Export

P3. Add primary key constraint and not null constraint to the employee table.

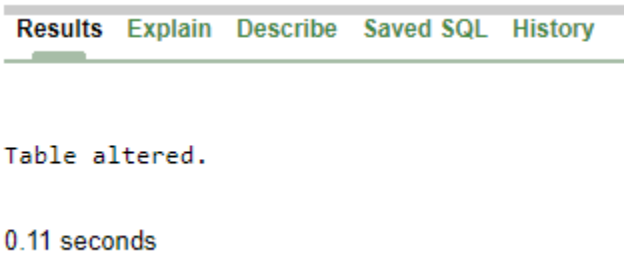
SOLUTION:

To add NOT NULL constraint to existing column to the employee table.

SYNTAX: ALTER TABLE tablename MODIFY columnname NOT NULL NOVALIDATE;

SQL>

ALTER TABLE employee MODIFY empno NOT NULL NOVALIDATE;



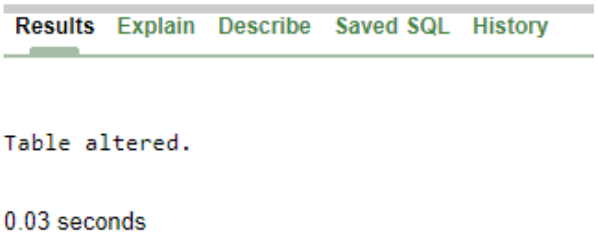
To add Primary key to the employee table.

SYNTAX:

ALTER TABLE tablename ADD CONSTRAINT constraintname PRIMARY KEY (column1, column2…columnn);

SQL>

ALTER TABLE employee ADD CONSTRAINT pk_emp PRIMARY KEY (empno);



P4: Insert null values to the employee table and verify the result.

To Insert null values, use the insert command and pass the column value as null. Only NULLABLE columns accept the null values.

SQL>

1. insert into employee (empno, ename, job, manager_no, sal, commission) values(5,'Kumar','Admin',NULL,NULL,NULL);

2. insert into employee (empno, ename, job, manager_no, sal, commission) values (6,'Suhan', NULL, NULL, NULL, NULL);

3. insert into employee (empno, ename, job, manager_no, sal, commission) values (7, NULL, NULL, NULL, NULL, NULL);

Verify the Result

SQL> select * from employee;

Results Explain Describe Saved SQL History

EMPNO	ENAME	JOB	MANAGER_NO	SAL	COMMISSION
3	David	Clerk	1	9000	50
5	Kumar	Admin	-	-	-
1	Abhi	Manager	1	20000	1000
2	Rohan	Analyst	1	10000	100
6	Suhan	-	-	-	-
7	-	-	-	-	-

6 rows returned in 0.02 seconds CSV Export

Conclusion:

All the queries as described in the experiment 1 is executed on the oracle 10g DBMS system. The results are recorded and verified. The modification for the queries is also executed to understand the complete importance of the CREATE, INSERT and SELECT SQL queries.

EXPERIMENT 2:

AIM:

Create a table called Employee that contain attributes EMPNO, ENAME, JOB, MGR, SAL &execute the following.

1. Add a column commission with domain to the Employee table.
2. Insert any five records into the table.
3. Update the column details of job
4. Rename the column of Employ table using alter command.
5. Delete the employee whose Empno is 105.

DESIGN:

TABLE NAME: employee

Fields/Column name	Domain/Data Format (SQL keyword)	Size	Syntax

EMPNO	Number (number)	32-bit integer	empno number(10)
ENAME	Character (VARCHAR2)	10 Characters	ename varchar2(10)
JOB	Character (VARCHAR2)	10 Characters	job varchar2(10)
MGR	Number (number)	32-bit integer	mgr number(10)
SAL	Number (number)	32-bit integer	sal number(10)

PROBLEMS AND SOLUTION:

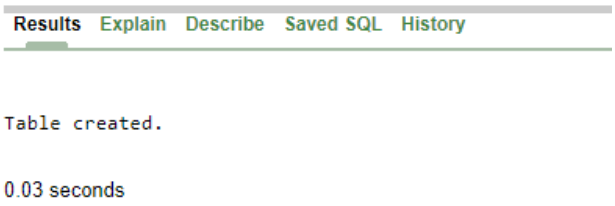
Pre Query:

Create a table called Employee & execute the following.

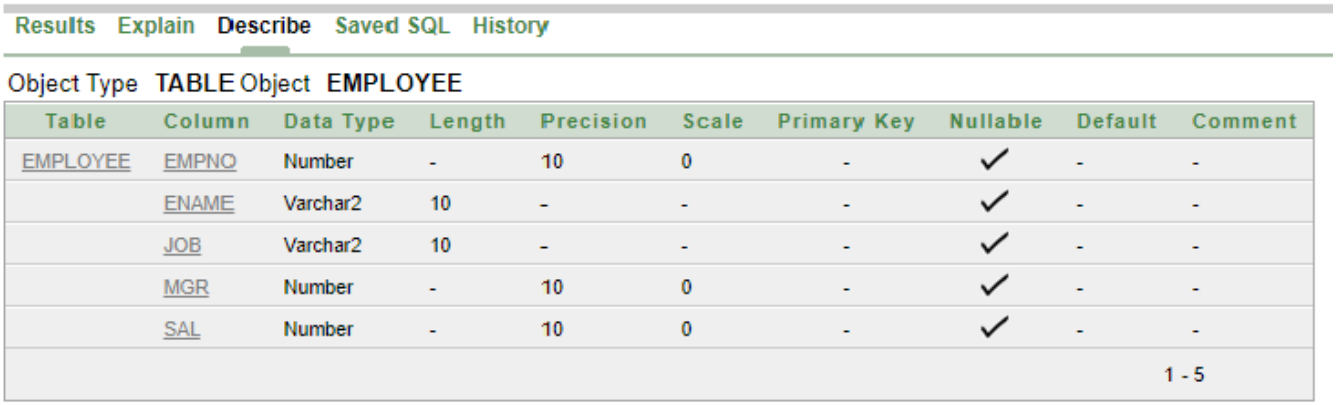
Employee (EMPNO, ENAME, JOB, MGR, SAL)

SOLUTION:

SQL> create table employee (empno number(10), ename varchar2(10), job varchar2(10), mgr number(10), sal number(10));



SQL> desc employee;



P1. Add a column commission with domain to the Employee table.

SOLUTION:

SQL> alter table employee add(commission number(10));

Results Explain Describe Saved SQL History

```
Table altered.

0.00 seconds
```

SQL> desc employee;

Results Explain Describe Saved SQL History

Object Type TABLE Object EMPLOYEE

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
EMPLOYEE	EMPNO	Number	-	10	0	-	✓	-	-
	ENAME	Varchar2	10	-	-	-	✓	-	-
	JOB	Varchar2	10	-	-	-	✓	-	-
	MGR	Number	-	10	0	-	✓	-	-
	SAL	Number	-	10	0	-	✓	-	-
	COMMISSION	Number	-	10	0	-	✓	-	-

1 - 6

P2. Insert any five records into the table.

SOLUTION:

SQL>

1. insert into employee (empno, ename, job, mgr, sal, commission)
values(101,'Kumar','Manager',100,20000,100);

Results Explain Describe Saved SQL History

```
1 row(s) inserted.

0.00 seconds
```

2. insert into employee (empno, ename, job, mgr, sal, commission)
values(102,'Rohan','Analyst',101,10000,100);

3. insert into employee (empno, ename, job, mgr, sal, commission)
values(103,'Kumar','Clerk',101,9000,50);

4. insert into employee (empno, ename, job, mgr, sal, commission)
values(104,'Chandu','Admin',101,9000,50);

5. insert into employee (empno, ename, job, mgr, sal, commission)
values(105,'Keerthi','Designer',101,9000,50);

Practice Query:

Check the result of insertion of rows into the employee table.

SQL> select * from employee;

Results Explain Describe Saved SQL History

EMPNO	ENAME	JOB	MGR	SAL	COMMISSION
101	Kumar	Manager	100	20000	100
102	Rohan	Analyst	101	10000	100
103	Kumar	Clerk	101	9000	50
104	Chandu	Admin	101	9000	50
105	Keerthi	Designer	101	9000	50

5 rows returned in 0.00 seconds CSV Export

P3. Update the column details of job.

SOLUTION:

Syntax: UPDATE table SET column1 = expression1, column2 = expression2, ... column_n = expression_n WHERE conditions;

SQL> update employee set job='trainee' where empno=103;

Results Explain Describe Saved SQL History

```
1 row(s) updated.

0.00 seconds
```

Practice Query:

Check the result of updated rows in the employee table.

SQL> select * from employee;

Results Explain Describe Saved SQL History

EMPNO	ENAME	JOB	MGR	SAL	COMMISSION
101	Kumar	Manager	100	20000	100
102	Rohan	Analyst	101	10000	100
103	Kumar	trainee	101	9000	50
104	Chandu	Admin	101	9000	50
105	Keerthi	Designer	101	9000	50

5 rows returned in 0.00 seconds CSV Export

P4. Rename the column of Employ table using alter command.

SOLUTION:

Syntax: ALTER TABLE table_name MODIFY (column_1 column_type, column_2 column_type, ... column_n column_type);

SQL> alter table employee rename column mgr to manager_no;

Results Explain Describe Saved SQL History

```
Table altered.

0.02 seconds
```

Practice Query:

Check the result of altered column in the employee table.

SQL> desc employee;

Results Explain Describe Saved SQL History

Object Type TABLE Object EMPLOYEE

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
EMPLOYEE	EMPNO	Number	-	10	0	-	✓	-	-
	ENAME	Varchar2	10	-	-	-	✓	-	-
	JOB	Varchar2	10	-	-	-	✓	-	-
	MANAGER_NO	Number	-	10	0	-	✓	-	-
	SAL	Number	-	10	0	-	✓	-	-
	COMMISSION	Number	-	10	0	-	✓	-	-

1 - 6

P5. Delete the employee whose Empno is 105.

SOLUTION:

Syntax: DELETE FROM table_name WHERE conditions;

SQL> delete employee where empno=105;

Results Explain Describe Saved SQL History

```
1 row(s) deleted.

0.00 seconds
```

Practice Query:

Check the result of deleted row in the employee table.

SQL> select * from employee;

Results Explain Describe Saved SQL History

EMPNO	ENAME	JOB	MANAGER_NO	SAL	COMMISSION
101	Kumar	Manager	100	20000	100
102	Rohan	Analyst	101	10000	100
103	Kumar	trainee	101	9000	50
104	Chandu	Admin	101	9000	50

4 rows returned in 0.02 seconds CSV Export

EXPERIMENT 3:

AIM:

Queries using aggregate functions (COUNT, AVG, MIN, MAX, SUM), Group by, Order by.

Employee (E_id, E_name, Age, Salary)

1. Create Employee table containing all Records E_id, E_name, Age, Salary.
2. Count number of employee names from employee table
3. Find the Maximum age from employee table.
4. Find the Minimum age from employee table.
5. Find salaries of employee in Ascending Order.
6. Find grouped salaries of employees.

DESIGN:

TABLE NAME: employee

Fields/Column name	Domain/Data Format (SQL keyword)	Size	Syntax
EMPNO	Number (number)	32-bit integer	empno number(10)
ENAME	Character (VARCHAR2)	10 Characters	ename varchar2(10)
age	Number (number)	32-bit integer	age number(10)
SAL	Number (number)	32-bit integer	sal number(10)

PROBLEMS AND SOLUTION:

P1:

Create a table called Employee & execute the following.

Employee (EMPNO, ENAME, AGE, SAL)

SOLUTION:

SQL> create table employee (empno number(10), ename varchar2(10), age number(10), sal number(10));

Results Explain Describe Saved SQL History

```
Table created.

0.01 seconds
```

Practice Query:

Check the result of the employee table.

SQL> desc employee;

Results Explain Describe Saved SQL History

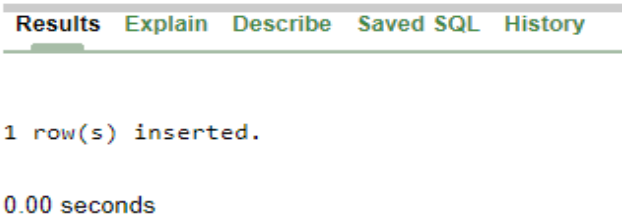
Object Type TABLE Object EMPLOYEE

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
EMPLOYEE	EMPNO	Number	-	10	0	-	✓	-	-
	ENAME	Varchar2	10	-	-	-	✓	-	-
	AGE	Number	-	10	0	-	✓	-	-
	SAL	Number	-	10	0	-	✓	-	-

1 - 4

Insert atleast five rows into the employee table.

1. insert into employee (empno, ename, age, sal) values(101,'Kumar',45,20000);



2. insert into employee (empno, ename, age, sal) values(102,'Rohan',42,10000);
3. insert into employee (empno, ename, age, sal) values(103,'Kumar',38,9000);
4. insert into employee (empno, ename, age, sal) values(104,'Chandu',35,9000);
5. insert into employee (empno, ename, age, sal) values(105,'Keerthi',36,9000);

Practice Query:

Check the result of insertion of rows in the employee table.

SQL> select * from employee;

Results Explain Describe Saved SQL History

EMPNO	ENAME	AGE	SAL
101	Kumar	45	20000
102	Rohan	42	10000
103	Kumar	38	9000
104	Chandu	35	9000
105	Keerthi	36	9000

5 rows returned in 0.00 seconds CSV Export

P2: Count number of employee names from employee table.

SOLUTION:

SQL> select count(ename) from employee;

Results Explain Describe Saved SQL History

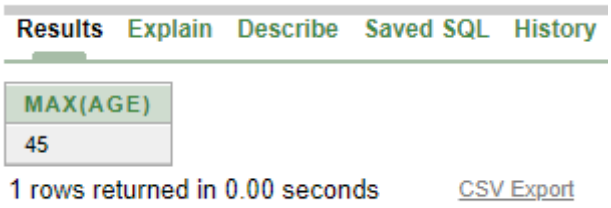
COUNT(ENAME)
5

1 rows returned in 0.00 seconds CSV Export

P3: Find the Maximum age from employee table.

SOLUTION:

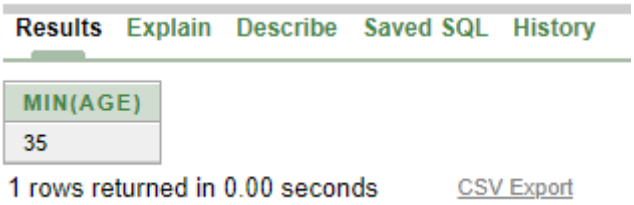
SQL> select max(age) from employee;



P4: Find the Minimum age from employee table.

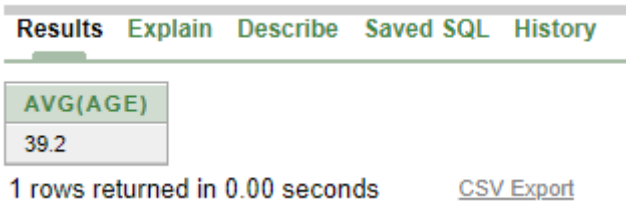
SOLUTION:

SQL> select min(age) from employee;



Practice Query:

SQL> select avg(age) from employee;



P5: Find salaries of employee in Ascending Order.

SOLUTION:

SQL> select ename, sal from employee order by sal;

Results Explain Describe Saved SQL History

ENAME	SAL
Kumar	9000
Chandu	9000
Keerthi	9000
Rohan	10000
Kumar	20000

5 rows returned in 0.01 seconds CSV Export

Practice Query:

Find salaries of employee in DescendingOrder.

SQL> select ename,sal from employee order by sal desc;

Results Explain Describe Saved SQL History

ENAME	SAL
Kumar	20000
Rohan	10000
Chandu	9000
Keerthi	9000
Kumar	9000

5 rows returned in 0.00 seconds CSV Export

P6: Find grouped salaries of employees.

SOLUTION:

SQL> select sal from employee group by sal;

Results Explain Describe Saved SQL History

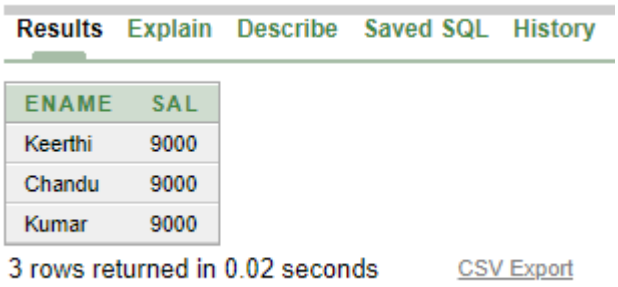
SAL
9000
10000
20000

3 rows returned in 0.00 seconds CSV Export

Practice Query:

Find employee name and salaries of employee whose age is below 40 and salary is above 5000.

SQL> select ename, sal from employee where age<40 group by ename, sal having sal>5000;



Results Explain Describe Saved SQL History

ENAME	SAL
Keerthi	9000
Chandu	9000
Kumar	9000

3 rows returned in 0.02 seconds CSV Export

EXPERIMENT 4:

AIM:

Create a row level trigger for the customers table that would fire for INSERT or UPDATE or DELETE operations performed on the CUSTOMERS table. This trigger will display the salary difference between the old & new Salary.

CUSTOMERS (ID, NAME, AGE, ADDRESS, SALARY)

DESIGN:

TABLE NAME: customers

Fields/Column name	Domain/Data Format (SQL keyword)	Size	Syntax
ID	Number (number)	32-bit integer	id number(10)
NAME	Character (VARCHAR2)	30 Characters	name varchar2(30)
AGE	Number (number)	32-bit integer	age number(10)
ADDRESS	Number (number)	32-bit integer	address varchar2(50)
SALARY	Number (number)	32-bit integer	salary number(10)

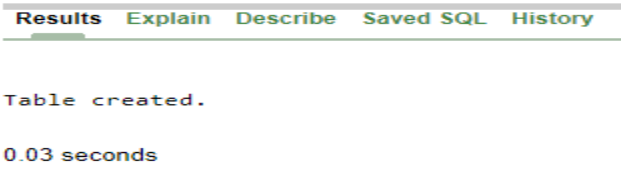
PRE QUERIES:

Create a table called customers & execute the following.

customers (ID, NAME, AGE, ADDRESS, SALARY)

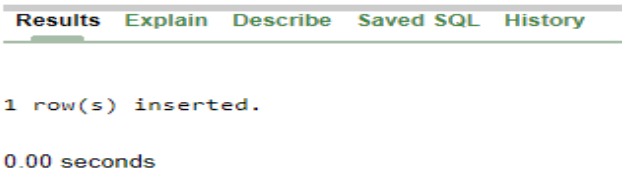
SOLUTION:

SQL> create table customers (id number(10), name varchar2(30), age number(10), address varchar2(50), salary number(10));



Insert atleast five rows into the employee table.

1. insert into customers (id, name, age, address, salary) values(101,'Kumar',45, 'Bengalure',20000);



2. insert into customers (id, name, age, address, salary) values(102,'Rohan',42, 'Tumkuru', 10000);

3. insert into customers (id, name, age, address, salary) values(103,'Kumar',38, 'Belagavi', 9000);

4. insert into customers (id, name, age, address, salary) values(104,'Chandu',35, 'Tiptur ', 9000);

5. insert into customers (id, name, age, address, salary) values(105,'Keerthi',36, 'Mangaluru',9000);

Practice Query:

Check the result of insertion of rows in the employee table.

SQL> select * from employee;

Results Explain Describe Saved SQL History

ID	NAME	AGE	ADDRESS	SALARY
101	Kumar	45	Bengalure	20000
102	Rohan	42	Tumkuru	10000
103	Kumar	38	Belagavi	9000
104	Chandu	35	Tiptur	9000

4 rows returned in 0.00 seconds CSV Export

PROBLEMS AND SOLUTION:

P1: Create a row level trigger for the customers table that would fire for INSERT or UPDATE or DELETE operations performed on the CUSTOMERS table. This trigger will display the salary difference between the old & new Salary.

SOLUTION:

SQL>

```
CREATE OR REPLACE TRIGGER display_salary_changes
    BEFORE DELETE OR INSERT OR UPDATE
    ON customers FOR EACH ROW WHEN (NEW.ID > 0)
    DECLARE sal_diff number;
    BEGIN
        sal_diff := :NEW.salary - :OLD.salary;
        dbms_output.put_line('Old salary: ' || :OLD.salary);
        dbms_output.put_line('New salary: ' || :NEW.salary);
        dbms_output.put_line('Salary difference: ' || sal_diff);
    END;
```


```
Trigger created.

0.12 seconds
```

To check whether trigger works during INSERT statement.

SQL>

insert into customers (id, name, age, address, salary) values (105,'Keerthi',36, 'Mangaluru',9000);


```
Old salary:
New salary: 9000
Salary difference:

1 row(s) inserted.

0.00 seconds
```

To check whether trigger works during UPDATE statement.

SQL>

update customers SET salary=10000 where id=105;


```
Old salary: 9000
New salary: 10000
Salary difference: 1000

1 row(s) updated.

0.00 seconds
```

To check whether trigger works during DELETE statement.

SQL>

delete from customers where id=105;


```
1 row(s) deleted.

0.00 seconds
```

EXPERIMENT 5:

AIM:

Create cursor for Employee table & extract the values from the table. Declare the variables, Open the cursor & extract the values from the cursor. Close the cursor.

Employee(E_id, E_name, Age, Salary)

DESIGN:

TABLE NAME: employee

Fields/Column name	Domain/Data Format (SQL keyword)	Size	Syntax
E_ID	Number (number)	32-bit integer	e_id number(10)
E_NAME	Character (VARCHAR2)	30 Characters	e_name varchar2(30)
AGE	Number (number)	32-bit integer	age number(10)
SALARY	Number (number)	32-bit integer	salary number(10)

PRE QUERIES:

Create a table called customers & execute the following.

employee (E_ID,E_NAME, AGE, SALARY)

SOLUTION:

SQL> create table employee (e_id number(10), e_name varchar2(30), age number(10), salary number(10));

Results Explain Describe Saved SQL History

```
Table created.

0.03 seconds
```

SQL> desc employee;

Results Explain Describe Saved SQL History

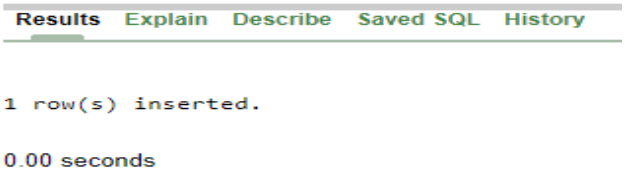
Object Type TABLE Object EMPLOYEE

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
EMPLOYEE	E_ID	Number	-	10	0	-	✓	-	-
	E_NAME	Varchar2	30	-	-	-	✓	-	-
	AGE	Number	-	10	0	-	✓	-	-
	SALARY	Number	-	10	0	-	✓	-	-

1 - 4

Insert atleast five rows into the employee table.

1. insert into employee (e_id, e_name, age, salary) values(101,'Kumar',45,20000);



2. insert into employee (e_id, e_name, age, salary) values(102,'Rohan',42, 10000);

3. insert into employee (e_id, e_name, age, salary) values(103,'Kumar',38, 9000);

4. insert into employee (e_id, e_name, age, salary) values(104,'Chandu',35, 9000);

5. insert into employee (e_id, e_name, age, salary) values(105,'Keerthi',36, 9000);

Practice Query:

Check the result of insertion of rows in the employee table.

SQL> select * from employee;

Results Explain Describe Saved SQL History

E_ID	E_NAME	AGE	SALARY
101	Kumar	45	20000
102	Rohan	42	10000
103	Kumar	38	9000
104	Chandu	35	9000
105	Keerthi	36	9000

5 rows returned in 0.01 seconds CSV Export

PROBLEMS AND SOLUTION:

P1: Create cursor for Employee table & extract the values from the table. Declare the variables, Open the cursor & extract the values from the cursor. Close the cursor.

Employee(E_id, E_name, Age, Salary)

SOLUTION:

SQL>

```
DECLARE   CURSOR emp_cursor IS     SELECT E_id, E_name, Age, Salary FROM Employee;
  v_e_id NUMBER;
  v_e_name VARCHAR2(20);
  v_age NUMBER;
  v_salary NUMBER;
```

```
BEGIN
  OPEN emp_cursor;
  LOOP
    FETCH emp_cursor INTO v_e_id, v_e_name, v_age, v_salary;
    EXIT WHEN emp_cursor%NOTFOUND;
    DBMS_OUTPUT.PUT_LINE('Employee ID: ' || v_e_id);
    DBMS_OUTPUT.PUT_LINE('Employee Name: ' || v_e_name);
    DBMS_OUTPUT.PUT_LINE('Employee Age: ' || v_age);
    DBMS_OUTPUT.PUT_LINE('Employee Salary: ' || v_salary);
  END LOOP;
  CLOSE emp_cursor;
END;
```

EXPERIMENT 6:

AIM:

Write a PL/SQL block of code using parameterized Cursor, that will merge the data availablein 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.

DESIGN:

TABLE NAME: N_RollCall

Fields/Column name	Domain/Data Format (SQL keyword)	Size	Syntax
ROLL	Number (number)	32-bit integer	roll number(10)
NAME	Character (VARCHAR2)	30 Characters	name varchar2(30)

TABLE NAME: O_RollCall

Fields/Column name	Domain/Data Format (SQL keyword)	Size	Syntax
ROLL	Number (number)	32-bit integer	roll number(10)
NAME	Character (VARCHAR2)	30 Characters	name varchar2(30)

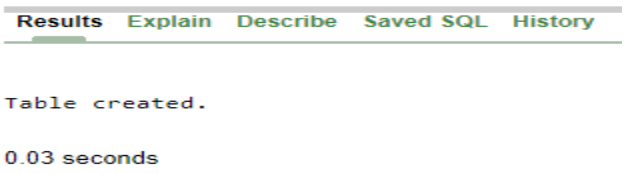
PRE QUERIES:

Create a table called N_RollCall & execute the following.

N_RollCall (ROLL,NAME)

SOLUTION:

SQL> create table N_RollCall (roll number(10), name varchar2(30));



SQL> desc N_RollCall;

Results Explain Describe Saved SQL History

Object Type TABLE Object N_ROLLCALL

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
N_ROLLCALL	ROLL	Number	-	10	0	-	✓	-	-
	NAME	Varchar2	30	-	-	-	✓	-	-

1 - 2

Insert atleast five rows into the N_RollCall table.

1. insert into N_RollCall (roll,name) values(101,'Kumar');

Results Explain Describe Saved SQL History

```
1 row(s) inserted.

0.00 seconds
```

2. insert into N_RollCall (roll,name)values(102,'Rohan');

3. insert into N_RollCall (roll,name) values(103,'Kumar');

4. insert into N_RollCall (roll,name) values(104,'Chandu');

5. insert into N_RollCall (roll,name) values(105,'Keerthi');

Practice Query:

Check the result of insertion of rows in the N_RollCall table.

SQL> select * from N_RollCall;

Results Explain Describe Saved SQL History

ROLL	NAME
101	Kumar
102	Rohan
103	Kumar
104	Chandu
105	Keerthi

5 rows returned in 0.00 seconds CSV Export

Create a table called O_RollCall & execute the following.

O_RollCall (ROLL,NAME)

SOLUTION:

SQL> create table O_RollCall (roll number(10), name varchar2(30));

Results Explain Describe Saved SQL History

```
Table created.

0.03 seconds
```

SQL> desc O_RollCall;

Results Explain **Describe** Saved SQL History

Object Type TABLE Object O_ROLLCALL

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
O_ROLLCALL	ROLL	Number	-	10	0	-	✓	-	-
	NAME	Varchar2	30	-	-	-	✓	-	-

1 - 2

Insert atleast five rows into the O_RollCall table.

1. insert into O_RollCall (roll,name) values(201,'Abhi');

Results Explain Describe Saved SQL History

```
1 row(s) inserted.

0.00 seconds
```

2. insert into O_RollCall (roll,name)values(202,'Rohit');
3. insert into O_RollCall (roll,name) values(203,'Kumar');
4. insert into O_RollCall (roll,name) values(204,'Chandu');
5. insert into O_RollCall (roll,name) values(505,'Keerthi');

Practice Query:

Check the result of insertion of rows in the O_RollCall table.

SQL> select * from O_RollCall;

Results Explain Describe Saved SQL History

ROLL	NAME
201	Abhi
202	Rohit
203	Kumar
204	Chandu
505	Keerthi

5 rows returned in 0.00 seconds CSV Export

PROBLEMS AND SOLUTION:

P1: Write a PL/SQL block of code using parameterized Cursor, that will merge the data availablein 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.

SOLUTION:

SQL>

```
DECLARE
  CURSOR c1 IS SELECT * FROM O_RollCall;
   roll_no number;
    counter number(10);
BEGIN
  counter:=0;
  FOR c1_rec IN c1 LOOP
           select count(*) into roll_no from N_RollCall where roll=c1_rec.roll;
      if roll_no=0 then
        INSERT INTO N_RollCall VALUES (c1_rec.roll, c1_rec.name);
          counter:=counter+1;
      end if;
  END LOOP;
  COMMIT;
  DBMS_OUTPUT.PUT_LINE ('Number of Rows Inserted:'||counter);
END;
```

Results Explain Describe Saved SQL History

```
Number of Rows Inserted:6

Statement processed.

0.00 seconds
```

EXPERIMENT 7:

AIM:

Install an Open Source NoSQL Data base MangoDB & perform basic CRUD (Create, Read, Update & Delete) operations. Execute MangoDB basic Queries using CRUD operations.

DESIGN:

TABLE NAME: N_RollCall

Fields/Column name	Domain/Data Format (SQL keyword)	Size	Syntax
ROLL	Number (number)	32-bit integer	roll number(10)
NAME	Character (VARCHAR2)	30 Characters	name varchar2(30)

TABLE NAME: O_RollCall

Fields/Column name	Domain/Data Format (SQL keyword)	Size	Syntax
ROLL	Number (number)	32-bit integer	roll number(10)
NAME	Character (VARCHAR2)	30 Characters	name varchar2(30)

PROBLEM 1:

Install an Open Source NoSQL Data base MangoDB.

SOLUTION:

MongoDB is an open-source document-oriented database. It is categorized under the NoSQL(Not only SQL) database because the storage and retrieval of data in MongoDB are not in the form of tables.

Requirements to Install MongoDB on Windows

MongoDB 4.4 and later only support 64-bit versions of Windows.

MongoDB 7.0 Community Edition supports the following 64-bit versions of Windows on x86_64 architecture:

Windows Server 2022

Windows Server 2019

Windows 11

Ensure that the user is running mongod and mongos has the necessary permissions from the following groups:

Performance Monitor Users

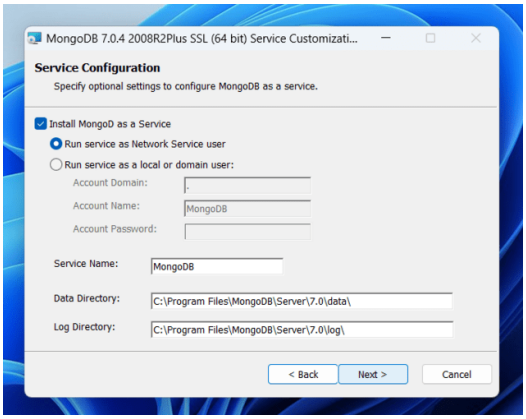
Performance Log Users

Install MangoDB:

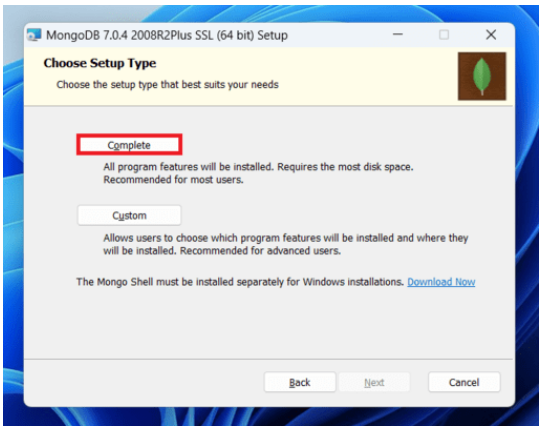
Step1: To install MongoDB on windows, first, download the MongoDB server and then install the MongoDB shell.

Step 2: When the download is complete open the msi file and click the next button in the startup screen:

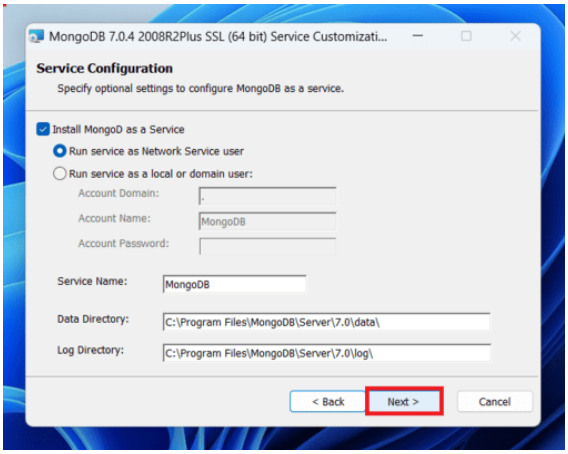
Step 3: Now accept the End-User License Agreement and click the next button:



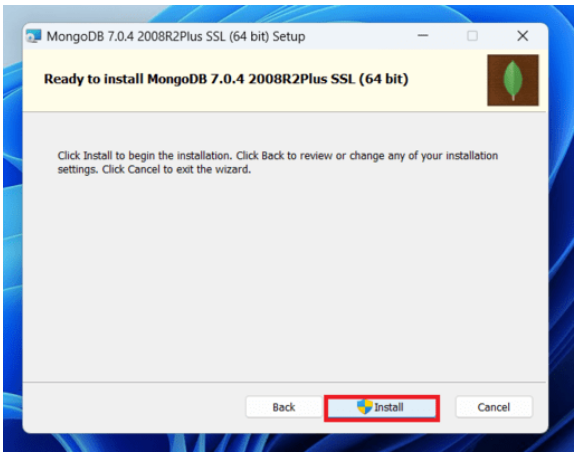
Step 4: Now select the complete option to install all the program features.



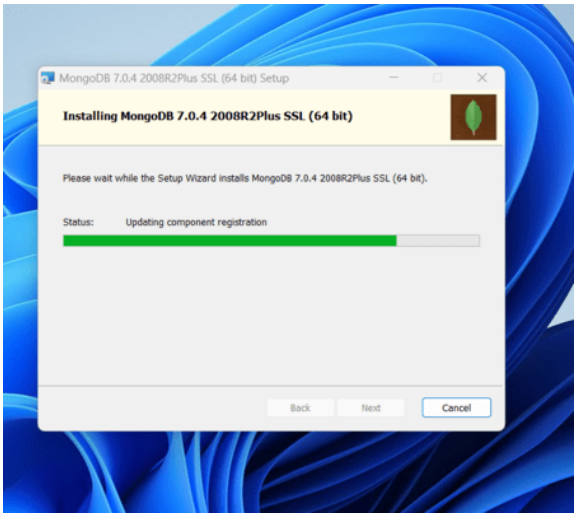
Step 5: Select “Run service as Network Service user” and copy the path of the data directory. Click Next:



Step 6: Click the Install button to start the MongoDB installation process:

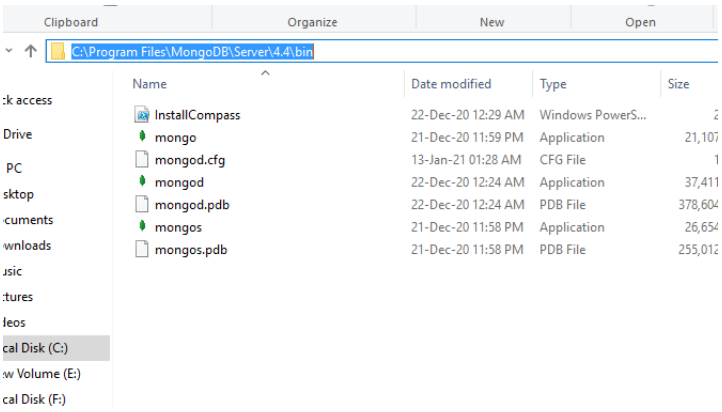


Step 7: After clicking on the install button installation of MongoDB begins:

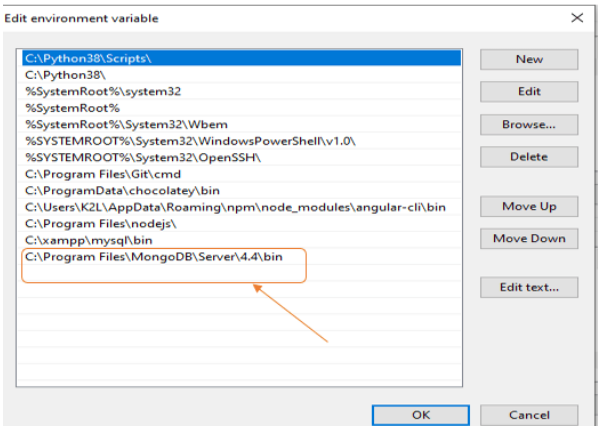


Step 8: Now click the Finish button to complete the MongoDB installation process:

Step 9: Now we go to the location where MongoDB installed in step 5 in your system and copy the bin path:



Step 10: Now, to create an environment variable open system properties << Environment Variable << System variable << path << Edit Environment variable and paste the copied link to your environment system and click Ok:



Step 11: After setting the environment variable, we will run the MongoDB server, i.e. mongod. So, open the command prompt and run the following command:

mongod

When you run this command you will get an error i.e. C:/data/db/ not found.

Step 12: Now, Open C drive and create a folder named “data” inside this folder create another folder named “db”. After creating these folders. Again open the command prompt and run the following command:

mongod

Now, this time the MongoDB server(i.e., mongod) will run successfully.

```
C:\Users\NIkhil Chhipa>mongod
{"t":{"$date":"2021-01-31T00:56:54.081+05:30"},"s":"I",  "c":"CONTROL",  "id":23285,   "ctx"
ify --sslDisabledProtocols 'none'"}
{"t":{"$date":"2021-01-31T00:56:54.087+05:30"},"s":"W",  "c":"ASIO",     "id":22601,   "ctx"
}
{"t":{"$date":"2021-01-31T00:56:54.088+05:30"},"s":"I",  "c":"NETWORK",  "id":4648602, "ctx"
{"t":{"$date":"2021-01-31T00:56:54.090+05:30"},"s":"I",  "c":"STORAGE",  "id":4615611, "ctx"
bPath":"C:/data/db/","architecture":"64-bit","host":"DESKTOP-L9MUQ7N"}}
{"t":{"$date":"2021-01-31T00:56:54.090+05:30"},"s":"I",  "c":"CONTROL",  "id":23398,   "ctx"
rgetMinOS":"Windows 7/Windows Server 2008 R2"}}
{"t":{"$date":"2021-01-31T00:56:54.090+05:30"},"s":"I",  "c":"CONTROL",  "id":23403,   "ctx"
gitVersion":"913d6b62acfbb344dde1b116f4161360acd8fd13","modules":[],"allocator":"tcmalloc","
}}}}
{"t":{"$date":"2021-01-31T00:56:54.090+05:30"},"s":"I",  "c":"CONTROL",  "id":51765,   "ctx"
ndows 10","version":"10.0 (build 14393)"}}}
{"t":{"$date":"2021-01-31T00:56:54.090+05:30"},"s":"I",  "c":"CONTROL",  "id":21951,   "ctx"
{"t":{"$date":"2021-01-31T00:56:54.157+05:30"},"s":"I",  "c":"STORAGE",  "id":22270,   "ctx"
:{"dbpath":"C:/data/db/","storageEngine":"wiredTiger"}}
{"t":{"$date":"2021-01-31T00:56:54.158+05:30"},"s":"I",  "c":"STORAGE",  "id":22315,   "ctx"
ize=1491M,session_max=33000,eviction=(threads_min=4,threads_max=4),config_base=false,statist
le_manager=(close_idle_time=100000,close_scan_interval=10,close_handle_minimum=250),statisti
ess],"}}
{"t":{"$date":"2021-01-31T00:56:54.395+05:30"},"s":"I",  "c":"STORAGE",  "id":22430,   "ctx"
95788][3708:140713908197088], txn-recover: [WT_VERB_RECOVERY_PROGRESS] Recovering log 20 thr
{"t":{"$date":"2021-01-31T00:56:54.631+05:30"},"s":"I",  "c":"STORAGE",  "id":22430,   "ctx"
```

Run mongo Shell

Step 13: Now we are going to connect our server (mongod) with the mongo shell. So, keep that mongod window and open a new command prompt window and write mongo. Now, our mongo shell will successfully connect to the mongod.

PROBLEM 1:

Install an Open Source NoSQL Data base MangoDB.

SOLUTION:

SQL>

perform basic CRUD (Create, Read, Update & Delete) operations. Execute MangoDB basic Queries using CRUD operations.

P1: CREATE OPERATION

SOLUTION:

Create a New Database Using Mongo Shell

```
use college
```

Show List of Databases

```
show dbs
```

Check Current Database

```
db
```

Create Operations

Method	Description
db.collection.insertOne()	It is used to insert a single document in the collection.
db.collection.insertMany()	It is used to insert multiple documents in the collection.
db.createCollection()	It is used to create an empty collection.

Exercise1:

Insert Single Document

```
db.student.insert({Name: "Akshay", Marks: 500})
db.student.find().pretty()
db.student.insertOne({Name: "Akshay", Marks: 500})
db.student.insertOne({_id: "Stu102", Name: "Vishal", Marks: 230})
db.student.insertMany([{name:"Ajay",age:20},
                {name:"Bina",age:24},
                {name:"Ram",age:23}])
db.student.insertMany([{_id:"stu200", name:"Ammu", age:18},
                {_id:"stu201", name:"Priya", age:29}])
```

In MongoDB, the Bulk.insert() method is used to perform insert operations in bulk.

```
var bulk = db.students.initializeUnorderedBulkOp();
bulk.insert( { first_name: "Sachin", last_name: "Tendulkar" } );
bulk.insert( { first_name: "Virender", last_name: "Sehwag" } );
bulk.insert( { first_name: "Shikhar", last_name: "Dhawan" } );
bulk.insert( { first_name: "Mohammed", last_name: "Shami" } );
bulk.insert( { first_name: "Shreyas", last_name: "Iyer" } );
bulk.execute();
```

P2: READ OPERATION

SOLUTION:

```
db.student.find({age:18})
db.student.find()
db.student.find({score:{math: 230, science: 234}})
```

db.student.findOne({language:"c++"})

db.student.findOne({name: "Avinash"}, {_id: 0, language:1})

P3: UPDATE OPERATION

SOLUTION:

db.student.update({name:"avi"},{$set:{name:"helloword"}})

db.student.update({name:"prachi"},{$set:{age:20}}

db.student.updateOne({name: "Annu"}, {$set:{age:25}})

db.student.updateOne({name:"Bhannu"},{$set:{name:"Babita"}})

P4: DELETE OPERATION

SOLUTION:

db.student.remove({name: "Akshay"})

db.student.remove({})

db.student.remove({age:{$eq:18}}, true)

db.student.deleteOne({age:17})

db.student.deleteOne({age:{$lt:18}})

BASIC OPERATION1:

SOLUTION:

We will drop the student collection in the gfg database. It drops the student collection and all the indexes associated with the collection:

db.student.drop()

BASIC OPERATION2:

SOLUTION:

MongoDB distinct() method finds the distinct values for a given field across a single collection and returns the results in an array. It takes three parameters, first one is the field for which to return distinct values and the others are optional.

```
> use gfg
switched to db gfg
> db.student.find().pretty()
{
        "_id" : ObjectId("6012ce730cf217478ba9358c"),
        "name" : "Nikhil",
        "language" : "C++"
}
{
        "_id" : ObjectId("6012cefe0cf217478ba9358d"),
        "name" : "Ammu",
        "language" : "Python",
        "detail" : {
                "age" : 23,
                "branch" : "CSE"
        }
}
{
        "_id" : ObjectId("6012cf250cf217478ba9358e"),
        "name" : "Avinash",
        "language" : "Python",
        "marks" : [
                23,
                45
        ]
}
>
```

db.student.distinct("name")

db.student.distinct("detail.age")

db.student.distinct("marks")

BASIC OPERATION3:

SOLUTION:

In MongoDB, the limit() method limits the number of records or documents that you want. It basically defines the max limit of records/documents that you want. Or in other words, this method uses on cursor to specify the maximum number of documents/ records the cursor will return.

db.gfg.find().limit(2)

db.gfg.find({"content":/c/i}).limit(2)

Here, content is key were we will check whether it contains ‘c’ character in the string or not. /c/ denotes that we are looking for strings that contain this ‘c’ character and in the end of /c/i, i denotes that it is case-insensitive.

db.gfg.find({"content":/c/i}).limit(3)

BASIC OPERATION4:

SOLUTION:

MongoDB provides a createIndex() method to create one or more indexes on collections. Using this method we can create different types of indexes like text index, 2dsphere index, 2d index, etc.

db.student.createIndex({name:1})

db.student.createIndex({language:-1})

db.student.createIndex({name:1,language:-1})

db.student.createIndex({name:1},{unique:true})

db.student.createIndex({"branch.$**":1})

BASIC OPERATION5:

SOLUTION:

MongoDB – Comparison Query Operators

Operators	Description
$eq	Matches the values of the fields that are equal to a specified value.
$ne	Matches all values of the field that are not equal to a specified value.
$gt	Matches values of the fields that are greater than a specified value.
$gte	Matches values of the fields that are greater than equal to the specified value.
$lt	Matches values of the fields that are less than a specified value
$lte	Matches values of the fields that are less than equal to the specified value
$in	Matches any of the values specified in an array.
$nin	Matches none of the values specified in an array.

```
> db.contributor.find()
{ "_id" : ObjectId("5e6f7a6692e6dfa3fc48ddbe"), "name" : "Rohit", "branch" : "
CSE", "joiningYear" : 2018, "language" : [ "C#", "Python", "Java" ], "personal
" : { "contactinfo" : 0, "state" : "Delhi", "age" : 24, "semesterMarks" : [ 70
, 73.3, 76.5, 78.6 ] }, "salary" : 1000 }
{ "_id" : ObjectId("5e7b9f0a92e6dfa3fc48ddbf"), "name" : "Amit", "branch" : "E
CE", "joiningYear" : 2017, "language" : [ "Python", "C#" ], "personal" : { "co
ntactinfo" : 234556789, "state" : "UP", "age" : 25, "semesterMarks" : [ 80, 80
.1, 98, 70 ] }, "salary" : 10000 }
{ "_id" : ObjectId("5e7b9f0a92e6dfa3fc48ddc0"), "name" : "Sumit", "branch" : "
CSE", "joiningYear" : 2017, "language" : [ "Java", "Perl" ], "personal" : { "c
ontactinfo" : 2300056789, "state" : "MP", "age" : 24, "semesterMarks" : [ 89,
80.1, 78, 71 ] } }
>
```

db.contributor.find({name: {$nin: ["Amit", "Suman"]}}).pretty()

db.contributor.find({name: {$in: ["Amit", "Suman"]}}).pretty()

db.contributor.find({salary: {$lt: 2000}}).pretty()

db.contributor.find({branch: {$eq: "CSE"}}).pretty()

db.contributor.find({branch: {$ne: "CSE"}}).pretty()

db.contributor.find({salary: {$gt: 1000}}).pretty()

db.contributor.find({joiningYear: {$gte: 2017}})

db.contributor.find({salary: {$lte: 1000}}).pretty()

VIVA QUESTIONS AND ANSWERS

1. What is database?

A database is a collection of information that is organized. So that it can easily be accessed, managed, and updated.

2. What is DBMS?

DBMS stands for Database Management System. It is a collection of programs that enables user to create and maintain a database.

3. What is a Database system?

The database and DBMS software together is called as Database system.

4. What are the advantages of DBMS?

I. Redundancy is controlled.
II. Providing multiple user interfaces.
III. Providing backup and recovery
IV. Unauthorized access is restricted.
V. Enforcing integrity constraints.

5. What is normalization?

It is a process of analysing the given relation schemas based on their Functional Dependencies (FDs) and primary key to achieve the properties

(1) Minimizing redundancy, (2). Minimizing insertion, deletion and update anomalies.

6. What is Data Model?

A collection of conceptual tools for describing data, data relationships data semantics and constraints.

7. What is E-R model?

This data model is based on real world that consists of basic objects called entities and of relationship among these objects. Entities are described in a database by a set of attributes.

8. What is Object Oriented model?

This model is based on collection of objects. An object contains values stored in instance variables with in the object. An object also contains bodies of code that operate on the object.

These bodies of code are called methods. Objects that contain same types of values and the same methods are grouped together into classes.

9. What is an Entity?

An entity is a thing or object of importance about which data must be captured.

10. What is DDL (Data Definition Language)?

A data base schema is specified by a set of definitions expressed by a special language called DDL.

11. What is DML (Data Manipulation Language)?

This language that enable user to access or manipulate data as organized by appropriate data model. Procedural DML or Low level: DML requires a user to specify what data are needed and how to get those data. Non-Procedural DML or High level: DML requires a user to specify what data are needed without specifying how to get those data

12. What is DML Compiler?

It translates DML statements in a query language into low-level instruction that the query evaluation engine can understand.

13. What is Query evaluation engine?

It executes low-level instruction generated by compiler.

14. Under what conditions should indexes be used?

Indexes can be created to enforce uniqueness, to facilitate sorting, and to enable fast retrieval by column values. A good candidate for an index is a column that is frequently used with equal conditions in WHERE clauses.

15. What is difference between SQL and SQL SERVER?

SQL is a language that provides an interface to RDBMS, developed by IBM. SQL SERVER is a RDBMS just like Oracle, DB2.