# 20MCA134 – ADVANCED DBMS LAB

*Lab Report Submitted By*

**JENNY JOHNSON**

**Reg. No.: AJC22MCA-2053**

*In Partial fulfilment for the Award of the Degree Of*

**MASTER OF COMPUTER APPLICATIONS (2 Year) (MCA)**

**APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY**


**AMAL JYOTHI COLLEGE OF ENGINEERING**

**KANJIRAPPALLY**

[Affiliated to APJ Abdul Kalam Technological University, Kerala. Approved by AICTE, Accredited by NAAC with 'A' grade. Koovapally, Kanjirappally, Kottayam, Kerala – 686518]

**2022-2023**

## DEPARTMENT OF COMPUTER APPLICATIONS

## AMAL JYOTHI COLLEGE OF ENGINEERING
## KANJIRAPPALLY

# CERTIFICATE

This is to certify that the lab report, "**20MCA134-ADVANCED DBMS LAB"** is the bonafide work of **JENNY JOHNSON (AJC22MCA-2053)** in partial fulfilment of the requirements for the award of the Degree of Master of Computer Applications under APJ Abdul Kalam Technological University during the year **2022-23.**

Mr. G S Ajith
**Lab In- Charge**

Rev. Fr. Dr. Rubin Thottupurathu Jose
**Head of the Department**

**Internal Examiner**

**External Examiner**

| Course Code | Course Name | Syllabus Year | L-T-P-C |
|---|---|---|---|
| 20MCA134 | Advanced DBMS Lab | 2020 | 0-1-3-2 |

**VISION**

To promote an academic and research environment conducive for innovation centric technical education.

**MISSION**

MS1 - Provide foundations and advanced technical education in both theoretical and applied Computer Applications in-line with Industry demands.

MS2 - Create highly skilled computer professionals capable of designing and innovating real life solutions.

MS3 - Sustain an academic environment conducive to research and teaching focused to generate up-skilledprofessionals with ethical values.

MS4 - Promote entrepreneurial initiatives and innovations capable of bridging and contributing withsustainable, socially relevant technology solutions.

**COURSE OUTCOME**

| CO | Outcome | Target |
|---|---|---|
| CO1 | Design and build a simple relational database system and demonstrate competence with the fundamentals tasks involved with modelling, designing and implementing a database. | 65 |
| CO2 | Apply PL/SQL for processing databases | 65 |
| CO3 | Comparison between relational and non-relational (NoSQL) databases and the configuration of NoSQL Databases. | 65 |
| CO4 | Apply CRUD operations and retrieve data in a NoSQL environment. | 65 |
| CO5 | Understand the basic storage architecture of distributed file systems. | 65 |
| CO6 | Design and deployment of NoSQL databases with real time requirements. | 60 |

**COURSE END SURVEY**

| CO | Survey Question | Answer Format |
|---|---|---|
| CO1 | To what extend you are able to design and build a simple realational database system and demonstrate competence with the fundamentals tasks involved with modelling , designing and implementing a database. | Excellent/Very Good/Good/Fair/Poor |
| CO2 | To what extend you are able to apply PL/SQL for processing datavases. | Excellent/Very Good/Good/Fair/Poor |
| CO3 | To what extend you are able to compare relational and non-relational(NoSQL)databses and configuration of NoSQL Databases. | Excellent/Very Good/Good/Fair/Poor |
| CO4 | To what extend you are able apply CRUD operations and retrieve data in a NoSQL environment. | Excellent/Very Good/Good/Fair/Poor |
| CO5 | To what extent you are able to understand the basic storage architecture of distributed file systems. | Excellent/Very Good/Good/Fair/Poor |
| CO6 | To what extend you are able to design and deployment of NoSQL databases with real time experiments. | Excellent/Very Good/Good/Fair/Poor |

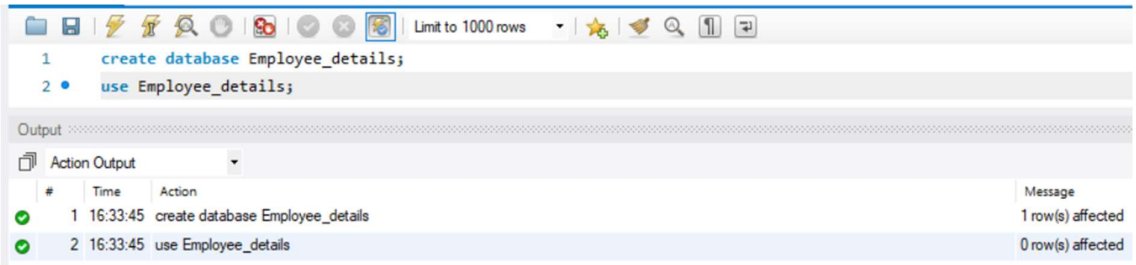
## Experiment No.: 1

**Aim:** Creation of a database using DDL commands.

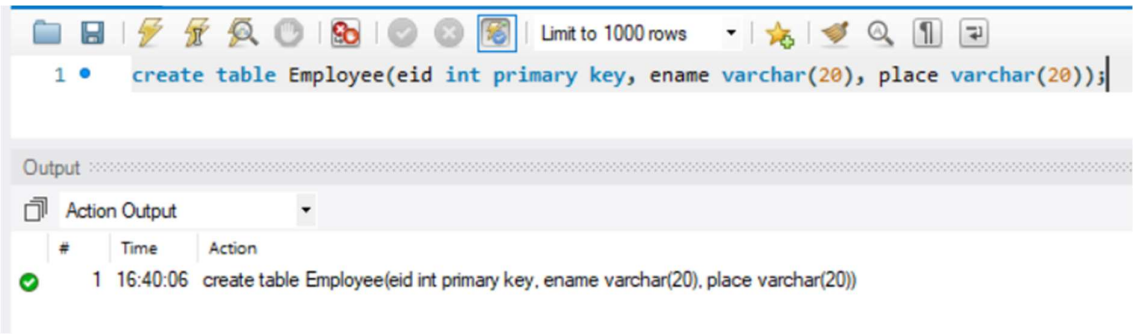
**CO1:** Design and build a simple relational database system and demonstrate competence with the fundamentals tasks involved with modelling, designing and implementing a database.

## Procedure

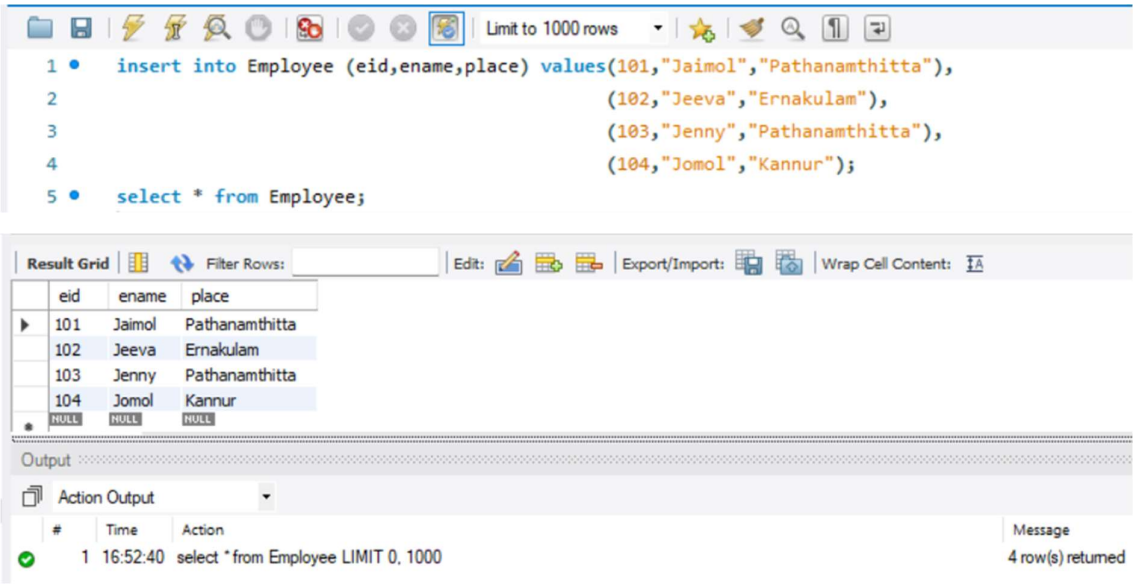
1.Create database Employee_details



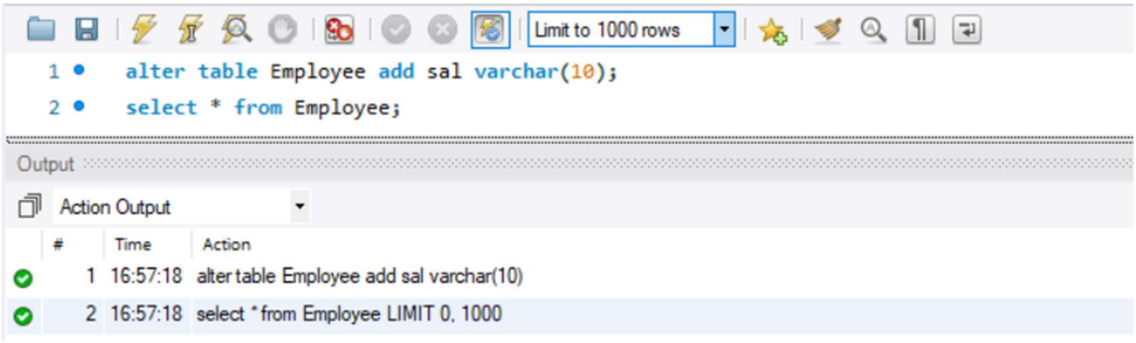

2.Create table Employee

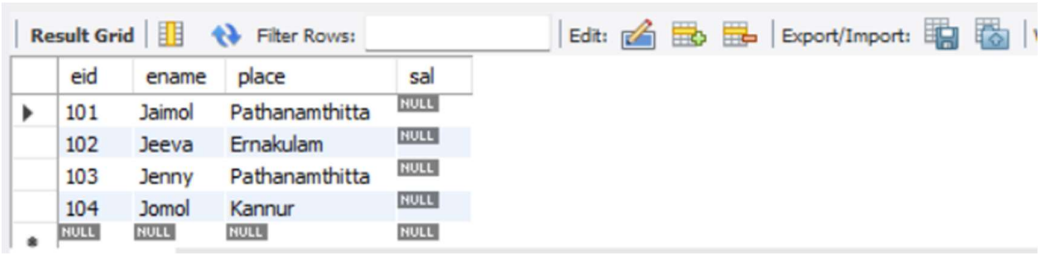



3.Insert values into table Employee

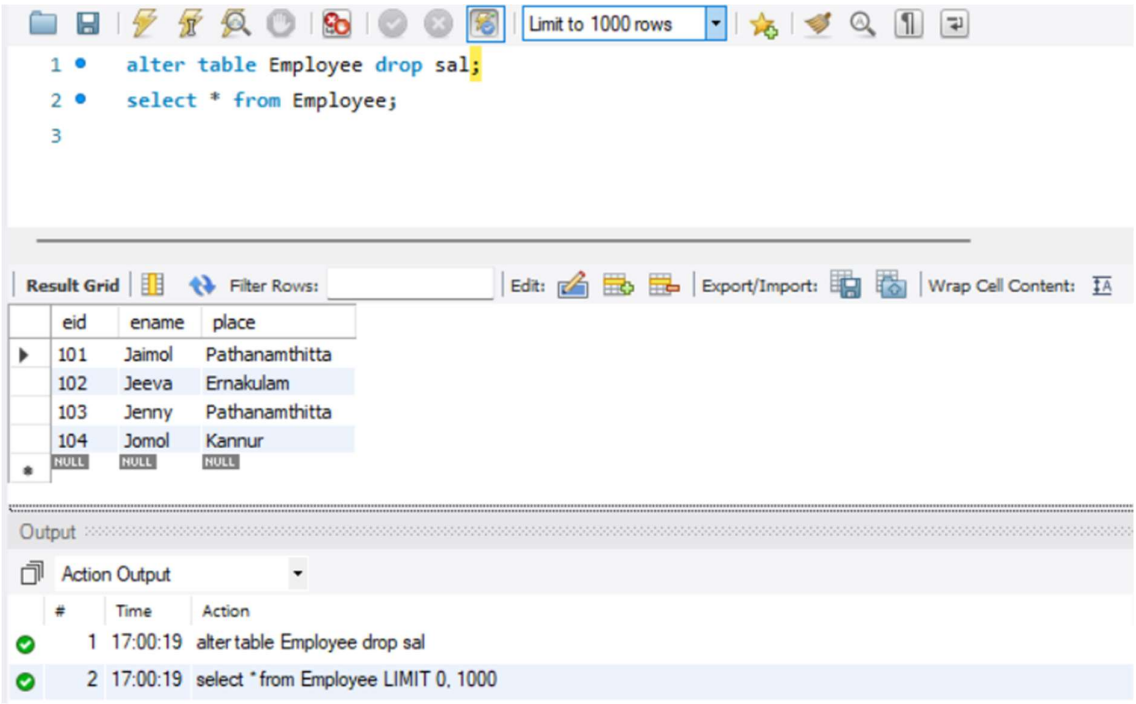


4.Alter table- add column sal to table Employee




5.Drop column sal



6.Rename column place to eplace

```
alter table Employee rename column place to eplace;
select * from Employee;
```

| eid | ename | eplace |
|---|---|---|
| 101 | Jaimol | Pathanamthitta |
| 102 | Jeeva | Ernakulam |
| 103 | Jenny | Pathanamthitta |
| 104 | Jomol | Kannur |

7.Truncate table Employee

```
truncate table Employee;
select * from Employee;
```

| eid | ename | eplace |
|---|---|---|

Output

| # | Time | Action | Message |
|---|---|---|---|
| 1 | 17:06:14 | truncate table Employee | 0 row(s) affected |
| 2 | 17:06:15 | select * from Employee LIMIT 0, 1000 | 0 row(s) returned |

8. Drop table Employee

```
drop table Employee;
select * from Employee;
```

Output

| # | Time | Action |
|---|---|---|
| 1 | 17:08:39 | drop table Employee |
| 2 | 17:08:39 | select * from Employee LIMIT 0, 1000 |

## Result

The program was executed and the result was successfully obtained. Thus, CO1 was obtained.

## Experiment No.: 2

**Aim:** Creation of a database using DML commands including integrity constraints.

**CO1:** Design and build a simple relational database system and demonstrate competence with the fundamentals tasks involved with modelling, designing and implementing a database.

## Procedure

1.Create table - Branch

```
CREATE TABLE BRANCH (
  BNAME VARCHAR(20) PRIMARY KEY,
  CITY VARCHAR(30) CHECK(CITY IN ('NAGPUR', 'DELHI', 'BANGLORE', 'BOMBAY')) NOT NULL
);
```

2. Create table - Customer

```
CREATE TABLE CUSTOMER (
  CNAME VARCHAR(15) PRIMARY KEY,
  CITY VARCHAR(30) NOT NULL
);
```

3. Create table - Deposit

```
CREATE TABLE DEPOSIT (
  ACTNO VARCHAR(5) PRIMARY KEY CHECK (ACTNO LIKE 'D%'),
  CNAME VARCHAR(15) REFERENCES CUSTOMER(CNAME),
  BNAME VARCHAR(20) REFERENCES BRANCH(BNAME),
  AMOUNT DECIMAL(8,2) NOT NULL CHECK (AMOUNT > 0),
  ADATE DATE
);
```

4. Create table - Borrow

```
CREATE TABLE BORROW (
        LOANNO VARCHAR(8) check(LOANNO like 'L%') primary key,
        CNAME VARCHAR(15) REFERENCES CUSTOMER(CNAME),
        BNAME VARCHAR(20) REFERENCES BRANCH(BNAME),
        AMOUNT FLOAT(8) NOT NULL CHECK (AMOUNT > 0)
        );
```

5. Insert values in table - Customer

```
INSERT INTO CUSTOMER (CNAME, CITY) VALUES ('ANIL', 'CALCUTTA');
INSERT INTO CUSTOMER (CNAME, CITY) VALUES ('SUNIL', 'DELHI');
INSERT INTO CUSTOMER (CNAME, CITY) VALUES ('MEHUL', 'BARODA');
INSERT INTO CUSTOMER (CNAME, CITY) VALUES ('MANDAR', 'PATNA');
INSERT INTO CUSTOMER (CNAME, CITY) VALUES ('MADHURI', 'NAGPUR');
INSERT INTO CUSTOMER (CNAME, CITY) VALUES ('PRAMOD', 'NAGPUR');
INSERT INTO CUSTOMER (CNAME, CITY) VALUES ('SANDIP', 'SURAT');
INSERT INTO CUSTOMER (CNAME, CITY) VALUES ('SHIVANI', 'BOMBAY');
INSERT INTO CUSTOMER (CNAME, CITY) VALUES ('KRANTI', 'BOMBAY');
INSERT INTO CUSTOMER (CNAME, CITY) VALUES ('NAREN', 'BOMBAY');
```

6. Insert values in table - Branch

```
INSERT INTO BRANCH (BNAME, CITY) VALUES ('VRCE', 'NAGPUR');
INSERT INTO BRANCH (BNAME, CITY) VALUES ('AJNI', 'NAGPUR');
INSERT INTO BRANCH (BNAME, CITY) VALUES ('KAROLBAGH', 'DELHI');
INSERT INTO BRANCH (BNAME, CITY) VALUES ('CHANDNI', 'DELHI');
INSERT INTO BRANCH (BNAME, CITY) VALUES ('DHARAMPETH', 'NAGPUR');
INSERT INTO BRANCH (BNAME, CITY) VALUES ('MG ROAD', 'BANGLORE');
INSERT INTO BRANCH (BNAME, CITY) VALUES ('ANDHERI', 'BOMBAY');
INSERT INTO BRANCH (BNAME, CITY) VALUES ('NEHRU PALACE', 'DELHI');
INSERT INTO BRANCH (BNAME, CITY) VALUES ('POWAI', 'BOMBAY');
```

7. Insert values in table - Borrow

```
INSERT INTO borrow (LOANNO, CNAME, BNAME, AMOUNT) VALUES ('L201', 'ANIL', 'VRCE', 1000.00);
INSERT INTO borrow (LOANNO, CNAME, BNAME, AMOUNT) VALUES ('L206', 'MEHUL', 'AJNI', 5000.00);
INSERT INTO borrow (LOANNO, CNAME, BNAME, AMOUNT) VALUES ('L311', 'SUNIL', 'DHARAMPETH', 3000.00);
INSERT INTO borrow (LOANNO, CNAME, BNAME, AMOUNT) VALUES ('L321', 'MADHURI', 'ANDHERI', 2000.00);
INSERT INTO borrow (LOANNO, CNAME, BNAME, AMOUNT) VALUES ('L371', 'PRAMOD', 'VIRAR', 8000.00);
INSERT INTO borrow (LOANNO, CNAME, BNAME, AMOUNT) VALUES ('L481', 'KRANTI', 'NEHRU PLACE', 3000.00);
```

8.. Insert values into table - Deposit

```
INSERT INTO Deposit (ACTNO, CNAME, BNAME, AMOUNT, ADATE) VALUES ('D100', 'ANIL', 'VRCE', 1000.00, '1995-03-01');
INSERT INTO Deposit (ACTNO, CNAME, BNAME, AMOUNT, ADATE) VALUES ('D101', 'SUNIL', 'ANJNI', 500.00, '1996-01-04');
INSERT INTO Deposit (ACTNO, CNAME, BNAME, AMOUNT, ADATE) VALUES ('D102', 'MEHUL', 'KAROLBAGH', 3500.00, '1995-11-17');
INSERT INTO Deposit (ACTNO, CNAME, BNAME, AMOUNT, ADATE) VALUES ('D104', 'MADHURI', 'CHANDNI', 1200.00, '1995-12-17');
INSERT INTO Deposit (ACTNO, CNAME, BNAME, AMOUNT, ADATE) VALUES ('D105', 'PRAMOD', 'MG ROAD', 3000.00, '1996-03-27');
INSERT INTO Deposit (ACTNO, CNAME, BNAME, AMOUNT, ADATE) VALUES ('D106', 'SANDIP', 'ANDHERI', 2000.00, '1996-03-31');
INSERT INTO Deposit (ACTNO, CNAME, BNAME, AMOUNT, ADATE) VALUES ('D107', 'SHIVANI', 'VIRAR', 1000.00, '1995-09-05');
INSERT INTO Deposit (ACTNO, CNAME, BNAME, AMOUNT, ADATE) VALUES ('D108', 'KRANTI', 'NEHRU PLACE', 5000.00, '1995-07-02');
INSERT INTO Deposit (ACTNO, CNAME, BNAME, AMOUNT, ADATE) VALUES ('D109', 'MINU', 'POWAI', 7000.00, '1995-08-10');
```

9. Add constraint - salary <= 5000

```
alter table emp add constraint chk_salary check (esalary >= 1500);
```

10. Add constraint - length(eno) >= 3

```
alter table emp add constraint chk_empno_length check (length(eno<=3));
```

## Result

The program was executed and the result was successfully obtained. Thus, CO1 was obtained.

## Experiment No.: 3

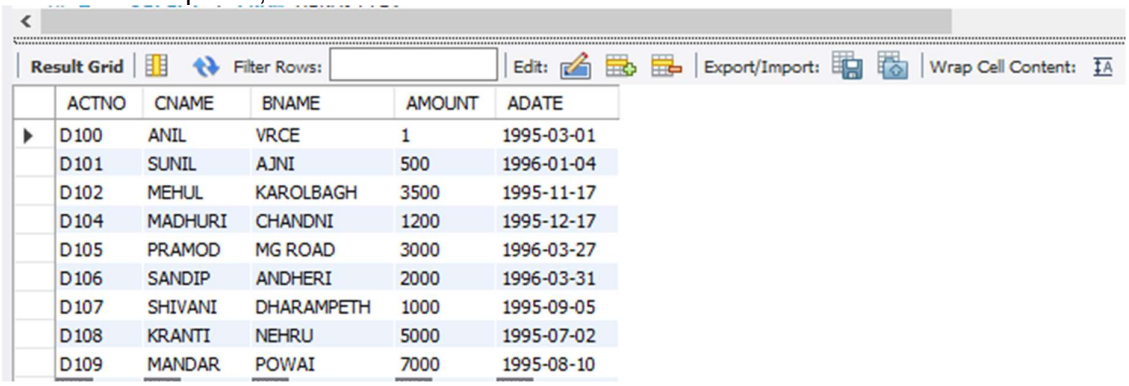
**Aim:** To familiarize with selecting data from single table.

**CO1:** Design and build a simple relational database system and demonstrate competence with the fundamentals tasks involved with modelling, designing and implementing a database.

## Procedure

### 1. list all data from table deposite

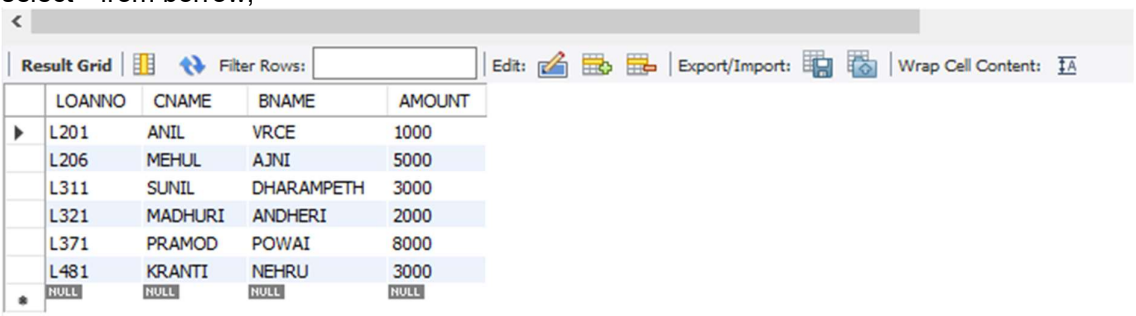
select * from deposite;



| ACTNO | CNAME | BNAME | AMOUNT | ADATE |
|---|---|---|---|---|
| D100 | ANIL | VRCE | 1 | 1995-03-01 |
| D101 | SUNIL | AJNI | 500 | 1996-01-04 |
| D102 | MEHUL | KAROLBAGH | 3500 | 1995-11-17 |
| D104 | MADHURI | CHANDNI | 1200 | 1995-12-17 |
| D105 | PRAMOD | MG ROAD | 3000 | 1996-03-27 |
| D106 | SANDIP | ANDHERI | 2000 | 1996-03-31 |
| D107 | SHIVANI | DHARAMPETH | 1000 | 1995-09-05 |
| D108 | KRANTI | NEHRU | 5000 | 1995-07-02 |
| D109 | MANDAR | POWAI | 7000 | 1995-08-10 |

### 2. list all data from borrow

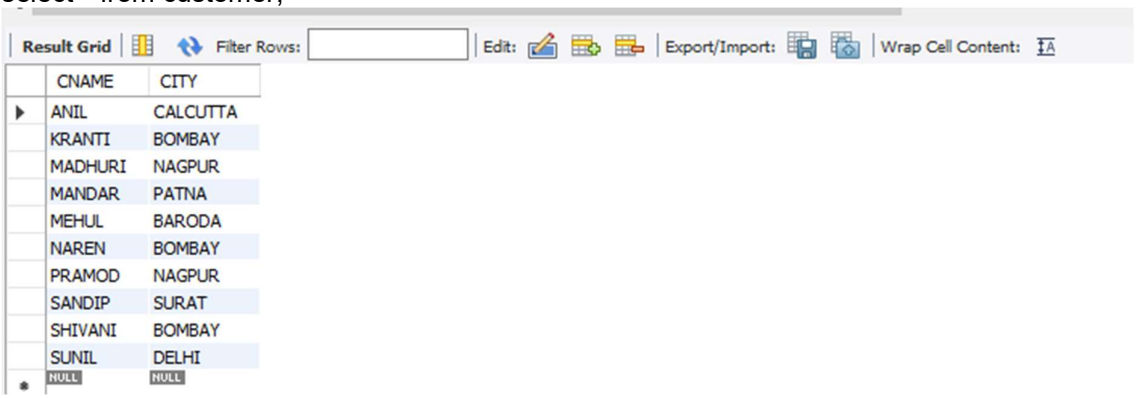
select * from borrow;



| LOANNO | CNAME | BNAME | AMOUNT |
|---|---|---|---|
| L201 | ANIL | VRCE | 1000 |
| L206 | MEHUL | AJNI | 5000 |
| L311 | SUNIL | DHARAMPETH | 3000 |
| L321 | MADHURI | ANDHERI | 2000 |
| L371 | PRAMOD | POWAI | 8000 |
| L481 | KRANTI | NEHRU | 3000 |
| NULL | NULL | NULL | NULL |

### 3. list all data from customer

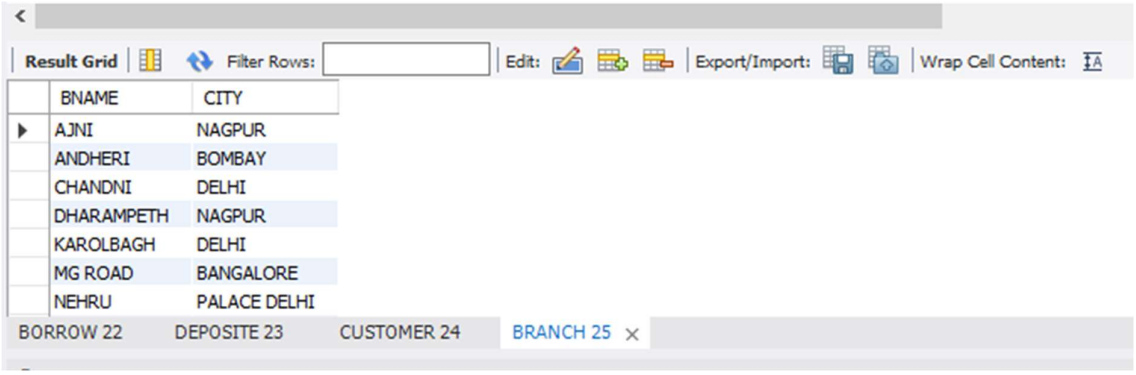
select * from customer;



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

## 4. list all data from branch

select * from branch;



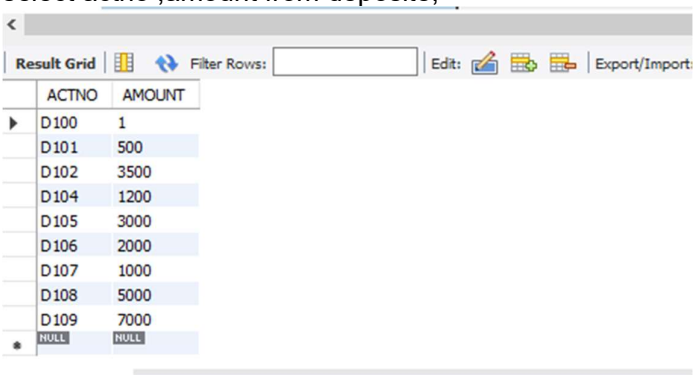
Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

| BNAME | CITY |
|---|---|
| AJNI | NAGPUR |
| ANDHERI | BOMBAY |
| CHANDNI | DELHI |
| DHARAMPETH | NAGPUR |
| KAROLBAGH | DELHI |
| MG ROAD | BANGALORE |
| NEHRU | PALACE DELHI |

BORROW 22 DEPOSITE 23 CUSTOMER 24 BRANCH 25

## 5. give account no and amount of deposite

select actno ,amount from deposite;

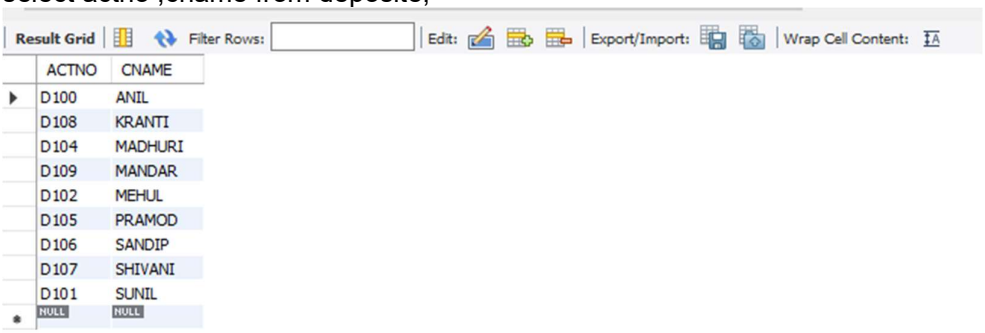


Result Grid | Filter Rows: | Edit: | Export/Import:

| ACTNO | AMOUNT |
|---|---|
| D100 | 1 |
| D101 | 500 |
| D102 | 3500 |
| D104 | 1200 |
| D105 | 3000 |
| D106 | 2000 |
| D107 | 1000 |
| D108 | 5000 |
| D109 | 7000 |
| NULL | NULL |

## 6. give customer name and account no of depositor

select actno ,cname from deposite;

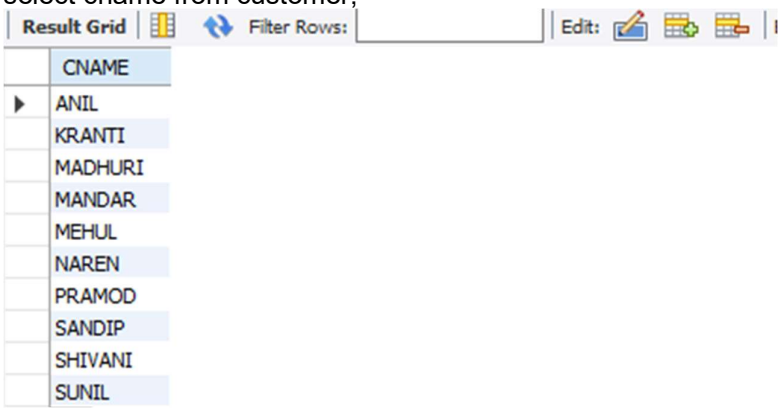


Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

| ACTNO | CNAME |
|---|---|
| D100 | ANIL |
| D108 | KRANTI |
| D104 | MADHURI |
| D109 | MANDAR |
| D102 | MEHUL |
| D105 | PRAMOD |
| D106 | SANDIP |
| D107 | SHIVANI |
| D101 | SUNIL |
| NULL | NULL |

## 7. give name of customers

select cname from customer;




## 8. give name of branches

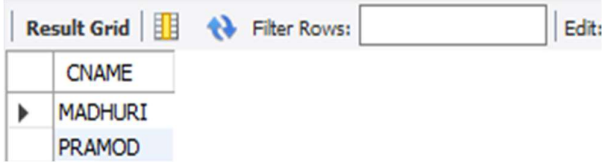
select bname from branch;


## 9. give name of borrows

select cname from borrow;

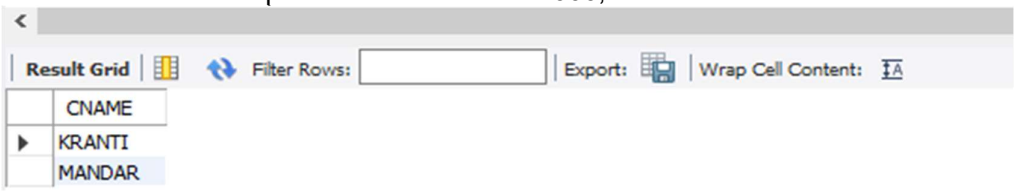

## 10. give names of customer living in city nagpur

select cname from customer where city="nagpur";



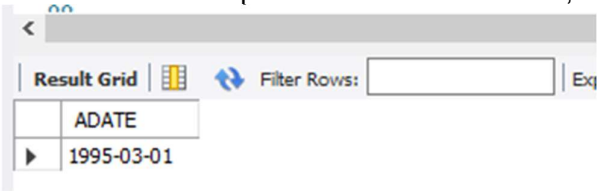
### 11. give names of depositors having amount greater than 4000

select cname from deposite where amount>4000;



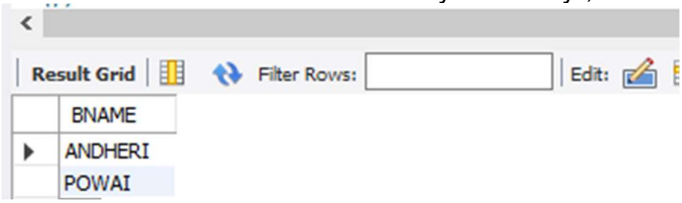

### 12. give account date of anil

select adate from deposite where cname="anil";



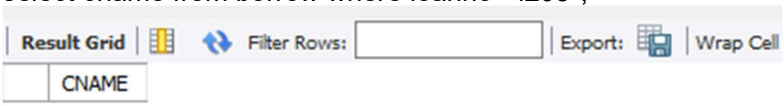

### 13. give name of all branches located in bombay

select bname from branch where city="bombay";



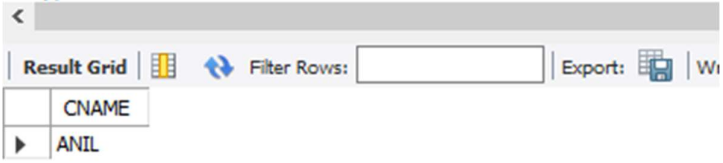

### 14. give name of borrower having loan number l205

select cname from borrow where loanno="l205";




### 15. give names of depositors having account at vrce

select cname from deposite where bname="vrce";

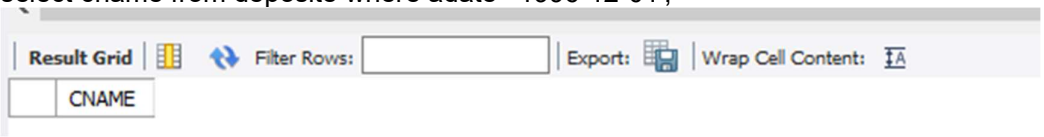


### 16. give names of all branched located in city delhi

select bname from branch where city="delhi";

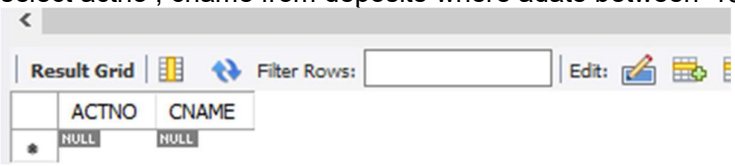

### 17. give name of the customers who opened account date '1-12-96'

select cname from deposite where adate='1996-12-01';




### 18. give account no and deposit amount of customers having account opened between dates '1-12-96' and '1-5-96'

select actno , cname from deposite where adate between '1996-12-01' and '1996-05-01';




### 19. give name of the city where branch karolbagh is located

select city from branch where bname="karolbagh";


### 20. give details of customer anil

select * from customer where cname="anil";

## Result

The program was executed and the result was successfully obtained. Thus, CO1 was obtained.

## Experiment No.:4

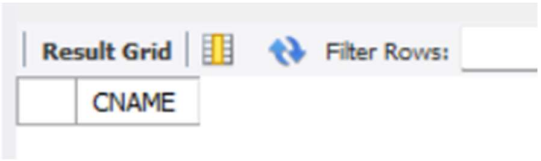
**Aim:** To familiarize with set operations.

**CO1:** Design and build a simple relational database system and demonstrate competence with the fundamentals tasks involved with modelling, designing and implementing a database.

## Procedure

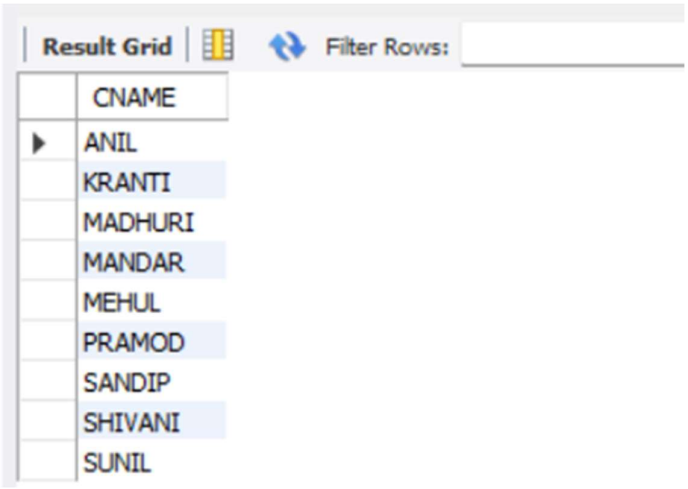
**1. list all the customers who are depositors but not borrowers.**

select cname from deposite where not exists (select cname from borrow);



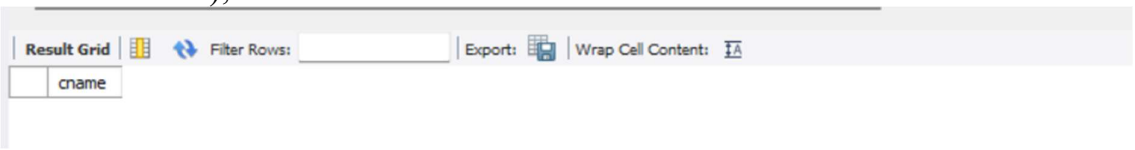

**2. list all the customers who are both depositors and borrowers**

select cname from deposite union (select cname from borrow);




**3. list all the depositors having deposit in all the branches where sunil is having account.**

select cname from deposit where bname in (select bname from deposit where cname="sunil");



**4. list all the customers living in city nagpur and having branch city Bombay or delhi.**

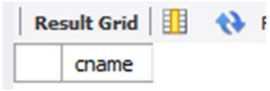
select cname from customer where city="nagpur" and city in(select city from branch where city="bombay" or city="delhi");


**5. list all the depositors living in city nagpur**

select cname from deposit where cname in(select cname from customer where city="nagpur");


**6. list all the depositors living in the city nagpur and having branch in city Bombay**

select cname from deposit where cname in(select cname from customer where city="nagpur" and bname in(select bname from branch where city="bombay"));




**7. list the branch cities of anil and sunil**

select city from branch where bname in(select bname from deposit where cname="anil" and cname="sunil");


**8. list the customers having deposit greater than 1000 and loan less than 10000.**

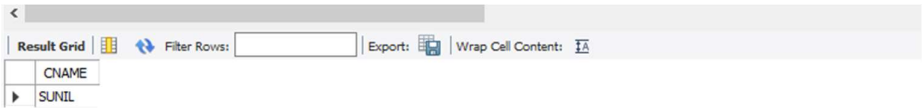
select cname from deposit where amount>1000 and amount in(select amount<10000 from borrow)

### 9. list the cities of depositors having branch vrce.

select city from customer where cname in(select cname from deposit where bname="vrce");

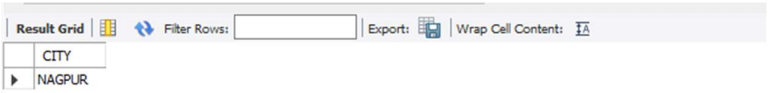

### 10. list the depositors having amount less than 1000 and living in the same city as anil

select d1.cname from deposite d1, customer c1 where amount<1000 and c1.city=(c1.cname='anil');



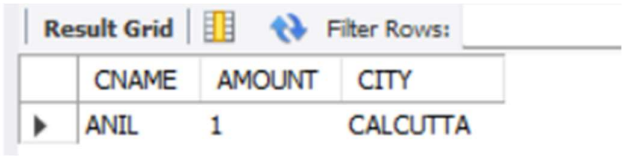

### 11. list all the cities where branches of anil and sunil are locate

select b1.city from branch b1 where b1.bname in (select d1.bname from deposite d1 where d1.cname in ('anil','sunil'));




### 12. list the amount for the depositors living in the city where anil is living

select distinct(d1.cname),d1.amount ,c1.city from deposite d1,customer c1, branch b1 where d1.cname=c1.cname and c1.city in(select c2.city from customer c2 where c2.cname='anil');




## Result

The program was executed and the result was successfully obtained. Thus, CO1 was obtained.

## Experiment No.: 5

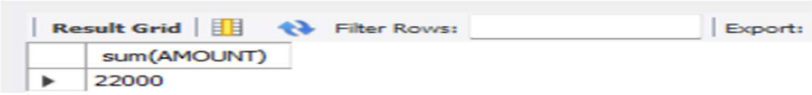
**Aim:** To familiarize with aggregate functions.

**CO1:** Design and build a simple relational database system and demonstrate competence with the fundamentals tasks involved with modelling, designing and implementing a database.

## Procedure

### 1.list total loan

select sum(amount) from borrow;

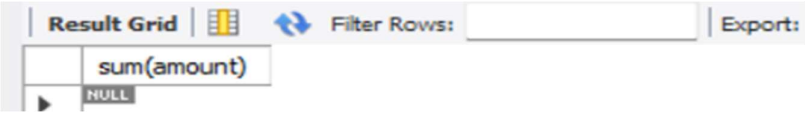



### 2.list total deposit

select sum(amount) from deposite;

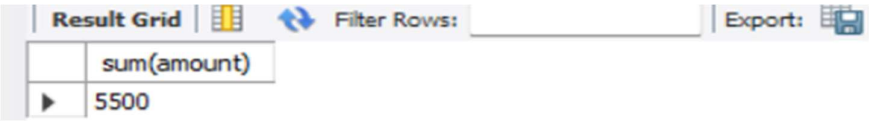

### 3.list total loan taken from karolbagh branch

select sum(amount) from borrow where bname="karolbagh";



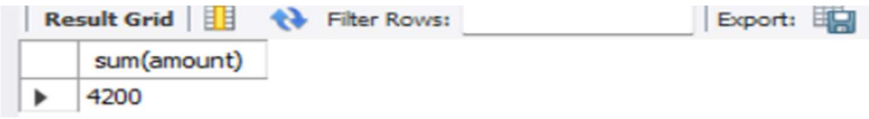

### 4.list total deposit of customers having account date later than 1-jan-96

select sum(amount) from deposite where adate>"1996-01-01";



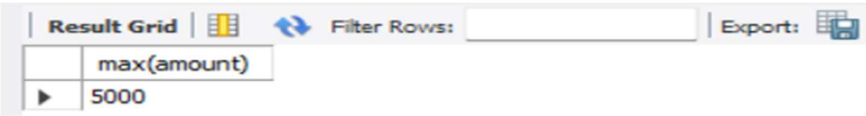

### 5.list total deposit of customers living in city nagpur

select sum(amount) from deposite where cname in(select cname from customer where city="nagpur");



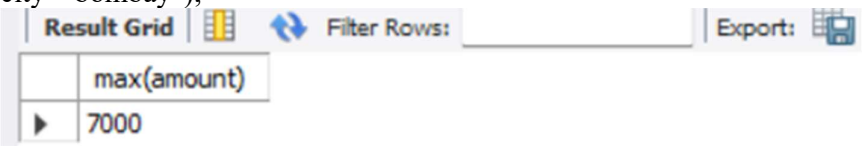
## 6.list maximum deposit of customer living in bombay

select max(amount) from deposite where cname in(select cname from customer where city="bombay");



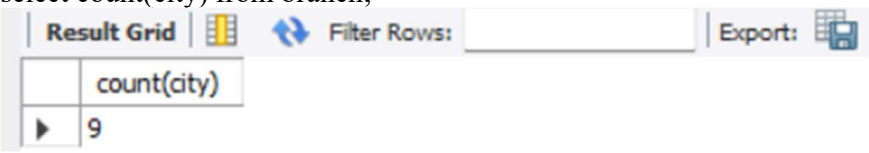

## 7.list total deposit of customer having branch in bombay

select max(amount) from deposite where bname in(select bname from branch where city="bombay");



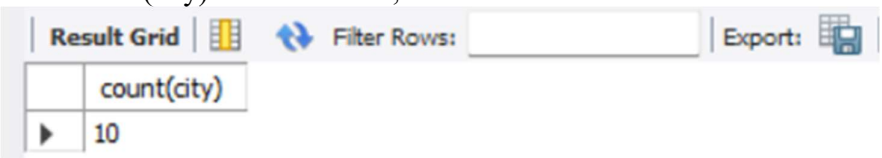

## 8.count total number of branch cities

select count(city) from branch;



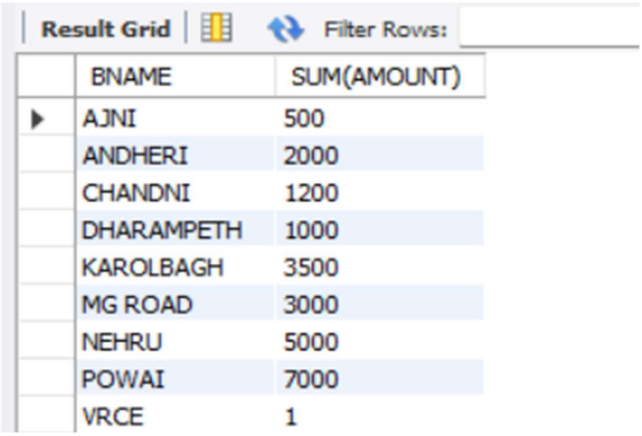

## 9.count total number of customers cities

select count(city) from customer;




## 10.give branch names and branch wise deposit

select bname , sum(amount) from deposite group by bname;



## 11.give city wise name and branch wise deposit

select c1.city , sum(d1.amount) from customer c1 , deposite d1 where d1.cname = c1.cname group by c1.city;

Result Grid | Filter Rows:

| CITY | SUM(D1.AMOUNT) |
|---|---|
| CALCUTTA | 1 |
| DELHI | 500 |
| BARODA | 3500 |
| NAGPUR | 4200 |
| SURAT | 2000 |
| BOMBAY | 6000 |
| PATNA | 7000 |

## 12.give the branch wise loan of customer living in Nagpur

select bname , sum(amount) from borrow,customer where city='nagpur' group by bname;

Result Grid | Filter Rows:

| BNAME | SUM(AMOUNT) |
|---|---|
| VRCE | 2000 |
| AJNI | 10000 |
| DHARAMPETH | 6000 |
| ANDHERI | 4000 |
| POWAI | 16000 |
| NEHRU | 6000 |

## 13.count total number of customers

select count(cname) from customer;

Result Grid | Filter Rows: | Export:

| count(cname) |
|---|
| 10 |

## 14.count total number of depositors branch wise

select bname, count(*) from deposite, customer where deposite.cname=customer.cname group by bname;

Result Grid | Filter Rows:

| BName | count(*) |
|---|---|
| VRCE | 1 |
| AJNI | 1 |
| KAROLBAGH | 1 |
| CHANDNI | 1 |
| MG ROAD | 1 |
| ANDHERI | 1 |
| DHARAMPETH | 1 |
| NEHRU | 1 |
| POWAI | 1 |

## 15.give maximum loan from branch vrce

select max(amount) from borrow where bname ='vrce';

Result Grid | Filter

| MAX(AMOUNT) |
|---|
| 1000 |

## 16.give the number of customers who are depositors as well as borrowers

Limit to 1000 rows

```
1   select count(customer.CNAME) from customer where customer.CNAME IN (select deposit.cname from deposit)
2   and customer.CNAME IN (select borrow.cname from borrow);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| count(customer.CNAME) |
|---|
| 6 |

## Result

The program was executed and the result was successfully obtained. Thus, CO1 was obtained.

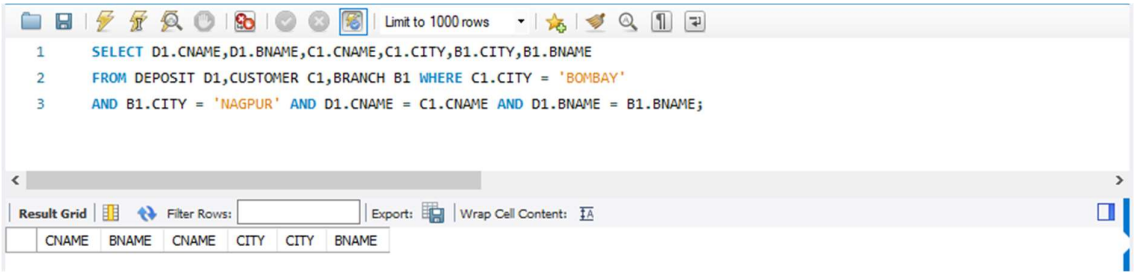
## Experiment No.:6

**Aim:** To familiarize with Join or Cartesian Product.

**CO1:** Design and build a simple relational database system and demonstrate competence with the fundamentals tasks involved with modelling, designing and implementing a database.

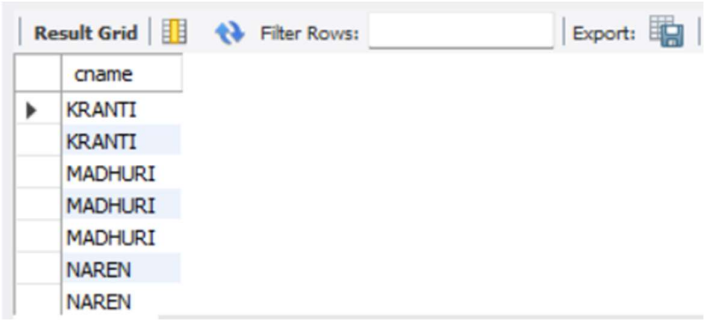
## Procedure

### 1.Give name of customers having living city bombay and branch city nagpur




### 2. give names of customers having the same living city as their branch city

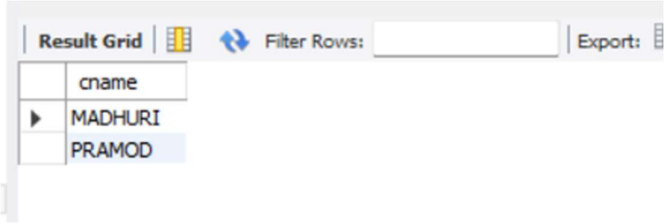
select cname from customer,branch where customer.city=branch.city;




### 3. give names of customers who are borrowers as well as depositors and having city

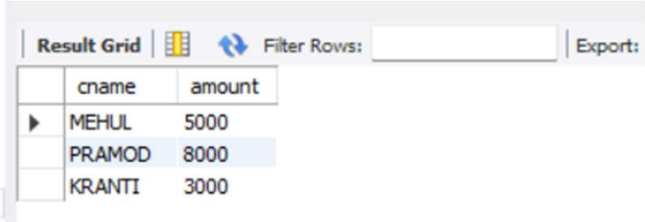
### nagpur.

select cname from customer where city="nagpur" and cname in(select cname from borrow) and cname in(select cname from deposite);



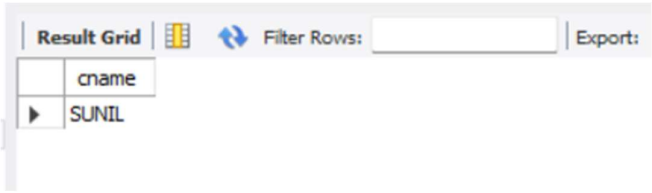
## 4. give names of borrowers having deposit amount greater than 1000 and loan amount greater than 2000.

select cname,amount from borrow where amount > 2000 and cname in(select cname from deposite where amount >1000);



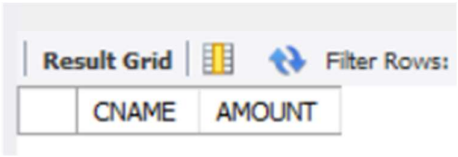

## 5. give names of depositors having the same branch as the branch of sunil

select cname from deposit where bname in(select bname from branch where cname="sunil");



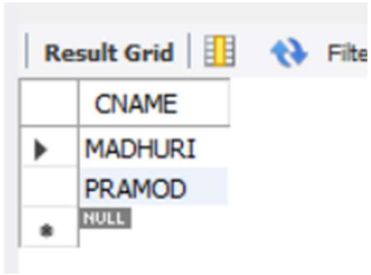

## 6.give names of borrowers having loan amount greater than the loan amount of pramod

select br1.cname,br1.amount from borrow br1 where br1.amount > all (select br2.amount from borrow br2 where br2.cname = 'pramod');



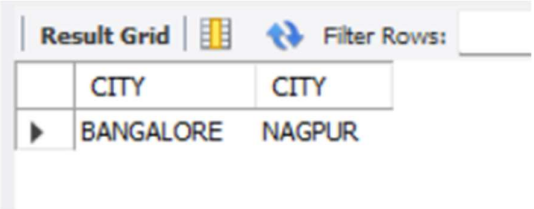

## 7.give the name of the customer living in the city where branch of depositor sunil is located.

select c.cname from customer c where c.city in (select b.city from branch b where b.bname in (select d.bname from deposite d where d.cname='sunil'));



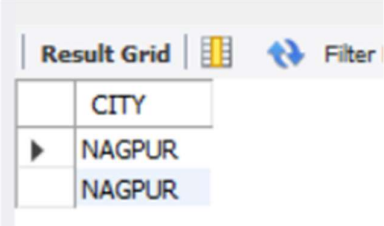
## 8. give branch city and living city of pramod

select b.city,c.city from branch b,customer c where cname ="pramod";



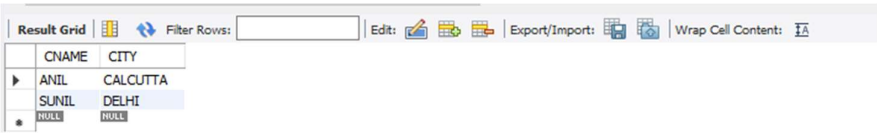

## 9.Give branch city of sunil and branch city of anil

select b1.city from deposite d1, branch b1 where d1.bname = b1.bname and d1.cname in ('sunil' ,'anil');




## 10.Give the living city of anil and the living city of sunil

select c1.cname, c1.city from customer c1 where c1.cname ='anil' or c1.cname = 'sunil';




## Result

The program was executed and the result was successfully obtained. Thus, CO1 was obtained.

## Experiment No.: 7

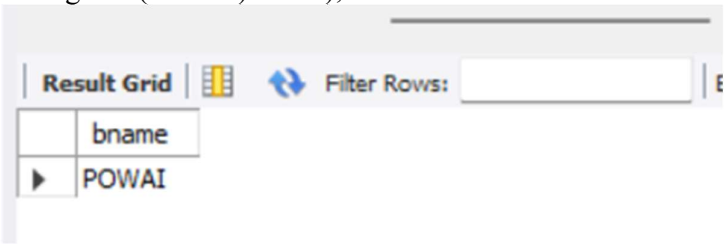
**Aim:** To familiarize with Group by and Having Clauses.

**CO1:** Design and build a simple relational database system and demonstrate competence with the fundamentals tasks involved with modelling, designing and implementing a database.

## Procedure

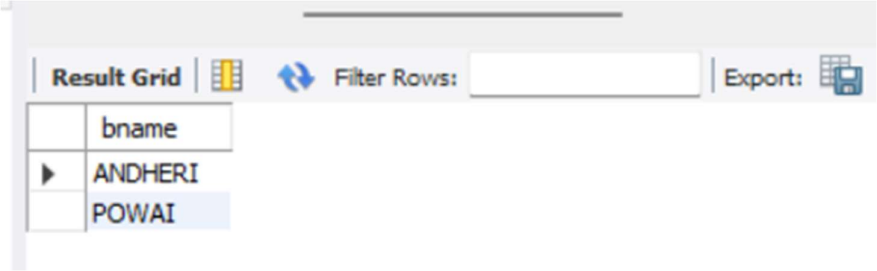
### 1.list the branches having sum of deposit more than 5000.

select bname from branch where bname in(select bname from deposite group by bname having sum(amount)>5000);



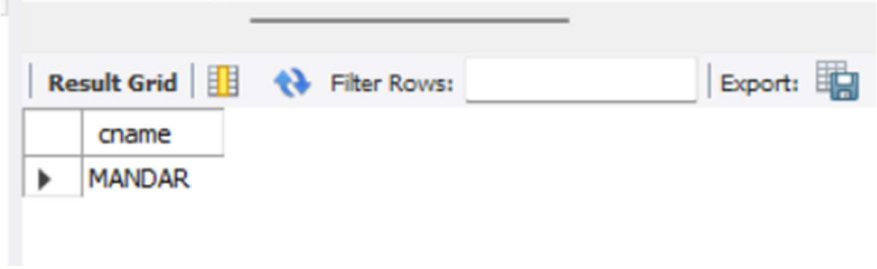

### 2.list the branches having sum of deposit more than 500 and located in city bombay

select bname from branch where bname in(select bname from deposite group by bname having sum(amount)>500 and city="bombay");



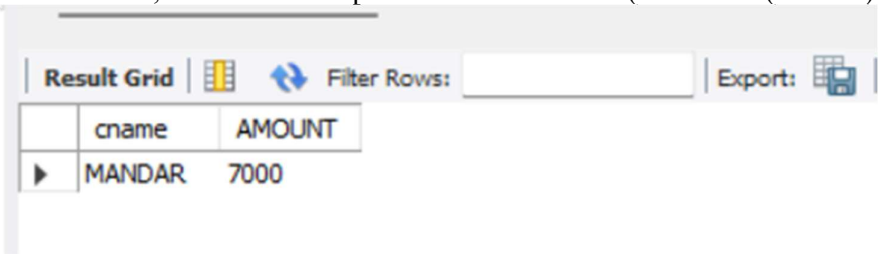

### 3.list the names of customers having deposited in the branches where the average deposit is more than 5000.

select cname from deposite where amount=(select avg(amount) from deposite group by bname having avg(amount)>5000);

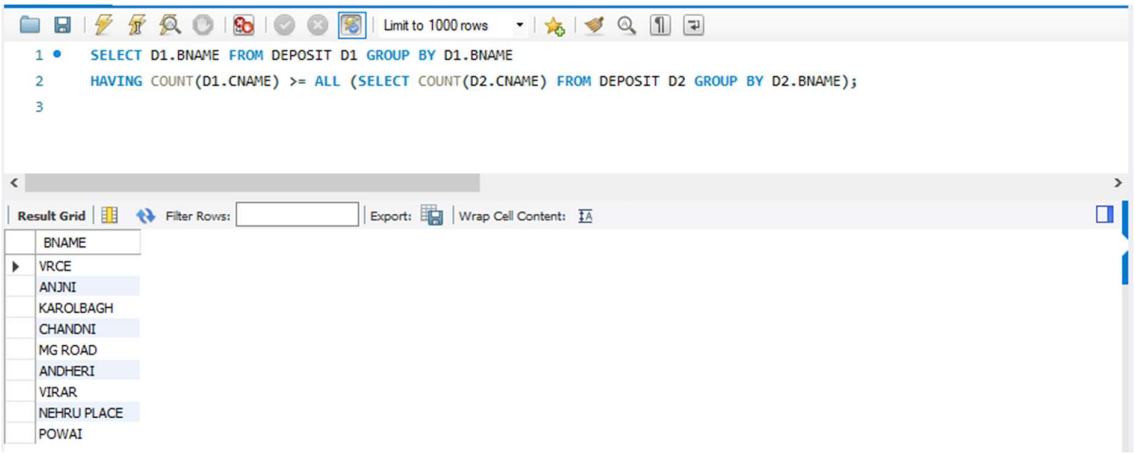


## 4.list the names of customers having maximum deposit

select cname,amount from deposite where amount =(select max(amount) from deposite);

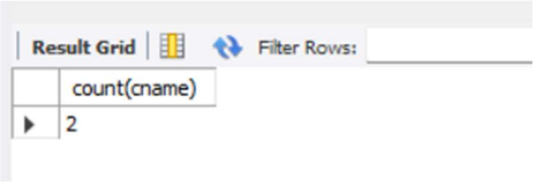



## 5.list the name of the branch having the highest number of depositors



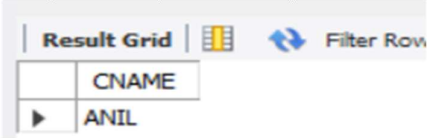

## 6.count the number of depositors living in nagpur.

select count(cname) from deposite where cname in( select cname from customer where city="nagpur");



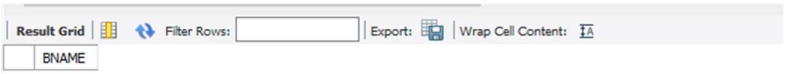

## 7.give names of customers in vrce branch having more deposit than any other customer in same branch.

select cname from deposite where bname='vrce' and amount=(select max(amount) from deposite where bname='vrce');



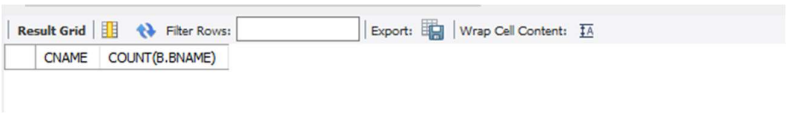
## 8.give the names of branch where number of depositors is more than 5

select bname from deposit group by bname having count(bname)>5;



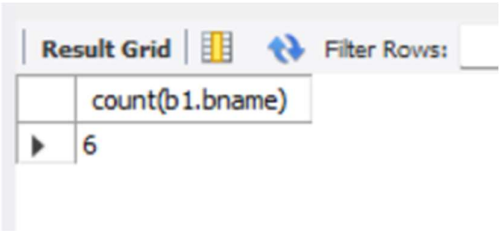

## 9.give the names of cities in which the maximum number of branches are located

select c.cname ,count(b.bname) from customer c inner join branch b on c.cname=b.bname group by c.cname order by count(b.bname) desc;




## 10.count the number of customers living in the city where branch is located

select count(b1.bname) from deposite d1 , borrow b1 , customer c1 where c1.cname=d1.cname and d1.cname=b1.cname and c1.city in (select city from customer);




## Result

The program was executed and the result was successfully obtained. Thus, CO1 was obtained.

## Experiment No.: 8

**Aim:** To familiarize with trigger functions.

**CO2:** Apply PL/SQL for processing databases

## Procedure

Step 1: Start

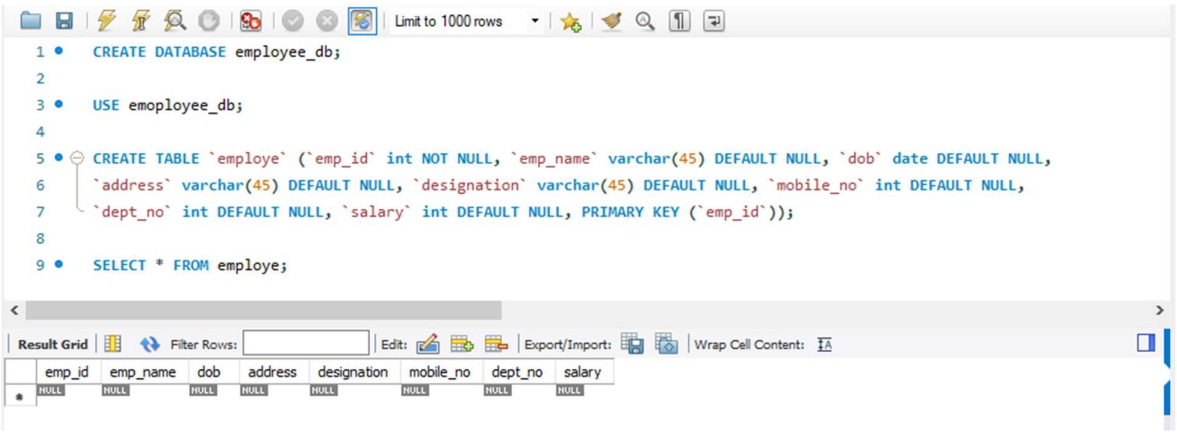
Step 2: Initialize the trigger.

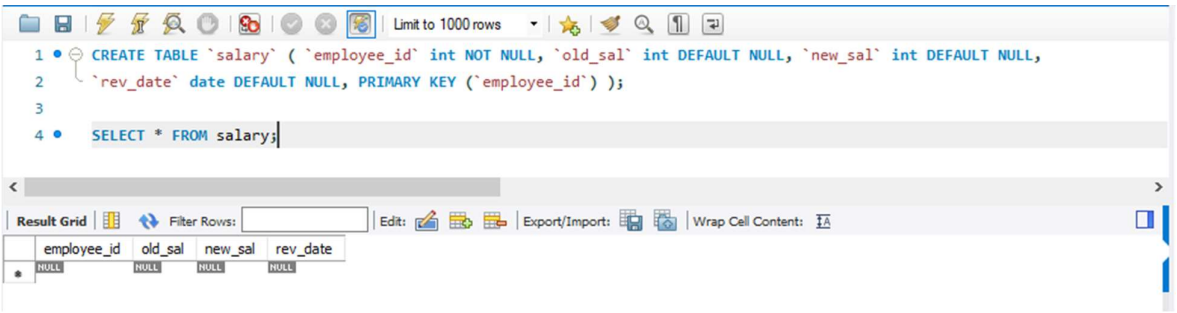
Step 3: On update the trigger has to be executed.

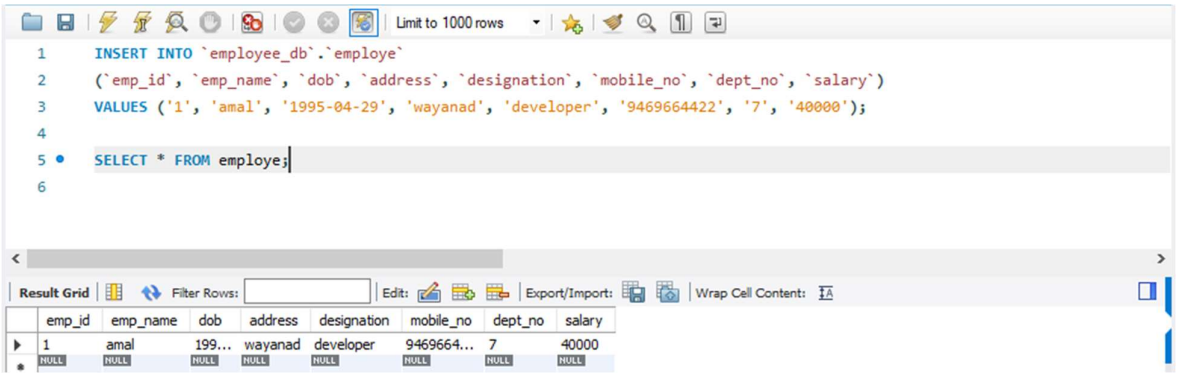
Step 4: Execute the trigger procedure after updating

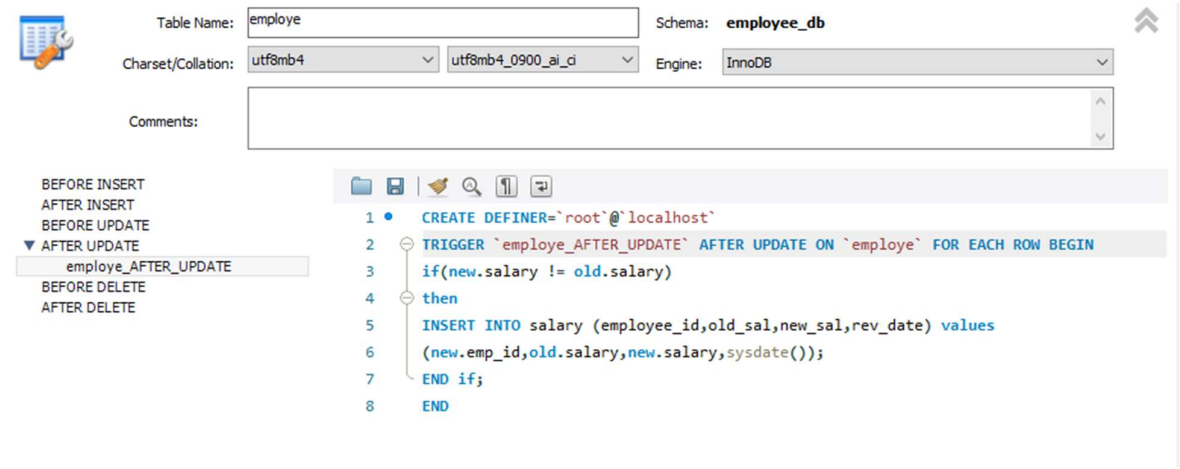
Step 5: Carry out the operation on the table to check for trigger execution.

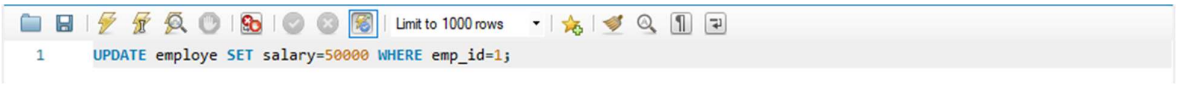
Step 6: Stop

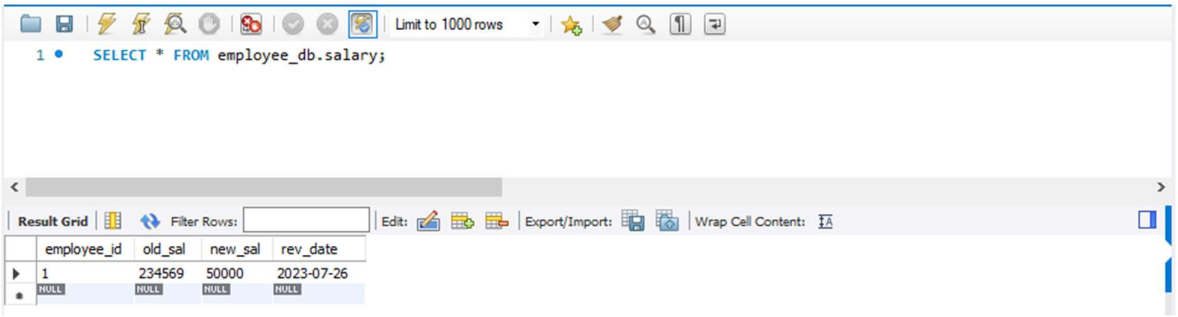





## Result

The program was executed and the result was successfully obtained. Thus, CO2 was obtained.

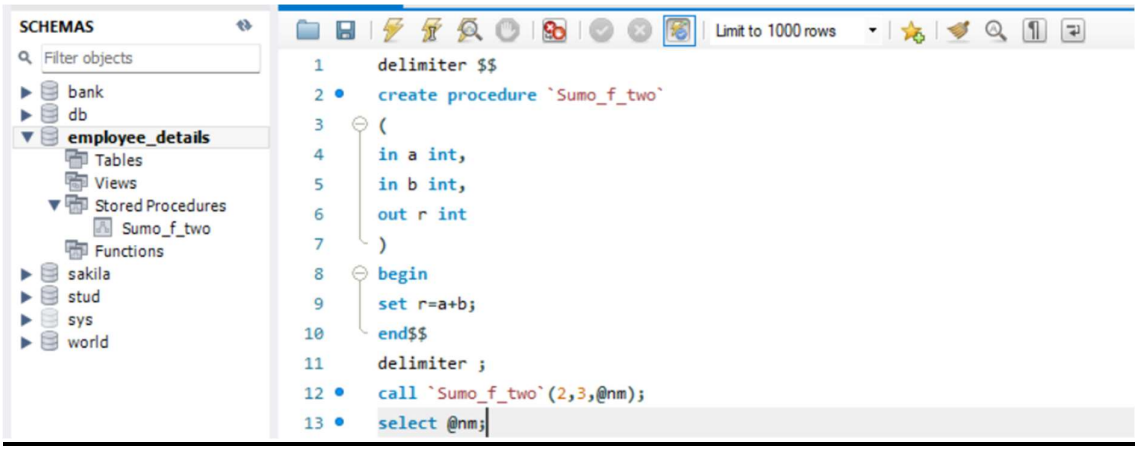
## Experiment No.: 9

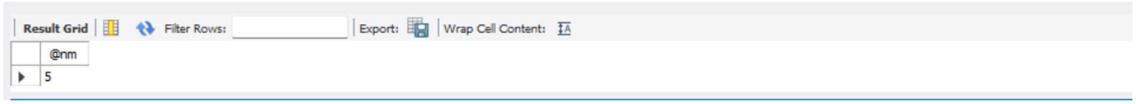
**Aim:** To familiarize with Stored functions and Procedure.

**CO2:** Apply PL/SQL for processing databases

## Procedure

### Create Stored Procedure Sumo_f_two




## Result

The program was executed and the result was successfully obtained. Thus, CO2 was obtained.

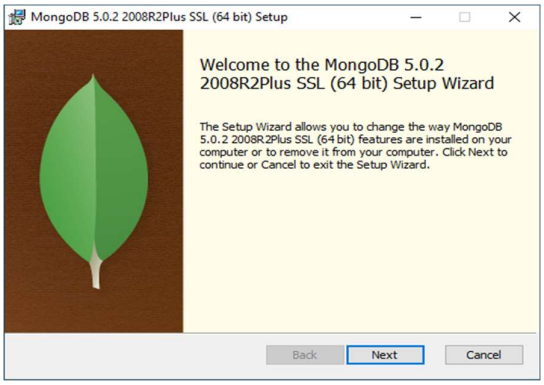
## Experiment No.: 10

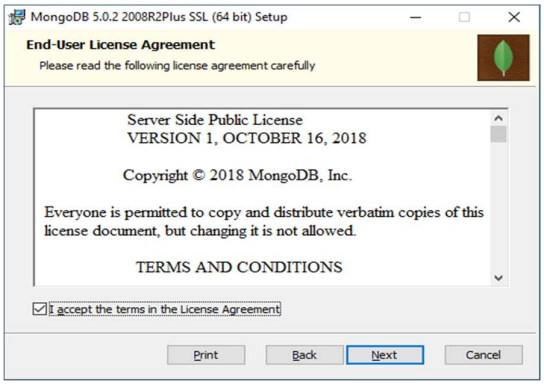
**Aim:** Understand the installation and configuration of NoSQL Databases: MongoDB.

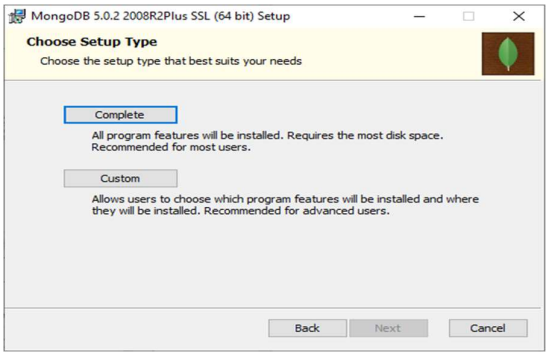
**CO3:** Comparison between relational and non-relational (NoSQL) databases and the configuration of NoSQL Databases.

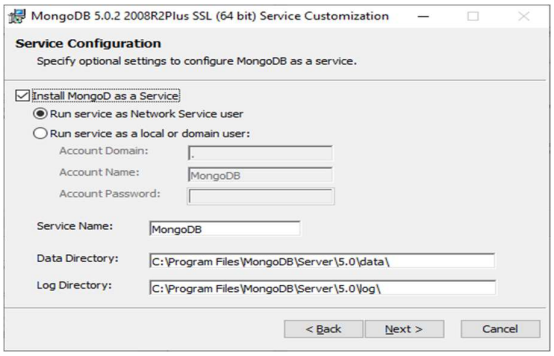
## Procedure

1. Setup Wizard

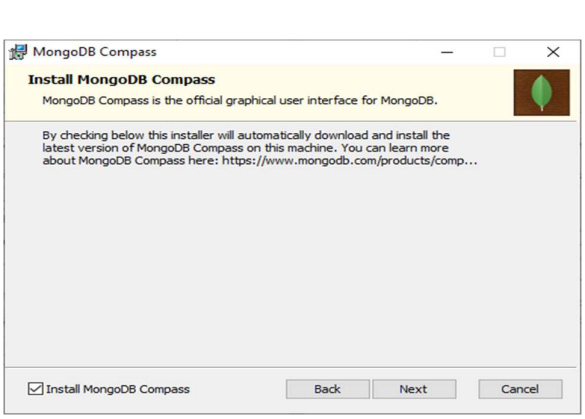


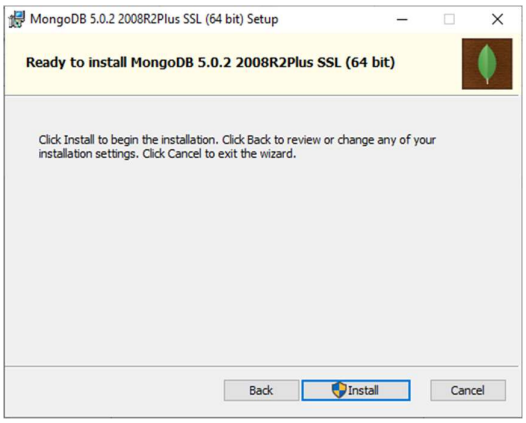



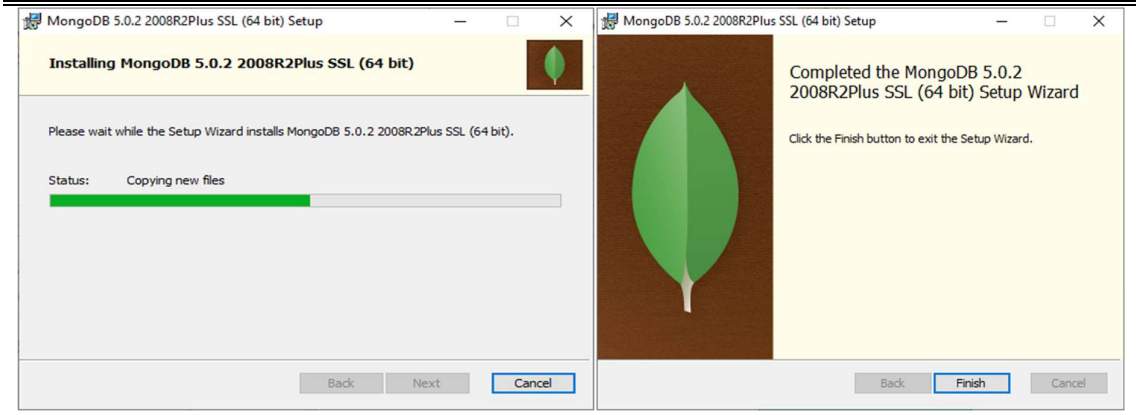



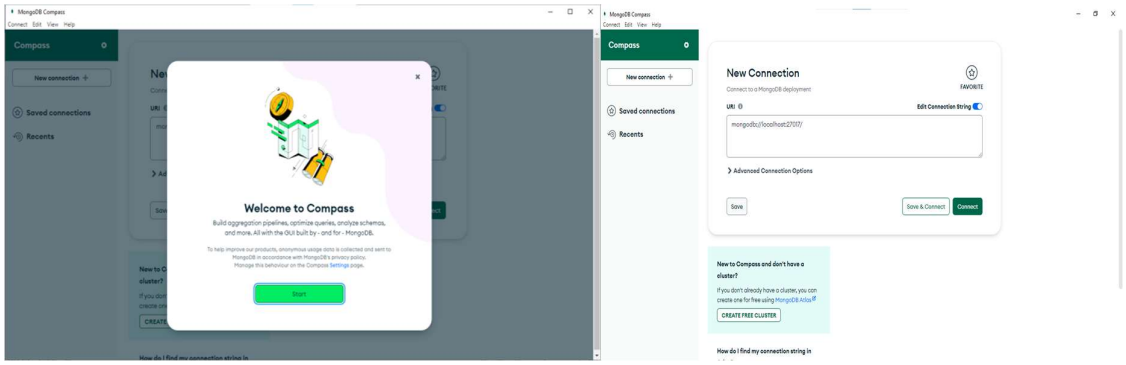

2. Installing MongoDB Compass

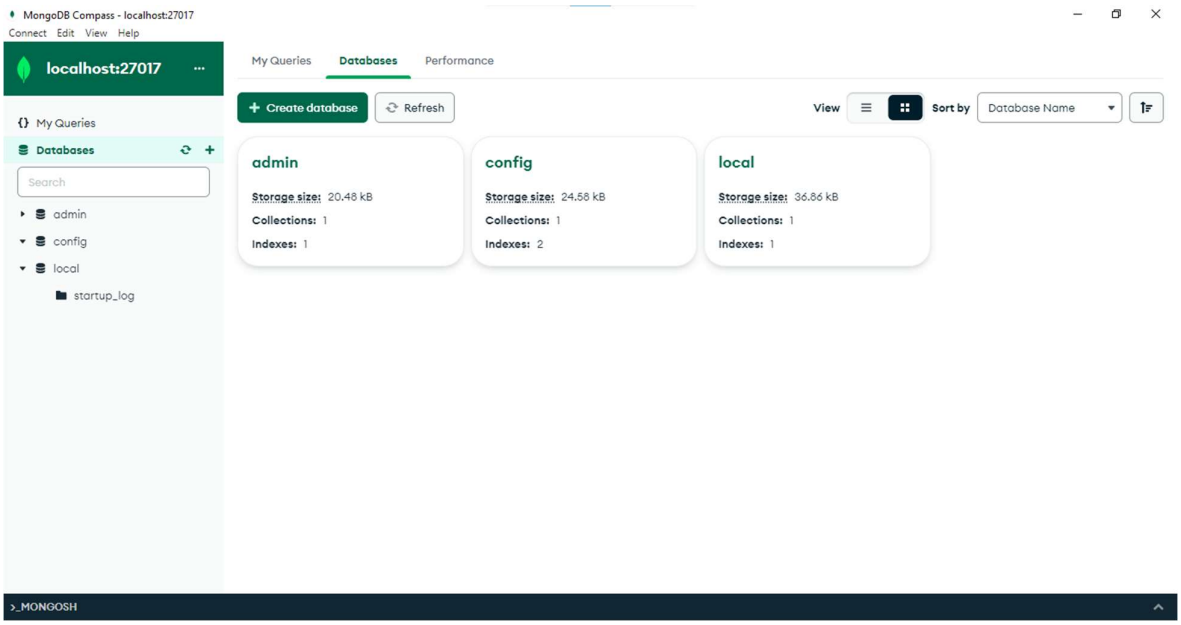


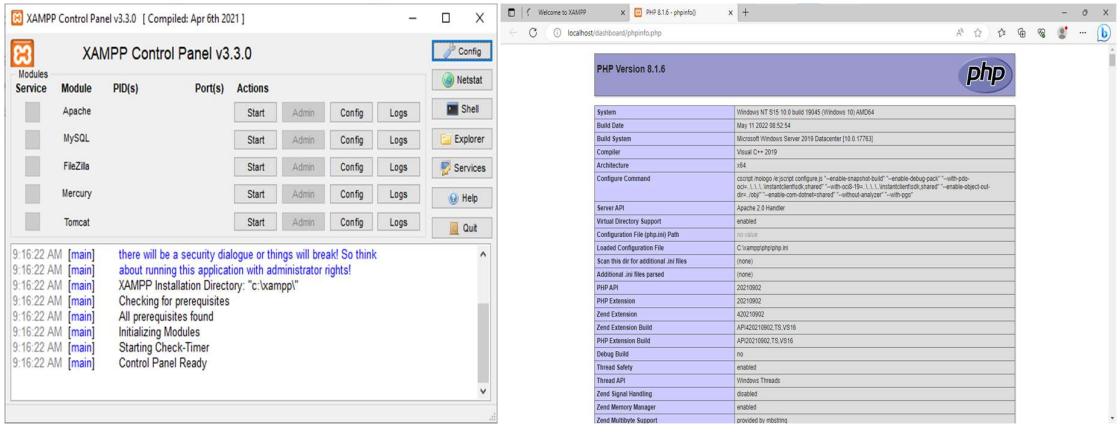



3. MongoDB Compass Home Page

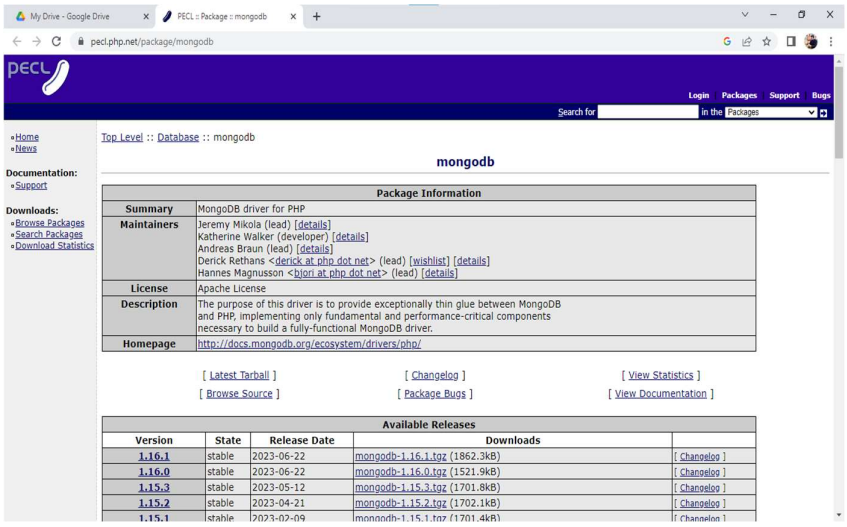


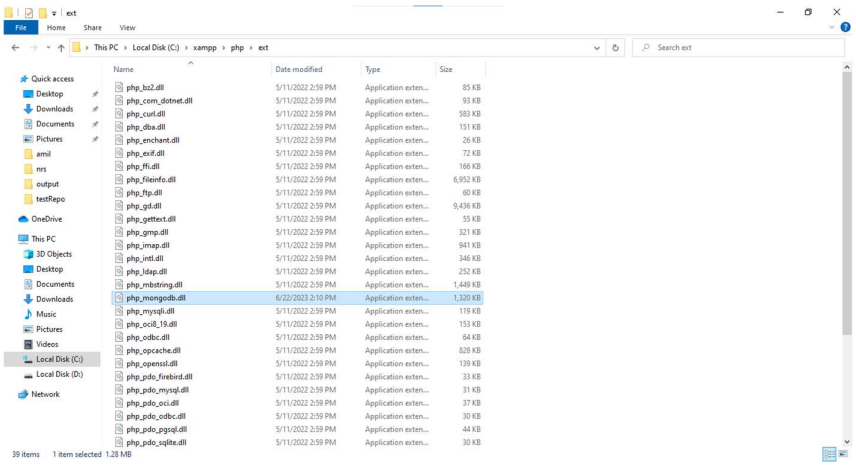

4. Configuring MongoDB to Connect to PHP

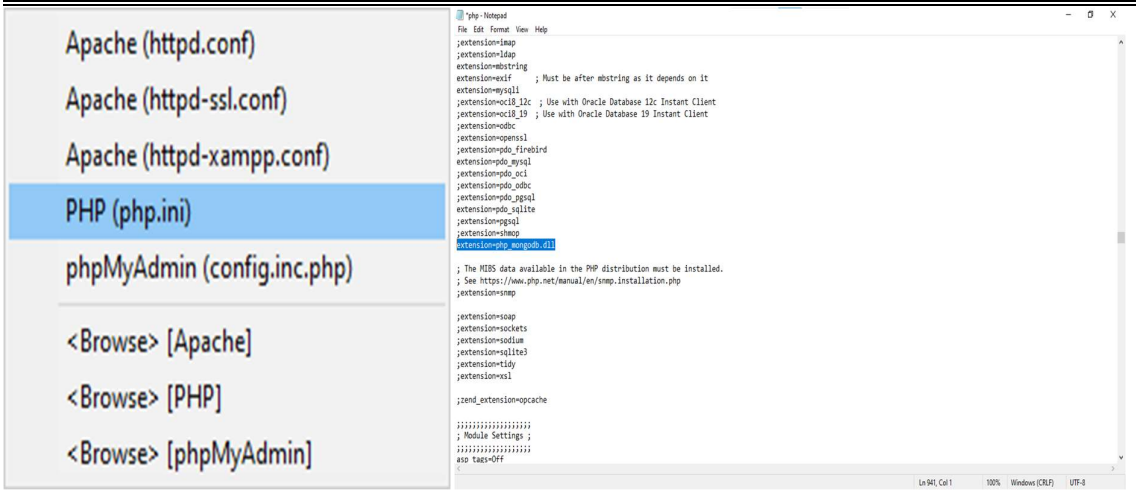


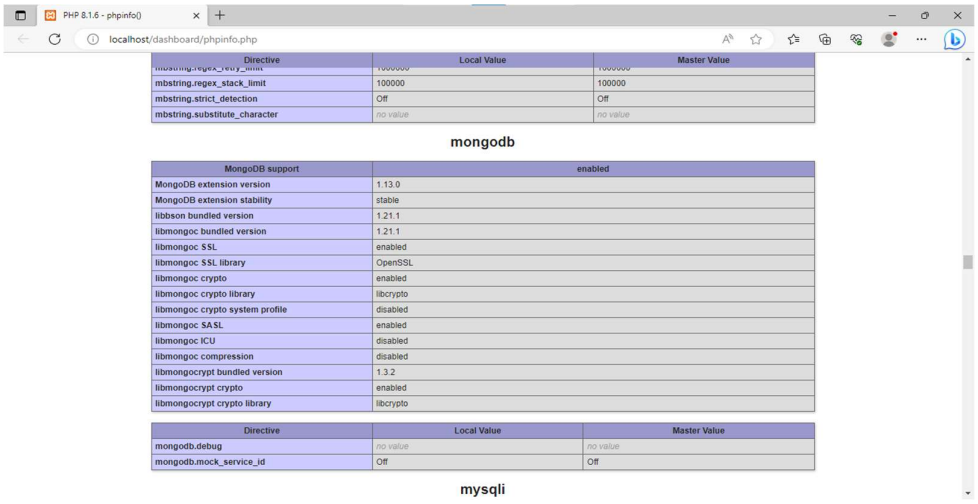

5. Installing MongoDB Package for PHP

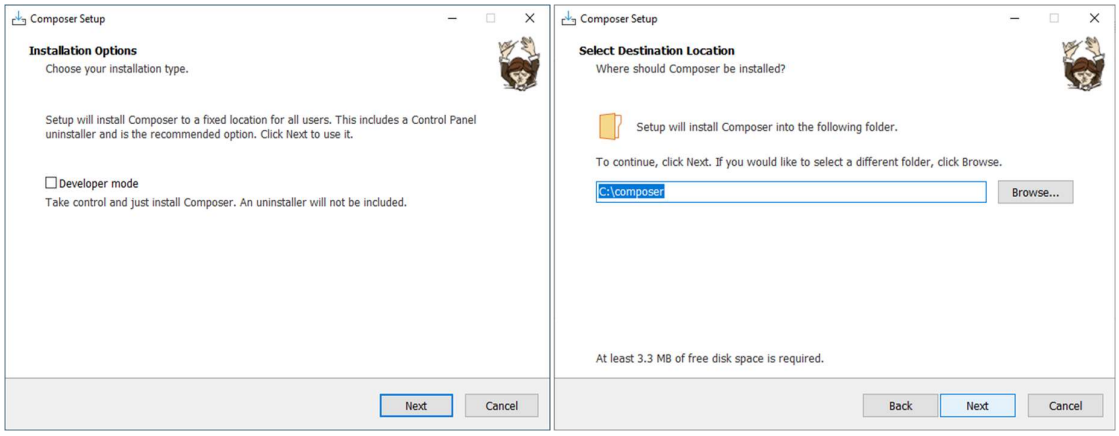


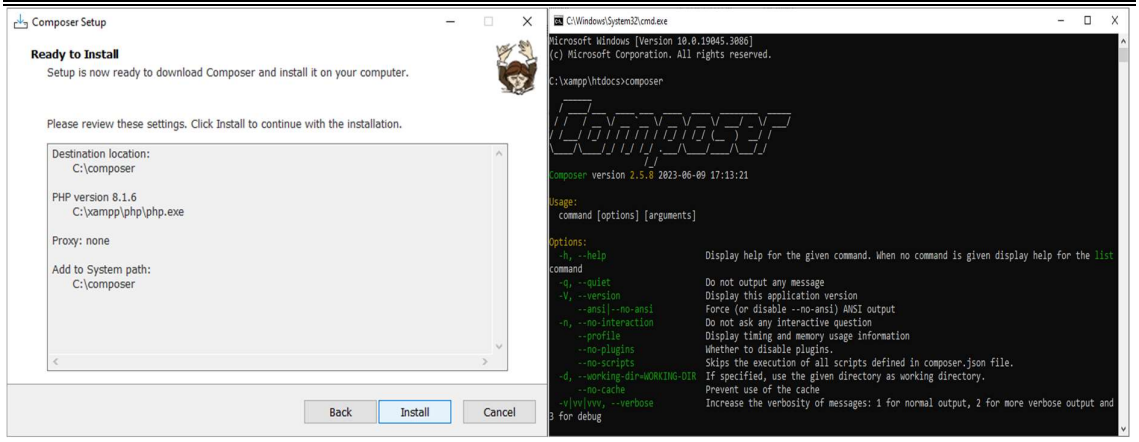

6. Configuring Apache

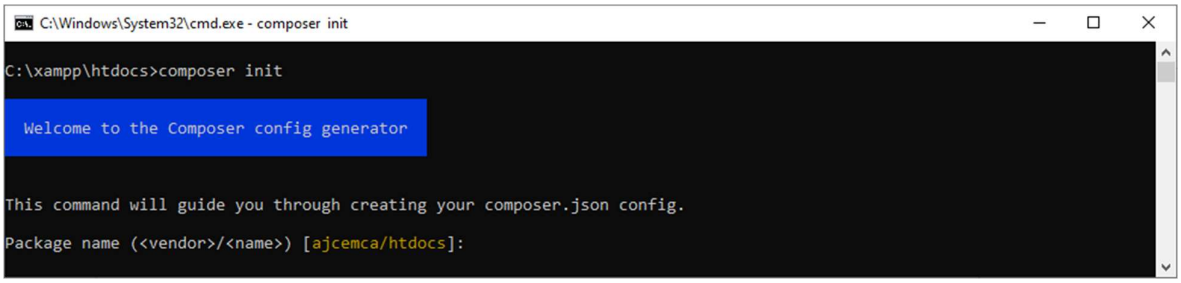


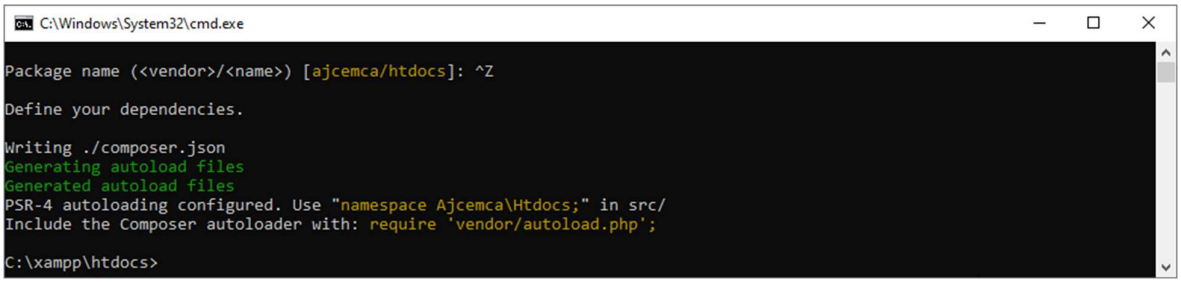

Apache (httpd.conf)
Apache (httpd-ssl.conf)
Apache (httpd-xampp.conf)
PHP (php.ini)
phpMyAdmin (config.inc.php)
<Browse> [Apache]
<Browse> [PHP]
<Browse> [phpMyAdmin]
*php - Notepad
File Edit Format View Help
;extension=imap
;extension=ldap
extension=mbstring
extension=exif ; Must be after mbstring as it depends on it
extension=mysqli
;extension=oci8_12c ; Use with Oracle Database 12c Instant Client
;extension=oci8_19 ; Use with Oracle Database 19 Instant Client
;extension=odbc
;extension=openssl
;extension=pdo_firebird
extension=pdo_mysql
;extension=pdo_oci
;extension=pdo_odbc
;extension=pdo_pgsql
extension=pdo_sqlite
;extension=pgsql
;extension=shmop
extension=php_mongodb.dll
; The MIBS data available in the PHP distribution must be installed.
; See https://www.php.net/manual/en/snmp.installation.php
;extension=snmp
;extension=soap
;extension=sockets
;extension=sodium
;extension=sqlite3
;extension=tidy
;extension=xsl
;zend_extension=opcache
;;;;;;;;;;;;;;;;;;;
; Module Settings ;
;;;;;;;;;;;;;;;;;;;
asp_tags=Off
Ln 941, Col 1
100%
Windows (CRLF)
UTF-8

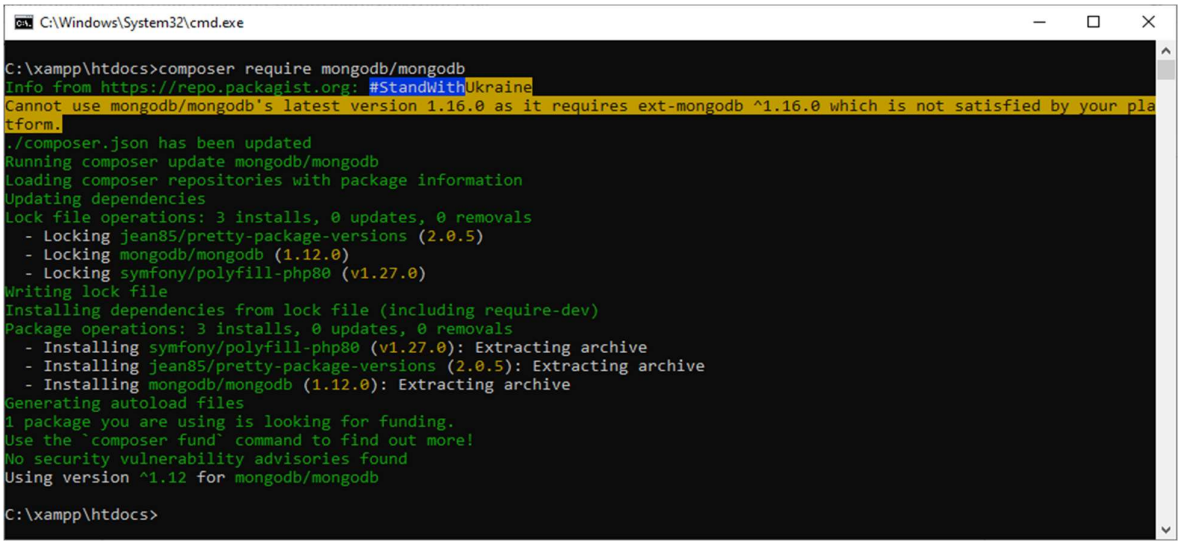

PHP 8.1.6 - phpinfo()
localhost/dashboard/phpinfo.php
Directive
Local Value
Master Value
mbstring.regex_stack_limit
100000
100000
mbstring.strict_detection
Off
Off
mbstring.substitute_character
no value
no value
mongodb
MongoDB support
enabled
MongoDB extension version
1.13.0
MongoDB extension stability
stable
libbson bundled version
1.21.1
libmongoc bundled version
1.21.1
libmongoc SSL
enabled
libmongoc SSL library
OpenSSL
libmongoc crypto
enabled
libmongoc crypto library
libcrypto
libmongoc crypto system profile
disabled
libmongoc SASL
enabled
libmongoc ICU
disabled
libmongoc compression
disabled
libmongocrypt bundled version
1.3.2
libmongocrypt crypto
enabled
libmongocrypt crypto library
libcrypto
Directive
Local Value
Master Value
mongodb.debug
no value
no value
mongodb.mock_service_id
Off
Off
mysqli


Composer Setup
Installation Options
Choose your installation type.
Setup will install Composer to a fixed location for all users. This includes a Control Panel uninstaller and is the recommended option. Click Next to use it.
Developer mode
Take control and just install Composer. An uninstaller will not be included.
Next
Cancel
Composer Setup
Select Destination Location
Where should Composer be installed?
Setup will install Composer into the following folder.
To continue, click Next. If you would like to select a different folder, click Browse.
C:\composer
Browse...
At least 3.3 MB of free disk space is required.
Back
Next
Cancel




## Result

The program was executed and the result was successfully obtained. Thus, CO3 was obtained.

## Experiment No.: 11

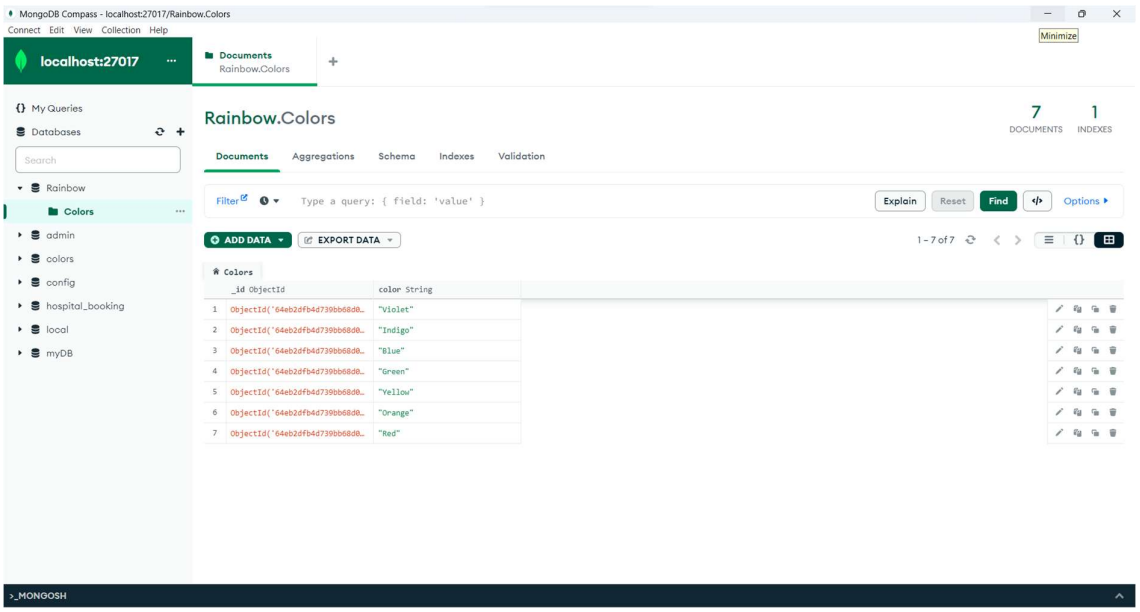
**Aim:** Create a database and use the insertMany method to insert the colors of the rainbow into the database.

**CO4:** Apply CRUD operations and retrieve data in a NoSQL environment.

## Procedure

```
<?php
require '../vendor/autoload.php';
$conn = new MongoDB\Client("mongodb://localhost:27017");
echo "Connection Successful.";
echo "<br>";
$db = $conn -> Rainbow;
echo " Rainbow database created.";
echo "<br>";
$collection = $db -> createCollection("Colors");
echo "Collection Colors created";
echo "<br>";
$collection = $db -> Colors;
echo "<br>";
$c1 = array('color' => 'Violet');
$c2 = array('color' => 'Indigo');
$c3 = array('color' => 'Blue');
$c4 = array('color' => 'Green');
$c5 = array('color' => 'Yellow');
$c6 = array('color' => 'Orange');
$c7 = array('color' => 'Red');
$collection -> insertMany([$c1, $c2, $c3, $c4, $c5, $c6, $c7]);
echo "Colors Inserted ";
echo "<br>";
?>
```

## Output Screenshot


## Result

The program was executed and the result was successfully obtained. Thus, CO4 was obtained.

## Experiment No.: 12

**Aim:** Implementing CRUD operations using PHP-MongoDB.

**CO4:** Apply CRUD operations and retrieve data in a NoSQL environment.

## Procedure

**connection.php**

```
<?php
require '../vendor/autoload.php';
$con=new MongoDB\Client("mongodb://localhost:27017");
//echo $con;
$db=$con->admin;
$collection=$db->Profile;
?>
```

**register.php**

```
<html>
<head><title>Registration form</title>
</head>
<body><table border="1" align="center">
<tr><td>
<h1>Registrarion Form</h1>
<form action="insert.php" method="post">
Name:<input type="text" name="name" required><br><br>
Email:<input type="text" name="email" required><br><br>
Place :<input type="text" name="place" required><br><br>
<input type="submit" name="submit" class="button" value="Register" >
</form>
</td><tr>
</table>
</body>
</html>
```

**edit.php**

```
<?php
include_once("connection.php");
if(isset($_POST['update'])){
echo $id = $_GET['id'];
$name = $_POST['name'];
$place = $_POST['place'];
$email = $_POST['email'];
$db->Profile->updateOne(['_id' => new MongoDB\BSON\ObjectID($id)],['$set' =>
['name'=> $name, 'place' => $place, 'email'=>$email,]]);
header("Location: select.php");
}?>
<html><head>
<title>Registration Form</title>
</head>
<?php
$id = $_GET['id'];
$result = $db->Profile->findOne(['_id' => new MongoDB\BSON\ObjectID($id)]);
$name = $result['name'];
$place = $result['place'];
$email = $result['email'];
?>
<body>
<h2>Profile Data</h2>
<form action="#" method="post">
<?php $id = $_GET['id'];?>
Name: <input type="text" name="name" value="<?php echo $name; ?>" required><br>
Place:<input type="place" name="place" value="<?php echo $place; ?>" required><br>
Email:<input type="email" name="email" value="<?php echo $email; ?>" required><br>
<input type="submit" name="update" class="button" value="Update">
```

```
</form></body>
</html>
```

**select.php**

```
<center><br><form action=register.php>
<input type="submit" value=register></form>
<form action=select.php>
<input type="submit" value=View>
</form><br>
<?php
include_once("connection.php");
$result=$collection->find();
?>
<html>
<head><title>select.php</title>
</head><body>
<table border="0" >
 <tr>
 <th align="center" bgcolor=green><font color=white size=4><b>SlNo.</th>
 <th align="center" bgcolor=green><font color=white size=4><b>Name</th>
 <th align="center" bgcolor=green><font color=white size=4><b>Email</th>
 <th align="center" bgcolor=green><font color=white size=4><b>Place</th>
 <th align="center" bgcolor=green><font color=white size=4><b>Delete</th>
 <th align="center" bgcolor=green><font color=white size=4><b>Edit</th>
 </tr><?php
 $no=1;
 foreach($result as $res){
 ?> <tr>
 <td><font color=DeepSkyBlue size=4><?php echo $no;?>
 <td><font color=steelblue size=4><?php echo $res['name'];?></td>
 <td><font color=pink size=4><?php echo $res['email'];?></td>
 <td><font color=RosyBrown size=4><?php echo $res['place'];?></td>
```

```
<td><a href="<?php echo "delete.php?id=$res[_id]";?>" onClick="return confirm('Are you sure you want to delete?')">Delete</a></td>
<td><a href="<?php echo "edit.php?id=$res[_id]";?>">Edit</a></td>
<?php
$no++;
} ?>
</tr></table></body></html>
```

**insert.php**

```
<?php
include_once("connection.php");
if(isset($_POST["submit"])){
        $user=array(
        'name'=>$_POST['name'],
        'email'=>$_POST['email'],
        'place'=>$_POST['place']
        );
        $collection->insertOne($user);
        echo "Inserted";
}
header ("location:select.php");
?>
```

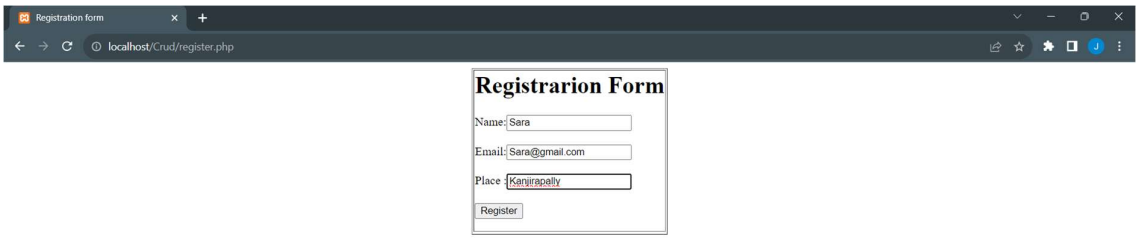
**delete.php**

```
<?php
include_once("connection.php");
$id = $_GET['id'];
$collection->deleteOne(['_id' => new MongoDB\BSON\ObjectID($id)]);
header("location:select.php");
?>
```

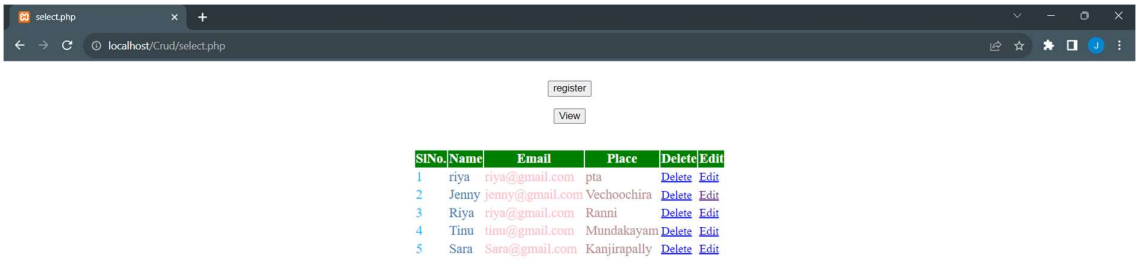
# Output Screenshot

## Connection.php

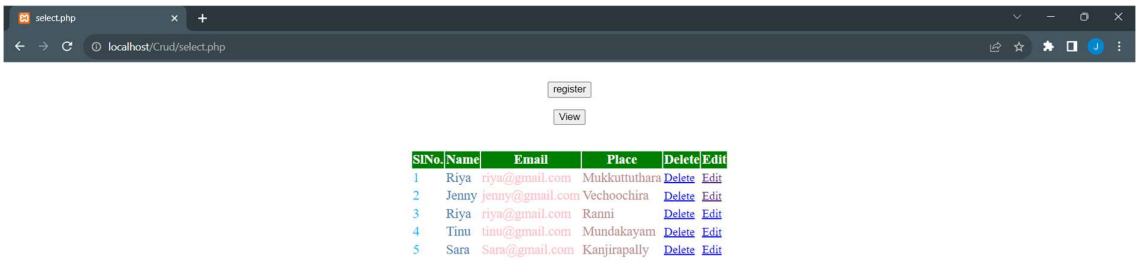

## Register.php



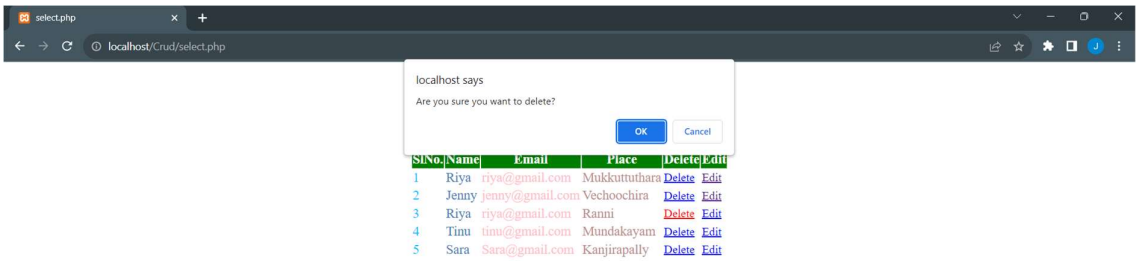

## Select.php

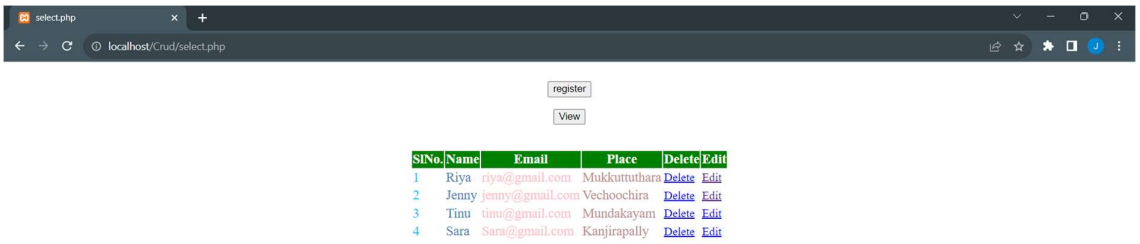


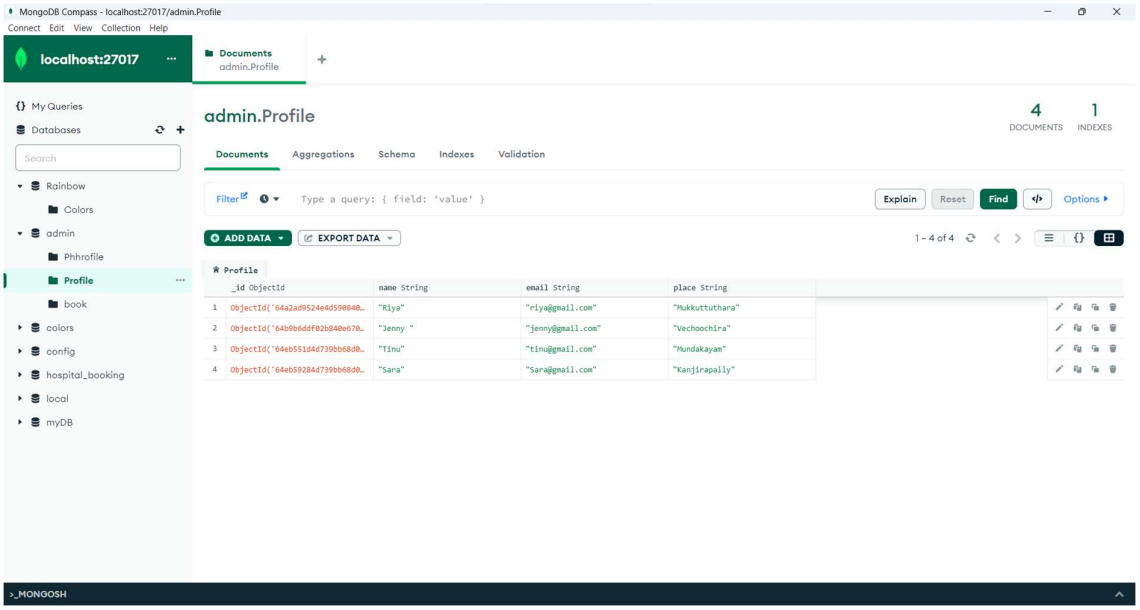

## Edit.php

## Delete.php




## Result

The program was executed and the result was successfully obtained. Thus, CO4 was obtained.

## Experiment No.: 13

**Aim:** Build sample collections/documents to perform the shell queries.

**CO5:** Understand the basic storage architecture of distributed file systems.

## Procedure

1. Create/Use a database
   > use expmongo


2. Display current database
   >db


3. Create a collection
   >db.createCollection("actors")


4. Insert data into the collection

5. Display documents in collection
   >db.actors.find()

```
C:\Program Files\MongoDB\Server\5.0\bin\mongo.exe
> db.actors.find()
{ "_id" : "trojan", "name" : "Ivan Trojan", "year" : 1964, "movies" : [ "samotari", "medvidek" ] }
{ "_id" : "machacek", "name" : "Jiri Machacek", "year" : 1966, "movies" : [ "medvidek", "vratnelahve", "samotari" ] }
{ "_id" : "schneiderova", "name" : "Jitka Schneiderova", "year" : 1973, "movies" : [ "samotari" ] }
{ "_id" : "sverak", "name" : "Zdenek Sverak", "year" : 1936, "movies" : [ "vratnelahve" ] }
{ "_id" : "geislerova", "name" : "Anna Geislerova", "year" : 1976 }
>
```

6. Display documents in collection
   >db.actors.find({ })

```
C:\Program Files\MongoDB\Server\5.0\bin\mongo.exe
> db.actors.find({ })
{ "_id" : "trojan", "name" : "Ivan Trojan", "year" : 1964, "movies" : [ "samotari", "medvidek" ] }
{ "_id" : "machacek", "name" : "Jiri Machacek", "year" : 1966, "movies" : [ "medvidek", "vratnelahve", "samotari" ] }
{ "_id" : "schneiderova", "name" : "Jitka Schneiderova", "year" : 1973, "movies" : [ "samotari" ] }
{ "_id" : "sverak", "name" : "Zdenek Sverak", "year" : 1936, "movies" : [ "vratnelahve" ] }
{ "_id" : "geislerova", "name" : "Anna Geislerova", "year" : 1976 }
>
```

7. Display
   >db.actors.find({ _id: "trojan" })

```
C:\Program Files\MongoDB\Server\5.0\bin\mongo.exe
> db.actors.find({ _id: "trojan" })
{ "_id" : "trojan", "name" : "Ivan Trojan", "year" : 1964, "movies" : [ "samotari", "medvidek" ] }
>
```

8. Display
   >db.actors.find({ name: "Ivan Trojan", year: 1964 })

```
C:\Program Files\MongoDB\Server\5.0\bin\mongo.exe
> db.actors.find({ name: "Ivan Trojan", year: 1964 })
{ "_id" : "trojan", "name" : "Ivan Trojan", "year" : 1964, "movies" : [ "samotari", "medvidek" ] }
>
```

9. Display
   >db.actors.find({ year: { $gte: 1960, $lte: 1980 } })

```
C:\Program Files\MongoDB\Server\5.0\bin\mongo.exe
> db.actors.find({ year: { $gte: 1960, $lte: 1980 } })
{ "_id" : "trojan", "name" : "Ivan Trojan", "year" : 1964, "movies" : [ "samotari", "medvidek" ] }
{ "_id" : "machacek", "name" : "Jiri Machacek", "year" : 1966, "movies" : [ "medvidek", "vratnelahve", "samotari" ] }
{ "_id" : "schneiderova", "name" : "Jitka Schneiderova", "year" : 1973, "movies" : [ "samotari" ] }
{ "_id" : "geislerova", "name" : "Anna Geislerova", "year" : 1976 }
>
```

10. Display
    >db.actors.find({ movies: { $exists: true } })

```
C:\Program Files\MongoDB\Server\5.0\bin\mongo.exe
> db.actors.find({ movies: { $exists: true } })
{ "_id" : "trojan", "name" : "Ivan Trojan", "year" : 1964, "movies" : [ "samotari", "medvidek" ] }
{ "_id" : "machacek", "name" : "Jiri Machacek", "year" : 1966, "movies" : [ "medvidek", "vratnelahve", "samotari" ] }
{ "_id" : "schneiderova", "name" : "Jitka Schneiderova", "year" : 1973, "movies" : [ "samotari" ] }
{ "_id" : "sverak", "name" : "Zdenek Sverak", "year" : 1936, "movies" : [ "vratnelahve" ] }
>
```

11. Display
    >db.actors.find({ movies: "medvidek" })

```
C:\Program Files\MongoDB\Server\5.0\bin\mongo.exe
> db.actors.find({ movies: "medvidek" })
{ "_id" : "trojan", "name" : "Ivan Trojan", "year" : 1964, "movies" : [ "samotari", "medvidek" ] }
{ "_id" : "machacek", "name" : "Jiri Machacek", "year" : 1966, "movies" : [ "medvidek", "vratnelahve", "samotari" ] }
>
```

12. Display
    >db.actors.find({ movies: { $in: [ "medvidek", "pelisky" ] } })

```
C:\Program Files\MongoDB\Server\5.0\bin\mongo.exe
> db.actors.find({ movies: { $in: [ "medvidek", "pelisky" ] } })
{ "_id" : "trojan", "name" : "Ivan Trojan", "year" : 1964, "movies" : [ "samotari", "medvidek" ] }
{ "_id" : "machacek", "name" : "Jiri Machacek", "year" : 1966, "movies" : [ "medvidek", "vratnelahve", "samotari" ] }
>
```

13. Display
    >db.actors.find({ movies: { $all: [ "medvidek", "pelisky" ] } })

```
C:\Program Files\MongoDB\Server\5.0\bin\mongo.exe
> db.actors.find({ movies: { $all: [ "medvidek", "pelisky" ] } })
>
```

14. Display
    >db.actors.find({ $or: [ { year: 1964 }, { rating: { $gte: 3 } } ] })

```
Select C:\Program Files\MongoDB\Server\5.0\bin\mongo.exe
> db.actors.find({ $or: [ { year: 1964 }, { rating: { $gte: 3 } } ] })
{ "_id" : "trojan", "name" : "Ivan Trojan", "year" : 1964, "movies" : [ "samotari", "medvidek" ] }
>
```

15. Display
    >db.actors.find({ rating: { $not: { $gte: 3 } } })

```
C:\Program Files\MongoDB\Server\5.0\bin\mongo.exe
> db.actors.find({ rating: { $not: { $gte: 3 } } })
{ "_id" : "trojan", "name" : "Ivan Trojan", "year" : 1964, "movies" : [ "samotari", "medvidek" ] }
{ "_id" : "machacek", "name" : "Jiri Machacek", "year" : 1966, "movies" : [ "medvidek", "vratnelahve", "samotari" ] }
{ "_id" : "schneiderova", "name" : "Jitka Schneiderova", "year" : 1973, "movies" : [ "samotari" ] }
{ "_id" : "sverak", "name" : "Zdenek Sverak", "year" : 1936, "movies" : [ "vratnelahve" ] }
{ "_id" : "geislerova", "name" : "Anna Geislerova", "year" : 1976 }
>
```

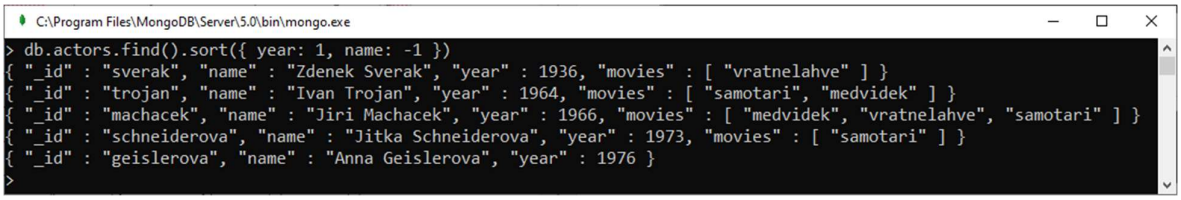
16. Display
    >db.actors.find({ }, { name: 1, year: 1 })

```
C:\Program Files\MongoDB\Server\5.0\bin\mongo.exe
> db.actors.find({ }, { name: 1, year: 1 })
{ "_id" : "trojan", "name" : "Ivan Trojan", "year" : 1964 }
{ "_id" : "machacek", "name" : "Jiri Machacek", "year" : 1966 }
{ "_id" : "schneiderova", "name" : "Jitka Schneiderova", "year" : 1973 }
{ "_id" : "sverak", "name" : "Zdenek Sverak", "year" : 1936 }
{ "_id" : "geislerova", "name" : "Anna Geislerova", "year" : 1976 }
>
```

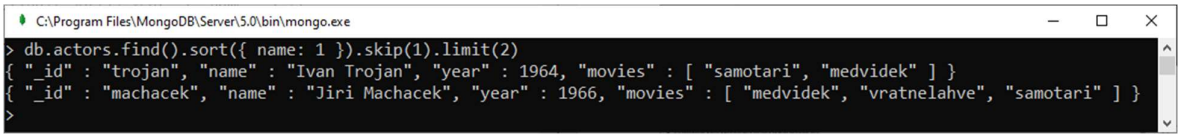
17. Display
    >db.actors.find({ }, { movies: 0, _id: 0 })


18. Display
    >db.actors.find({ }, { name: 1, movies: { $slice: 2 }, _id: 0 })

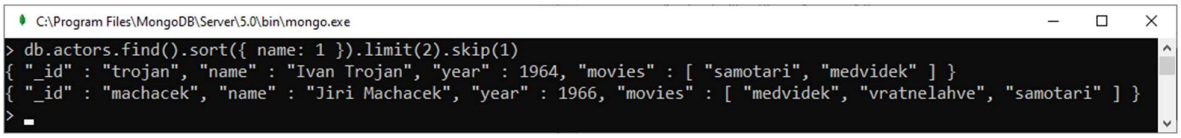

19. Display
    >db.actors.find().sort({ year: 1, name: -1 })




20. Display
    >db.actors.find().sort({ name: 1 }).skip(1).limit(2)




21. Display
    >db.actors.find().sort({ name: 1 }).limit(2).skip(1)




## Result

The program was executed and the result was successfully obtained. Thus, CO5 was obtained.

## Experiment No.: 14

**Aim:** To familiarize with indexing in MongoDB.

**CO5:** Understand the basic storage architecture of distributed file systems.

## Procedure

1.Input data –

db.products.insertMany( [

{ _id: 10, item: "large box", qty: 50 },

{ _id: 11, item: "medium box", qty: 30 },

{ _id: 12, item: "envelope", qty: 100},

{ _id: 13, item: "tape", qty: 20},

{ _id: 14, item: "bubble wrap", qty: 70}

])

```
> db.products.insertMany( [
... { _id: 10, item: "large box", qty: 50 },
... { _id: 11, item: "medium box", qty: 30 },
... { _id: 12, item: "envelope", qty: 100},
... { _id: 13, item: "tape", qty: 20},
... { _id: 14, item: "bubble wrap", qty: 70}
... ])
{ "acknowledged" : true, "insertedIds" : [ 10, 11, 12, 13, 14 ] }
```

```
> db.products.find()
{ "_id" : 10, "item" : "large box", "qty" : 50 }
{ "_id" : 11, "item" : "medium box", "qty" : 30 }
{ "_id" : 12, "item" : "envelope", "qty" : 100 }
{ "_id" : 13, "item" : "tape", "qty" : 20 }
{ "_id" : 14, "item" : "bubble wrap", "qty" : 70 }
```

Example 1 – Create ascending index on a field

db.collection.createIndex( { item: 1 } )

```
> db.collection.createIndex( { item: 1 } )
{
        "numIndexesBefore" : 1,
        "numIndexesAfter" : 2,
        "createdCollectionAutomatically" : true,
        "ok" : 1
}
```

Example 2 – Create descending index on a field

db.collection.createIndex( { qty: -1 } )

```
> db.collection.createIndex( { qty: -1 } )
{
        "numIndexesBefore" : 2,
        "numIndexesAfter" : 3,
        "createdCollectionAutomatically" : false,
        "ok" : 1
}
```

```
> db.products.getIndexes()
[ { "v" : 2, "key" : { "_id" : 1 }, "name" : "_id_" } ]
```

2.Specify the name to the Index

Example – Create index with the index name

db.products.createIndex(

{ item: 1, quantity: -1 } ,

{ name: "query for inventory" }

)

```
> db.products.createIndex( { item: 1, quantity: -1 } , { name: "query for inventory" } )
{
        "numIndexesBefore" : 1,
        "numIndexesAfter" : 2,
        "createdCollectionAutomatically" : true,
        "ok" : 1
}
```

```
> db.products.getIndexes()
[
        {
                "v" : 2,
                "key" : {
                        "_id" : 1
                },
                "name" : "_id_"
        },
        {
                "v" : 2,
                "key" : {
                        "item" : 1,
                        "quantity" : -1
                },
                "name" : "query for inventory"
        }
]
```

3.dropIndex() Method Example

Example 1 – Drop index by index document

db.products.dropIndex(

{ item: 1, quantity: -1 }

)

```
> db.products.dropIndex(
... { item: 1, quantity: -1 }
... )
{ "nIndexesWas" : 2, "ok" : 1 }
```

Example 2 – Drop index by index name

db.products.dropIndex( "query for inventory“ )

```
> db.products.dropIndex( "query for inventory" )
{ "nIndexesWas" : 2, "ok" : 1 }
```

## Result

The program was executed and the result was successfully obtained. Thus, CO5 was obtained.

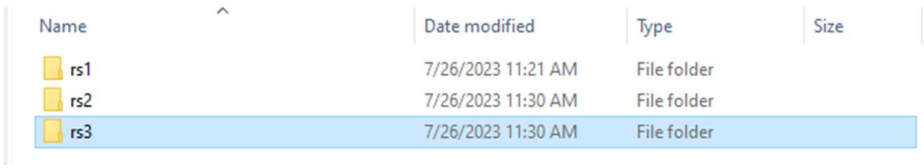
## Experiment No.: 15

**Aim:** Implementation of Replica Set on MongoDB

**CO5:** Understand the basic storage architecture of distributed file systems.

## Procedure

1. Creating 3 Replica set



| Name | Date modified | Type | Size |
|---|---|---|---|
| rs1 | 7/26/2023 11:21 AM | File folder | |
| rs2 | 7/26/2023 11:30 AM | File folder | |
| rs3 | 7/26/2023 11:30 AM | File folder | |

```
Microsoft Windows [Version 10.0.19045.3208]
(c) Microsoft Corporation. All rights reserved.

C:\Program Files\MongoDB\Server\5.0\bin>start mongod -replSet datascience -logpath \data\rs1\1.log -dbpath \data\rs1 -port 27018

C:\Program Files\MongoDB\Server\5.0\bin>start mongod -replSet datascience -logpath \data\rs2\2.log -dbpath \data\rs2 -port 27019

C:\Program Files\MongoDB\Server\5.0\bin>start mongod -replSet datascience -logpath \data\rs3\3.log -dbpath \data\rs3 -port 27020

C:\Program Files\MongoDB\Server\5.0\bin>
```

```
C:\Program Files\MongoDB\Server\5.0\bin>mongo --port 27018
MongoDB shell version v5.0.2
connecting to: mongodb://127.0.0.1:27018/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("16a80073-b42e-4e43-a85c-64128346f667") }
MongoDB server version: 5.0.2
================
Warning: the "mongo" shell has been superseded by "mongosh",
which delivers improved usability and compatibility.The "mongo" shell has been deprecated and will be removed in
an upcoming release.
We recommend you begin using "mongosh".
For installation instructions, see
https://docs.mongodb.com/mongodb-shell/install/
================
---
The server generated these startup warnings when booting:
        2023-07-26T11:10:33.628+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
        2023-07-26T11:10:33.628+05:30: This server is bound to localhost. Remote systems will be unable to connect to this server. Start the server with --bind_ip <addr
ess> to specify which IP addresses it should serve responses from, or with --bind_ip_all to bind to all interfaces. If this behavior is desired, start the server with
-bind_ip 127.0.0.1 to disable this warning
---
---
        Enable MongoDB's free cloud-based monitoring service, which will then receive and display
        metrics about your deployment (disk utilization, CPU, operation statistics, etc).

        The monitoring data will be available on a MongoDB website with a unique URL accessible to you
        and anyone you share the URL with. MongoDB may use this information to make product
        improvements and to suggest MongoDB products and deployment options to you.

        To enable free monitoring, run the following command: db.enableFreeMonitoring()
        To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---
```

2. Configuration of Replica set

```
---
> config={_id:"datascience",members:[{_id:0,host:"localhost:27018"},{_id:1,host:"localhost:27019"},{_id:2,host:"localhost:27020"}]}
{
        "_id" : "datascience",
        "members" : [
                {
                        "_id" : 0,
                        "host" : "localhost:27018"
                },
                {
                        "_id" : 1,
                        "host" : "localhost:27019"
                },
                {
                        "_id" : 2,
                        "host" : "localhost:27020"
                }
        ]
}
> rs.initiate(config)
{
        "ok" : 1,
        "$clusterTime" : {
                "clusterTime" : Timestamp(1690350364, 1),
                "signature" : {
                        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                        "keyId" : NumberLong(0)
                }
        },
        "operationTime" : Timestamp(1690350364, 1)
```

## 3. Status

```
datascience:SECONDARY> rs.status()
{
        "set" : "datascience",
        "date" : ISODate("2023-07-26T05:46:19.076Z"),
        "myState" : 1,
        "term" : NumberLong(1),
        "syncSourceHost" : "",
        "syncSourceId" : -1,
        "heartbeatIntervalMillis" : NumberLong(2000),
        "majorityVoteCount" : 2,
        "writeMajorityCount" : 2,
        "votingMembersCount" : 3,
        "writableVotingMembersCount" : 3,
        "optimes" : {
                "lastCommittedOpTime" : {
                        "ts" : Timestamp(1690350377, 1),
                        "t" : NumberLong(1)
                },
                "lastCommittedWallTime" : ISODate("2023-07-26T05:46:17.200Z"),
                "readConcernMajorityOpTime" : {
                        "ts" : Timestamp(1690350377, 1),
                        "t" : NumberLong(1)
                },
                "appliedOpTime" : {
                        "ts" : Timestamp(1690350377, 1),
                        "t" : NumberLong(1)
                },
                "durableOpTime" : {
                        "ts" : Timestamp(1690350377, 1),
                        "t" : NumberLong(1)
                },
                "lastAppliedWallTime" : ISODate("2023-07-26T05:46:17.200Z"),
                "lastDurableWallTime" : ISODate("2023-07-26T05:46:17.200Z")
        },
        "lastStableRecoveryTimestamp" : Timestamp(1690350375, 3),
        "electionCandidateMetrics" : {
                "lastElectionReason" : "electionTimeout",
                "lastElectionDate" : ISODate("2023-07-26T05:46:15.010Z"),
                "electionTerm" : NumberLong(1),
                "lastCommittedOpTimeAtElection" : {
                        "ts" : Timestamp(0, 0),
                        "t" : NumberLong(-1)
```

```
                "lastSeenOpTimeAtElection" : {
                        "ts" : Timestamp(1690350364, 1),
                        "t" : NumberLong(-1)
                },
                "numVotesNeeded" : 2,
                "priorityAtElection" : 1,
                "electionTimeoutMillis" : NumberLong(10000),
                "numCatchUpOps" : NumberLong(0),
                "newTermStartDate" : ISODate("2023-07-26T05:46:15.204Z"),
                "wMajorityWriteAvailabilityDate" : ISODate("2023-07-26T05:46:15.839Z"
        },
        "members" : [
                {
                        "_id" : 0,
                        "name" : "localhost:27018",
                        "health" : 1,
                        "state" : 1,
                        "stateStr" : "PRIMARY",
                        "uptime" : 347,
                        "optime" : {
                                "ts" : Timestamp(1690350377, 1),
                                "t" : NumberLong(1)
                        },
                        "optimeDate" : ISODate("2023-07-26T05:46:17Z"),
                        "syncSourceHost" : "",
                        "syncSourceId" : -1,
                        "infoMessage" : "",
                        "electionTime" : Timestamp(1690350375, 1),
                        "electionDate" : ISODate("2023-07-26T05:46:15Z"),
                        "configVersion" : 1,
                        "configTerm" : 1,
                        "self" : true,
                        "lastHeartbeatMessage" : ""
                },
                {
                        "_id" : 1,
                        "name" : "localhost:27019",
                        "health" : 1,
                        "state" : 2,
                        "stateStr" : "SECONDARY",
                        "uptime" : 14,
                        "optime" : {
                                "ts" : Timestamp(1690350375, 6),
                                "t" : NumberLong(1)
```

## Primary

```
datascience:PRIMARY> show dbs;
admin   0.000GB
config  0.000GB
local   0.000GB
datascience:PRIMARY> use db1;
switched to db db1
datascience:PRIMARY> db.student.insert({name:"johan"})
WriteResult({ "nInserted" : 1 })
datascience:PRIMARY> use admin
switched to db admin
datascience:PRIMARY> db.shutdownServer()
server should be down...
```

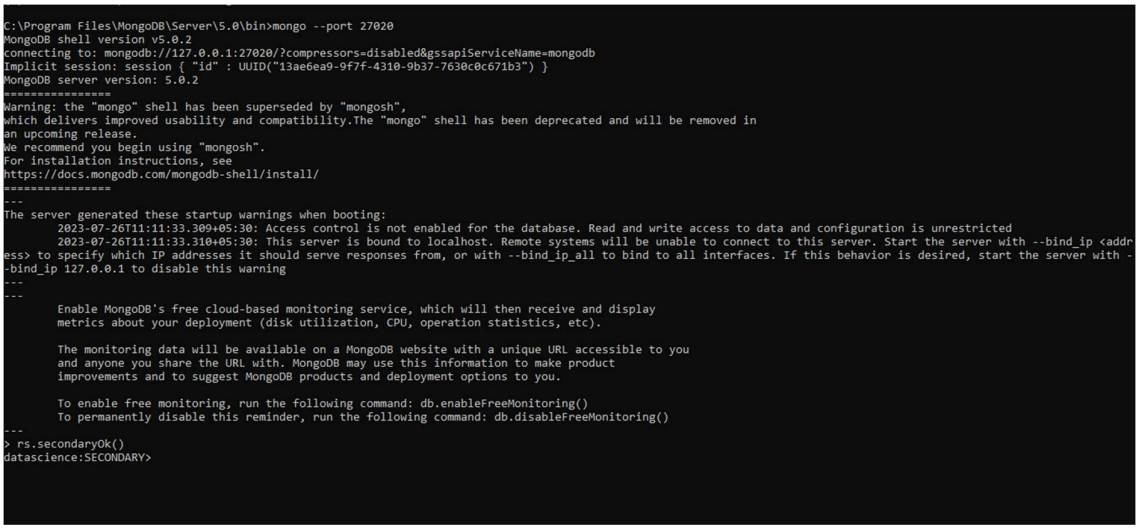
```
C:\Program Files\MongoDB\Server\5.0\bin>mongo --port 27019
MongoDB shell version v5.0.2
connecting to: mongodb://127.0.0.1:27019/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("42943329-b665-4cb5-a195-9342fd7dcde2") }
MongoDB server version: 5.0.2
================
Warning: the "mongo" shell has been superseded by "mongosh",
which delivers improved usability and compatibility.The "mongo" shell has been deprecated and will be removed in
an upcoming release.
We recommend you begin using "mongosh".
For installation instructions, see
https://docs.mongodb.com/mongodb-shell/install/
================
---
The server generated these startup warnings when booting:
        2023-07-26T11:11:03.582+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
        2023-07-26T11:11:03.583+05:30: This server is bound to localhost. Remote systems will be unable to connect to this server. Start the server with --bind_ip <addr
ess> to specify which IP addresses it should serve responses from, or with --bind_ip_all to bind to all interfaces. If this behavior is desired, start the server with -
-bind_ip 127.0.0.1 to disable this warning
---
---
        Enable MongoDB's free cloud-based monitoring service, which will then receive and display
        metrics about your deployment (disk utilization, CPU, operation statistics, etc).

        The monitoring data will be available on a MongoDB website with a unique URL accessible to you
        and anyone you share the URL with. MongoDB may use this information to make product
        improvements and to suggest MongoDB products and deployment options to you.

        To enable free monitoring, run the following command: db.enableFreeMonitoring()
        To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
```

## Secondary

```
> rs.secondaryOk()
datascience:SECONDARY> show dbs;
admin   0.000GB
config  0.000GB
db1     0.000GB
local   0.000GB
datascience:SECONDARY> use db1;
switched to db db1
datascience:SECONDARY> db.student.find()
{ "_id" : ObjectId("64c0b3ae5e25987f931f5dee"), "name" : "johan" }
datascience:SECONDARY> rs.status()
{
        "set" : "datascience",
        "date" : ISODate("2023-07-26T05:51:36.173Z"),
        "myState" : 1,
        "term" : NumberLong(2),
        "syncSourceHost" : "",
        "syncSourceId" : -1,
        "heartbeatIntervalMillis" : NumberLong(2000),
        "majorityVoteCount" : 2,
        "writeMajorityCount" : 2,
        "votingMembersCount" : 3,
        "writableVotingMembersCount" : 3,
        "optimes" : {
                "lastCommittedOpTime" : {
                        "ts" : Timestamp(1690350694, 1),
                        "t" : NumberLong(2)
                },
                "lastCommittedWallTime" : ISODate("2023-07-26T05:51:34.284Z"),
                "readConcernMajorityOpTime" : {
                        "ts" : Timestamp(1690350694, 1),
                        "t" : NumberLong(2)
                },
                "appliedOpTime" : {
                        "ts" : Timestamp(1690350694, 1),
                        "t" : NumberLong(2)
                },
                "durableOpTime" : {
                        "ts" : Timestamp(1690350694, 1),
                        "t" : NumberLong(2)
                },
                "lastAppliedWallTime" : ISODate("2023-07-26T05:51:34.284Z"),
                "lastDurableWallTime" : ISODate("2023-07-26T05:51:34.284Z")
```




## Result

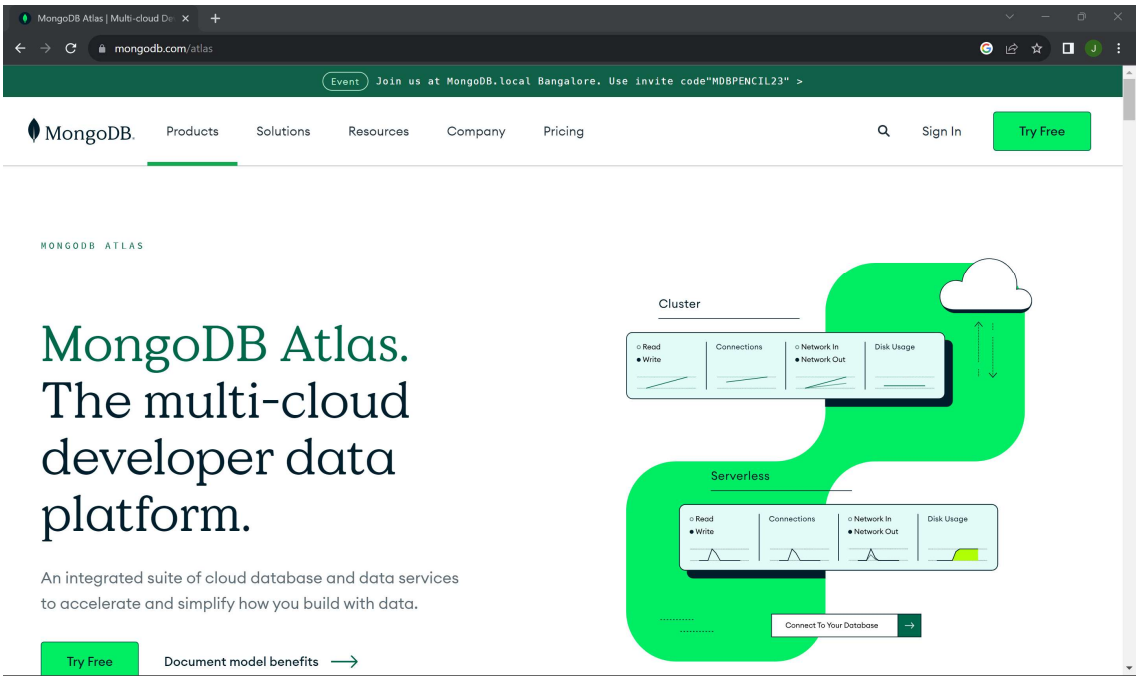
The program was executed and the result was successfully obtained. Thus, CO5 was obtained.

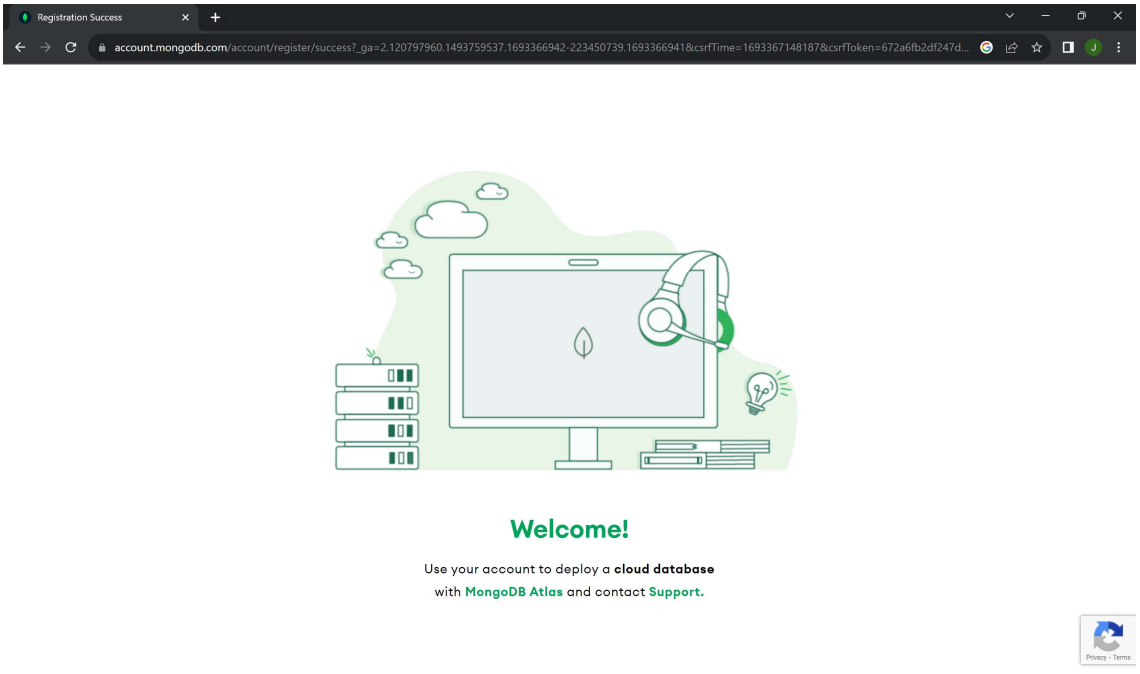
## Experiment No.: 16

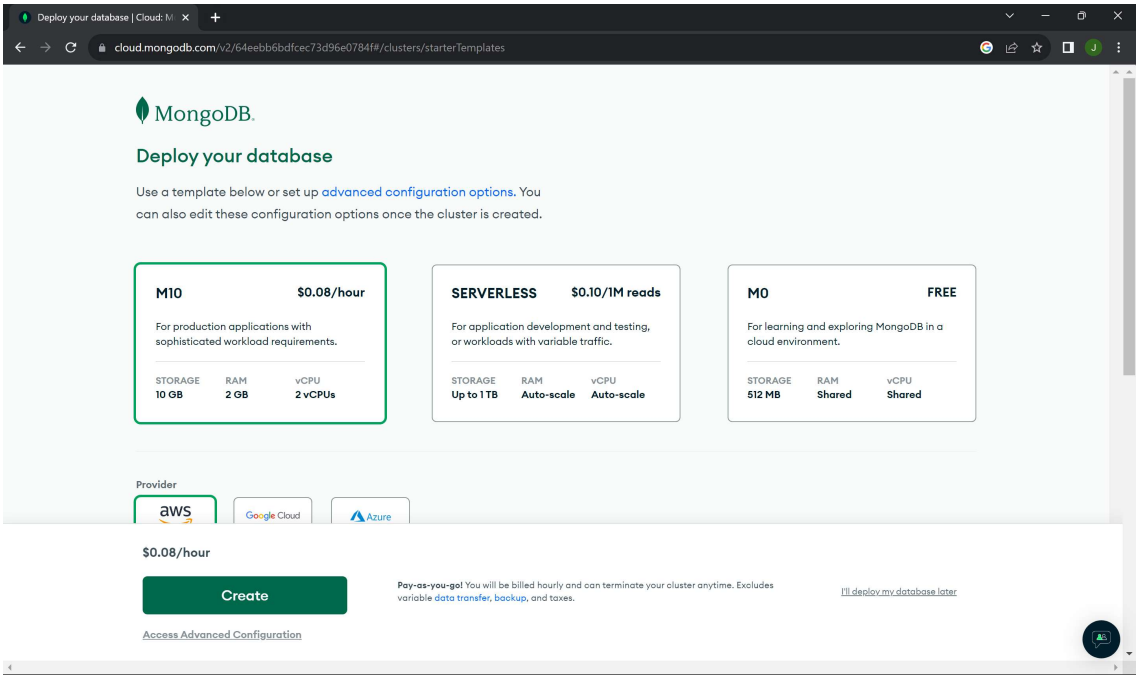
**Aim:** Usage of Cloud Storage Management Systems: MongoDB Atlas.

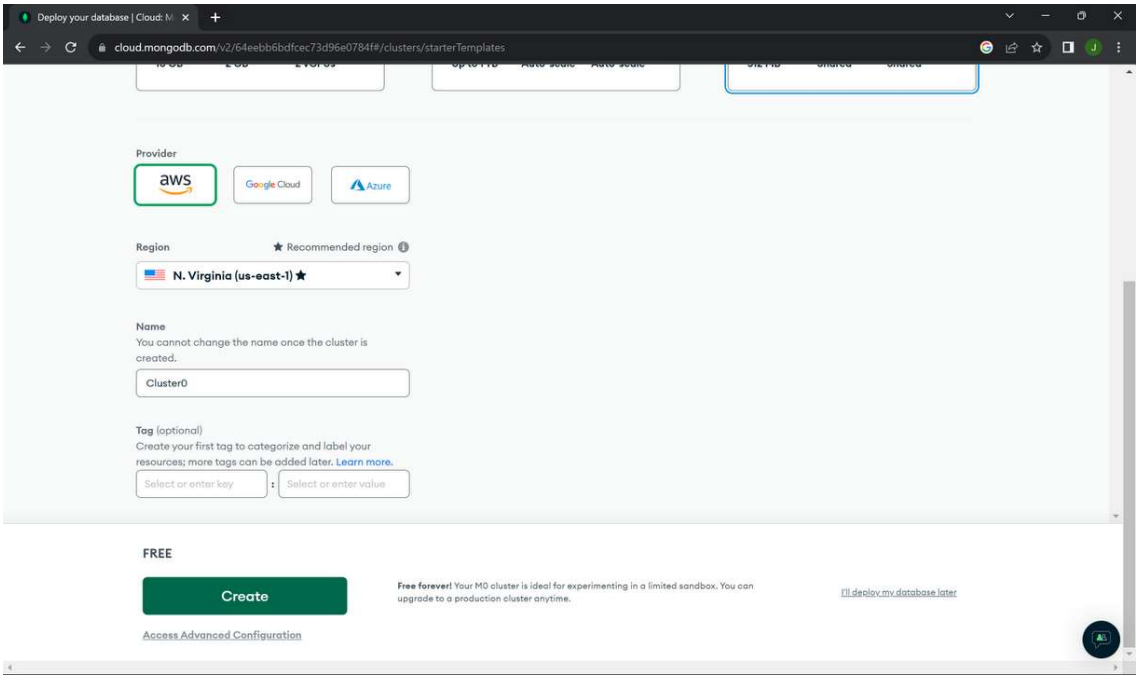
**CO6:** Design and deployment of NoSQL databases with real time requirements.

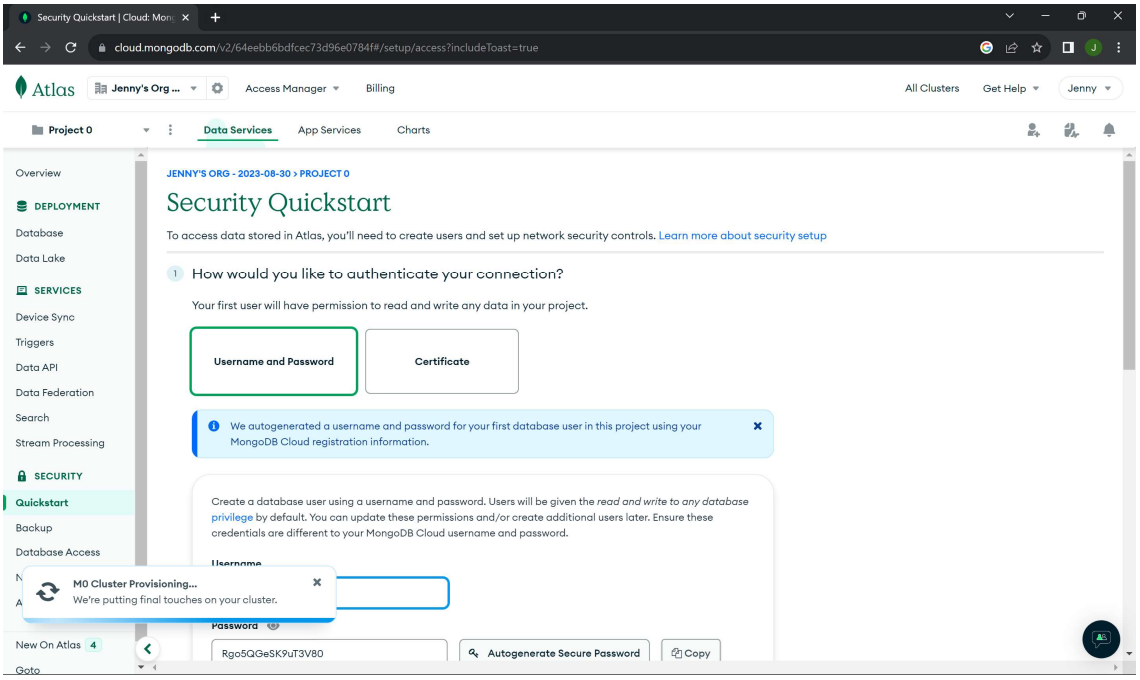
**Procedure**

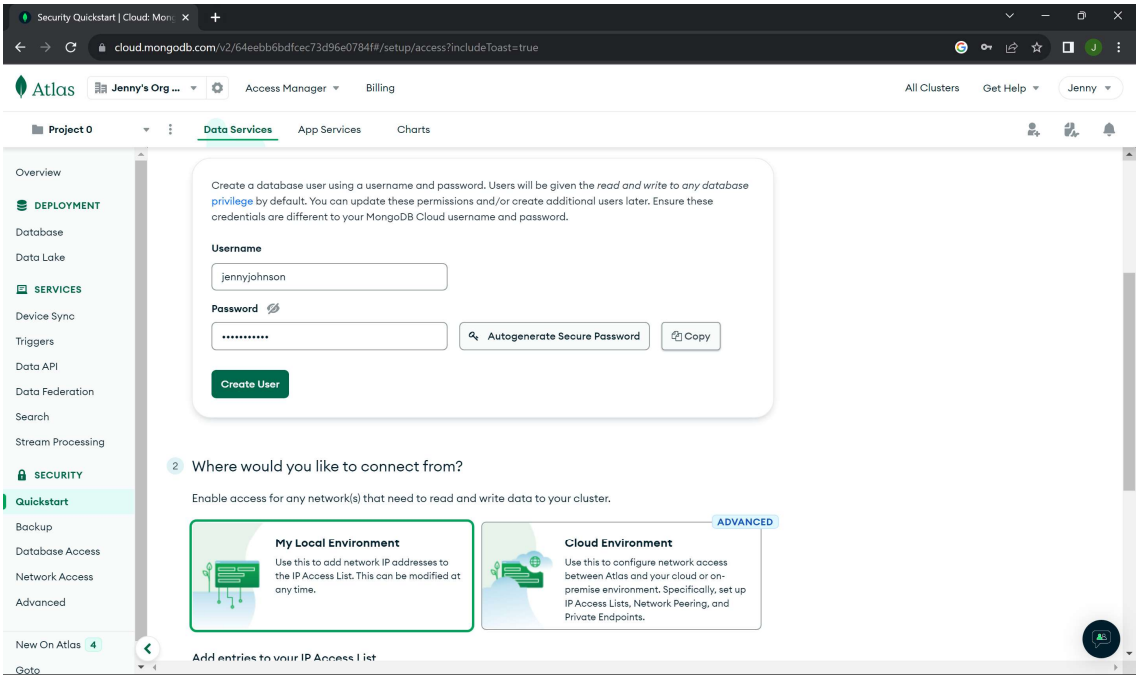


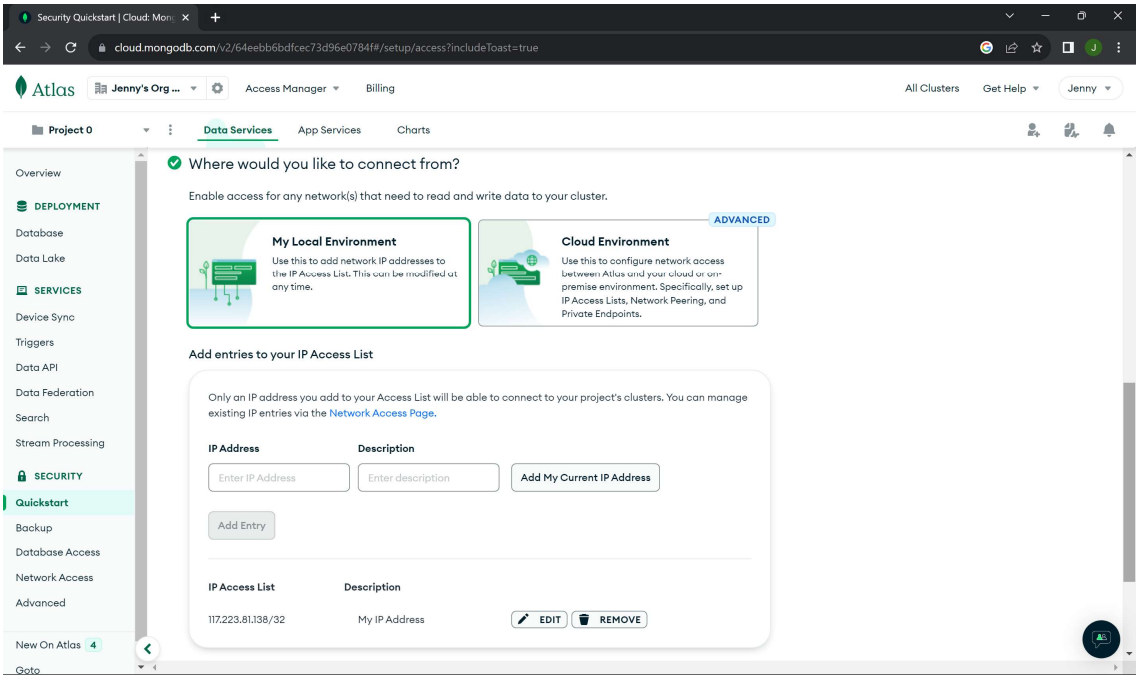



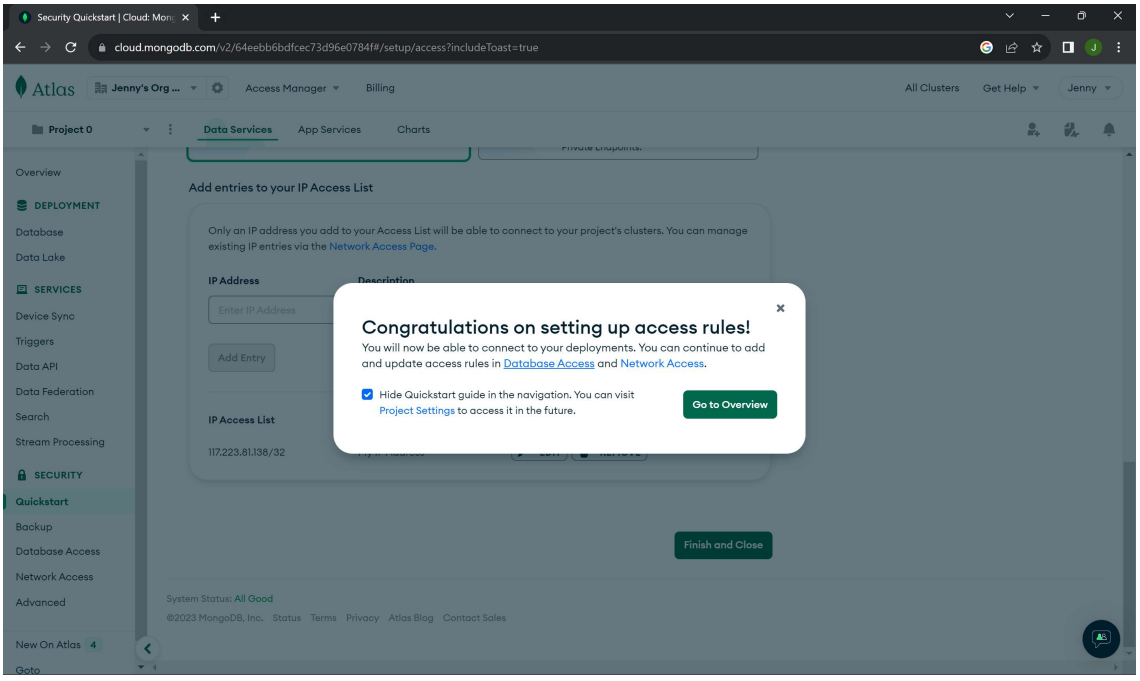

Deploy your database | Cloud: M
cloud.mongodb.com/v2/64eebb6bdfcec73d96e0784f#/clusters/starterTemplates
MongoDB.
Deploy your database
Use a template below or set up advanced configuration options. You can also edit these configuration options once the cluster is created.
M10 $0.08/hour
For production applications with sophisticated workload requirements.
STORAGE 10 GB RAM 2 GB vCPU 2 vCPUs
SERVERLESS $0.10/1M reads
For application development and testing, or workloads with variable traffic.
STORAGE Up to 1 TB RAM Auto-scale vCPU Auto-scale
M0 FREE
For learning and exploring MongoDB in a cloud environment.
STORAGE 512 MB RAM Shared vCPU Shared
Provider
aws
Google Cloud
Azure
$0.08/hour
Create
Pay-as-you-go! You will be billed hourly and can terminate your cluster anytime. Excludes variable data transfer, backup, and taxes.
I'll deploy my database later
Access Advanced Configuration

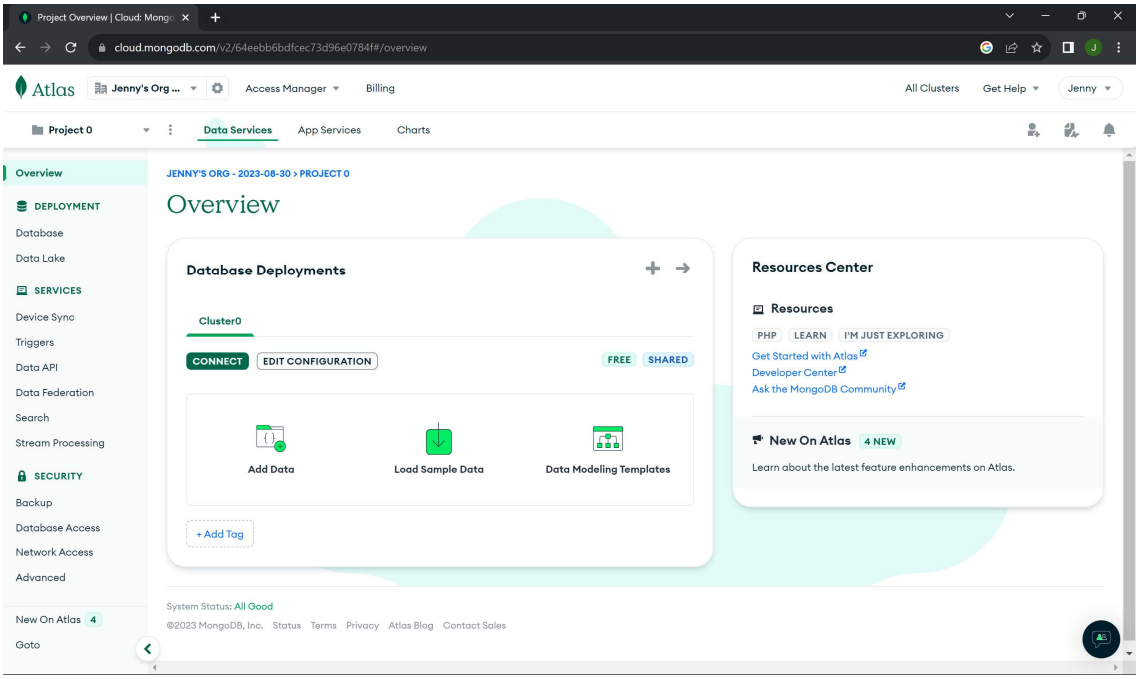

Deploy your database | Cloud: M
cloud.mongodb.com/v2/64eebb6bdfcec73d96e0784f#/clusters/starterTemplates
Provider
aws
Google Cloud
Azure
Region
Recommended region
N. Virginia (us-east-1)
Name
You cannot change the name once the cluster is created.
Cluster0
Tag (optional)
Create your first tag to categorize and label your resources; more tags can be added later. Learn more.
Select or enter key
Select or enter value
FREE
Create
Free forever! Your M0 cluster is ideal for experimenting in a limited sandbox. You can upgrade to a production cluster anytime.
I'll deploy my database later
Access Advanced Configuration

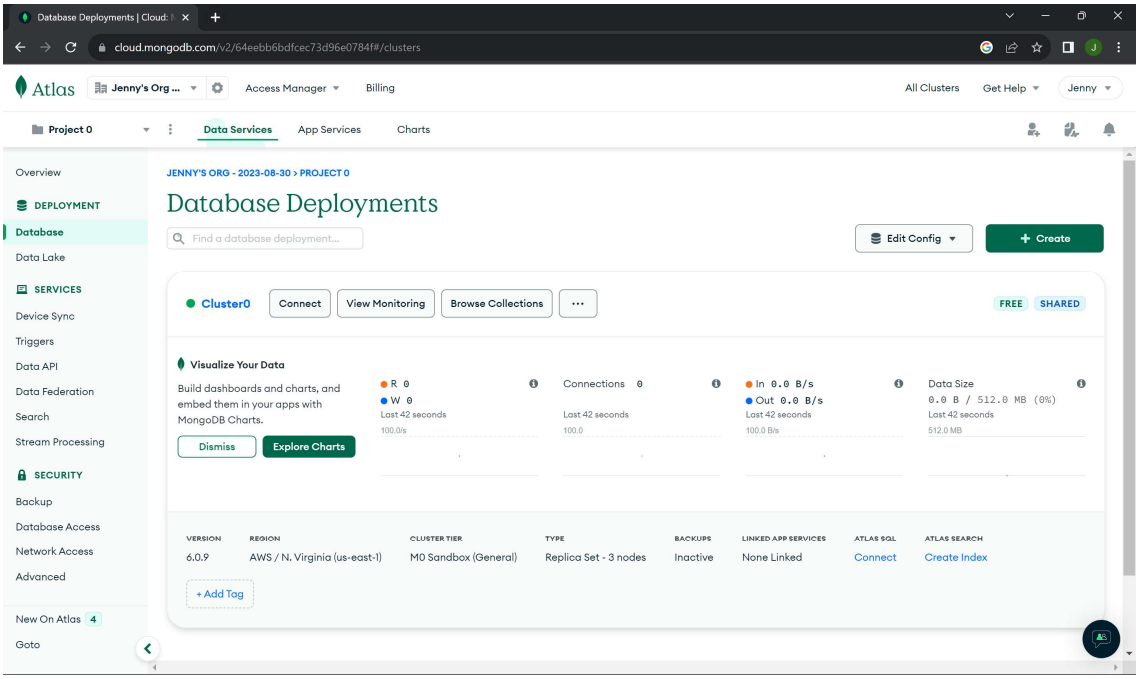

Security Quickstart | Cloud: Mon
cloud.mongodb.com/v2/64eebb6bdfcec73d96e0784f#/setup/access?includeToast=true
Atlas
Jenny's Org ...
Access Manager
Billing
All Clusters
Get Help
Jenny
Project 0
Data Services
App Services
Charts
Overview
DEPLOYMENT
Database
Data Lake
SERVICES
Device Sync
Triggers
Data API
Data Federation
Search
Stream Processing
SECURITY
Quickstart
Backup
Database Access
New On Atlas 4
Goto
JENNY'S ORG - 2023-08-30 > PROJECT 0
Security Quickstart
To access data stored in Atlas, you'll need to create users and set up network security controls. Learn more about security setup
1 How would you like to authenticate your connection?
Your first user will have permission to read and write any data in your project.
Username and Password
Certificate
We autogenerated a username and password for your first database user in this project using your MongoDB Cloud registration information.
Create a database user using a username and password. Users will be given the read and write to any database privilege by default. You can update these permissions and/or create additional users later. Ensure these credentials are different to your MongoDB Cloud username and password.
Username
Password
Rgo5QGeSK9uT3V80
Autogenerate Secure Password
Copy
M0 Cluster Provisioning...
We're putting final touches on your cluster.

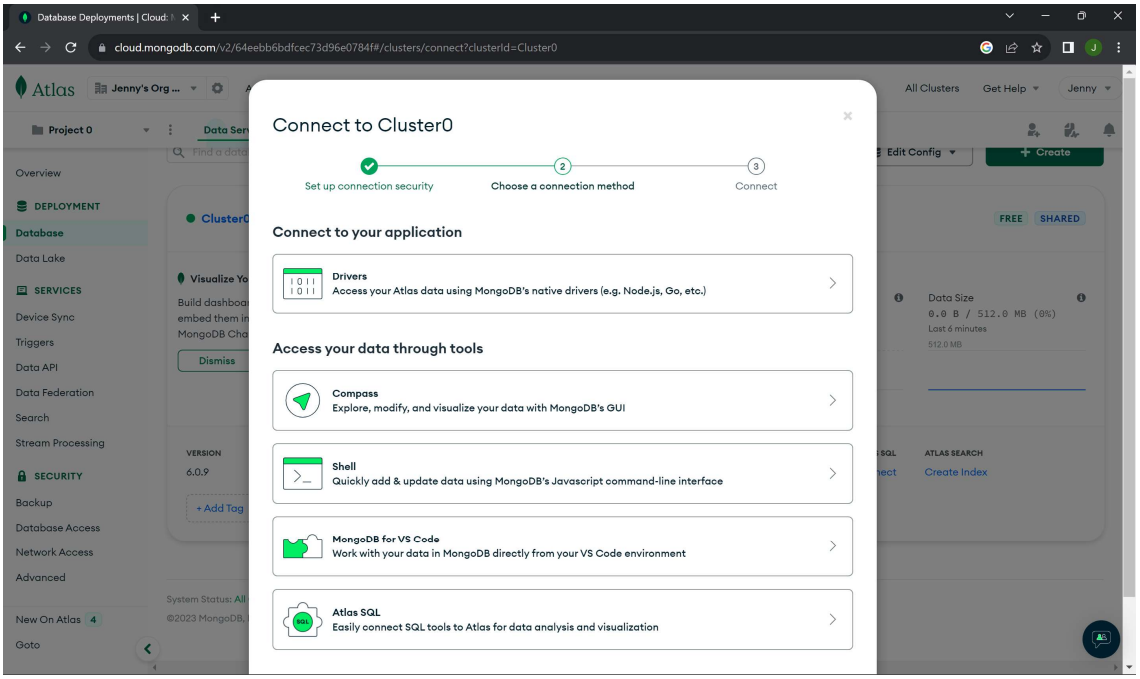

Security Quickstart | Cloud: Mon
cloud.mongodb.com/v2/64eebb6bdfcec73d96e0784f#/setup/access?includeToast=true
Atlas
Jenny's Org ...
Access Manager
Billing
All Clusters
Get Help
Jenny
Project 0
Data Services
App Services
Charts
Overview
DEPLOYMENT
Database
Data Lake
SERVICES
Device Sync
Triggers
Data API
Data Federation
Search
Stream Processing
SECURITY
Quickstart
Backup
Database Access
Network Access
Advanced
New On Atlas 4
Goto
Create a database user using a username and password. Users will be given the read and write to any database privilege by default. You can update these permissions and/or create additional users later. Ensure these credentials are different to your MongoDB Cloud username and password.
Username
jennyjohnson
Password
Autogenerate Secure Password
Copy
Create User
2 Where would you like to connect from?
Enable access for any network(s) that need to read and write data to your cluster.
My Local Environment
Use this to add network IP addresses to the IP Access List. This can be modified at any time.
ADVANCED
Cloud Environment
Use this to configure network access between Atlas and your cloud or on-premise environment. Specifically, set up IP Access Lists, Network Peering, and Private Endpoints.
Add entries to your IP Access List

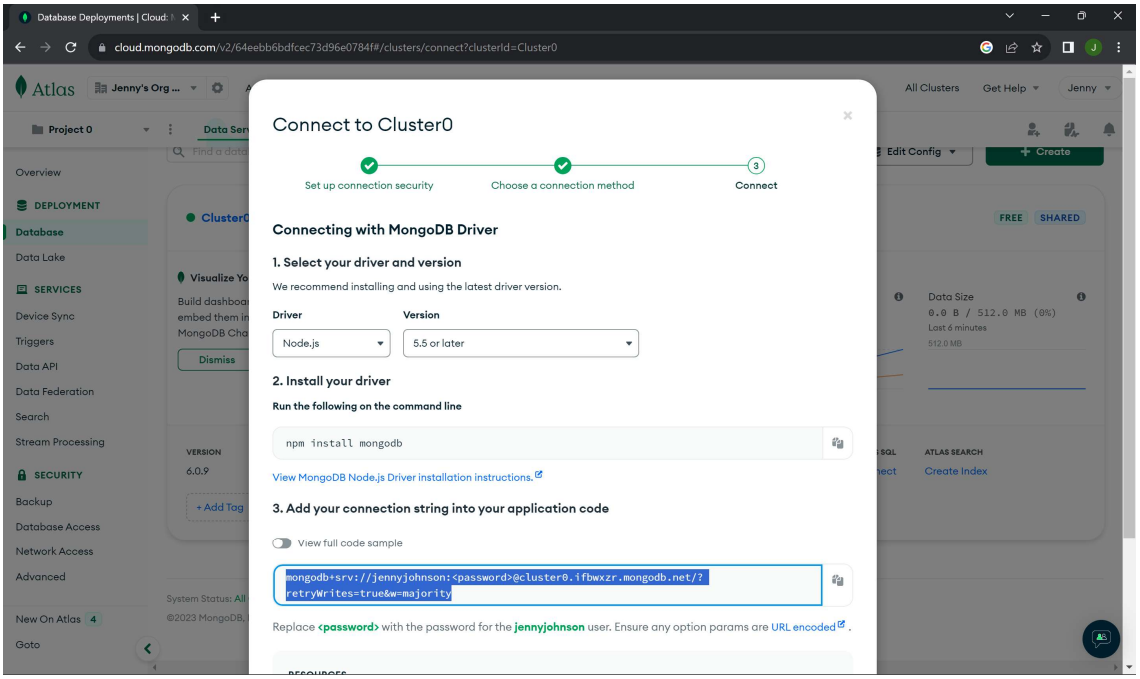

Security Quickstart | Cloud: Mon
cloud.mongodb.com/v2/64eebb6bdfcec73d96e0784f#/setup/access?includeToast=true
Atlas
Jenny's Org ...
Access Manager
Billing
All Clusters
Get Help
Jenny
Project 0
Data Services
App Services
Charts
Overview
DEPLOYMENT
Database
Data Lake
SERVICES
Device Sync
Triggers
Data API
Data Federation
Search
Stream Processing
SECURITY
Quickstart
Backup
Database Access
Network Access
Advanced
New On Atlas 4
Goto
Where would you like to connect from?
Enable access for any network(s) that need to read and write data to your cluster.
My Local Environment
Use this to add network IP addresses to the IP Access List. This can be modified at any time.
ADVANCED
Cloud Environment
Use this to configure network access between Atlas and your cloud or on-premise environment. Specifically, set up IP Access Lists, Network Peering, and Private Endpoints.
Add entries to your IP Access List
Only an IP address you add to your Access List will be able to connect to your project's clusters. You can manage existing IP entries via the Network Access Page.
IP Address
Description
Enter IP Address
Enter description
Add My Current IP Address
Add Entry
IP Access List
Description
117.223.81.138/32
My IP Address
EDIT
REMOVE

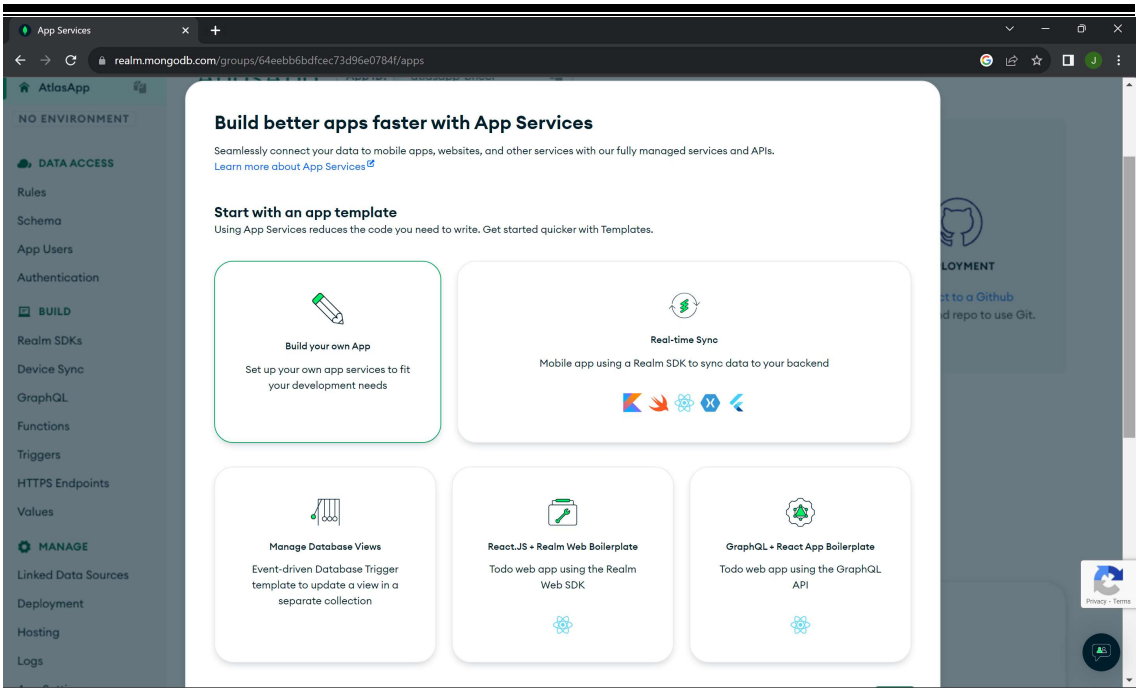

Security Quickstart | Cloud: Mon
cloud.mongodb.com/v2/64eebb6bdfcec73d96e0784f#/setup/access?includeToast=true
Atlas
Jenny's Org ...
Access Manager
Billing
All Clusters
Get Help
Jenny
Project 0
Data Services
App Services
Charts
Add entries to your IP Access List
Only an IP address you add to your Access List will be able to connect to your project's clusters. You can manage existing IP entries via the Network Access Page.
IP Address
Description
Enter IP Address
Add Entry
IP Access List
117.223.81.138/32
Congratulations on setting up access rules!
You will now be able to connect to your deployments. You can continue to add and update access rules in Database Access and Network Access.
Hide Quickstart guide in the navigation. You can visit Project Settings to access it in the future.
Go to Overview
Finish and Close
System Status: All Good
©2023 MongoDB, Inc. Status Terms Privacy Atlas Blog Contact Sales

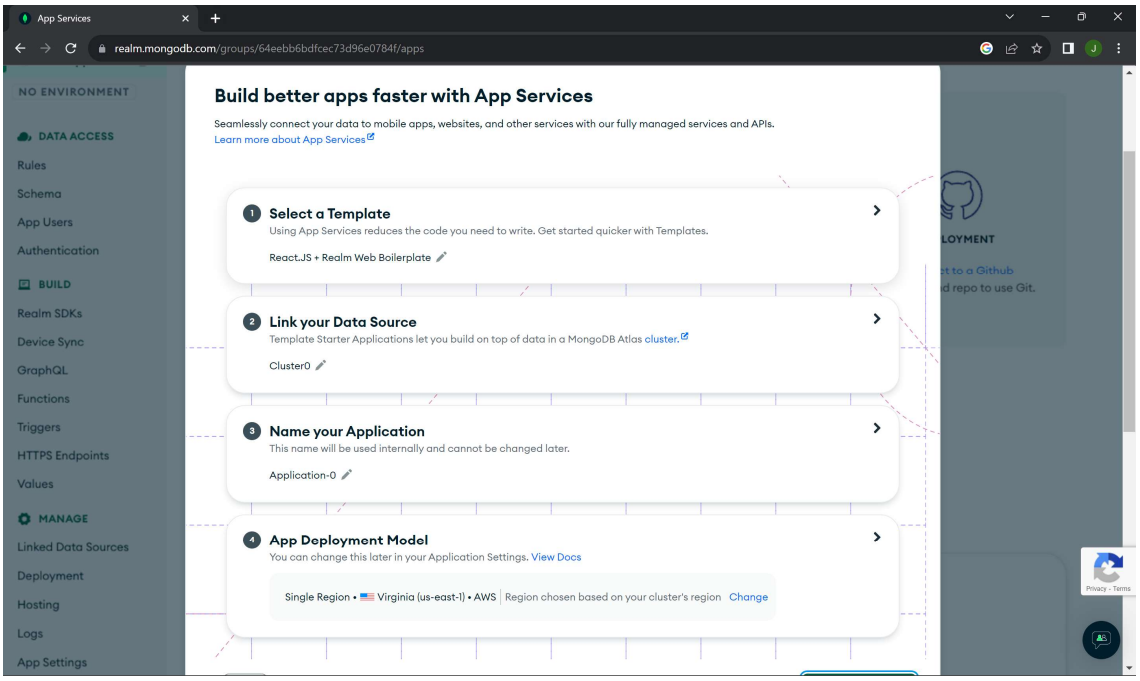

Project Overview | Cloud: Mongo
cloud.mongodb.com/v2/64eebb6bdfcec73d96e0784f#/overview
Atlas
Jenny's Org ...
Access Manager
Billing
All Clusters
Get Help
Jenny
Project 0
Data Services
App Services
Charts
Overview
DEPLOYMENT
Database
Data Lake
SERVICES
Device Sync
Triggers
Data API
Data Federation
Search
Stream Processing
SECURITY
Backup
Database Access
Network Access
Advanced
New On Atlas 4
Goto
JENNY'S ORG - 2023-08-30 > PROJECT 0
Overview
Database Deployments
Cluster0
CONNECT
EDIT CONFIGURATION
FREE
SHARED
Add Data
Load Sample Data
Data Modeling Templates
+ Add Tag
Resources Center
Resources
PHP
LEARN
I'M JUST EXPLORING
Get Started with Atlas
Developer Center
Ask the MongoDB Community
New On Atlas 4 NEW
Learn about the latest feature enhancements on Atlas.
System Status: All Good
©2023 MongoDB, Inc. Status Terms Privacy Atlas Blog Contact Sales

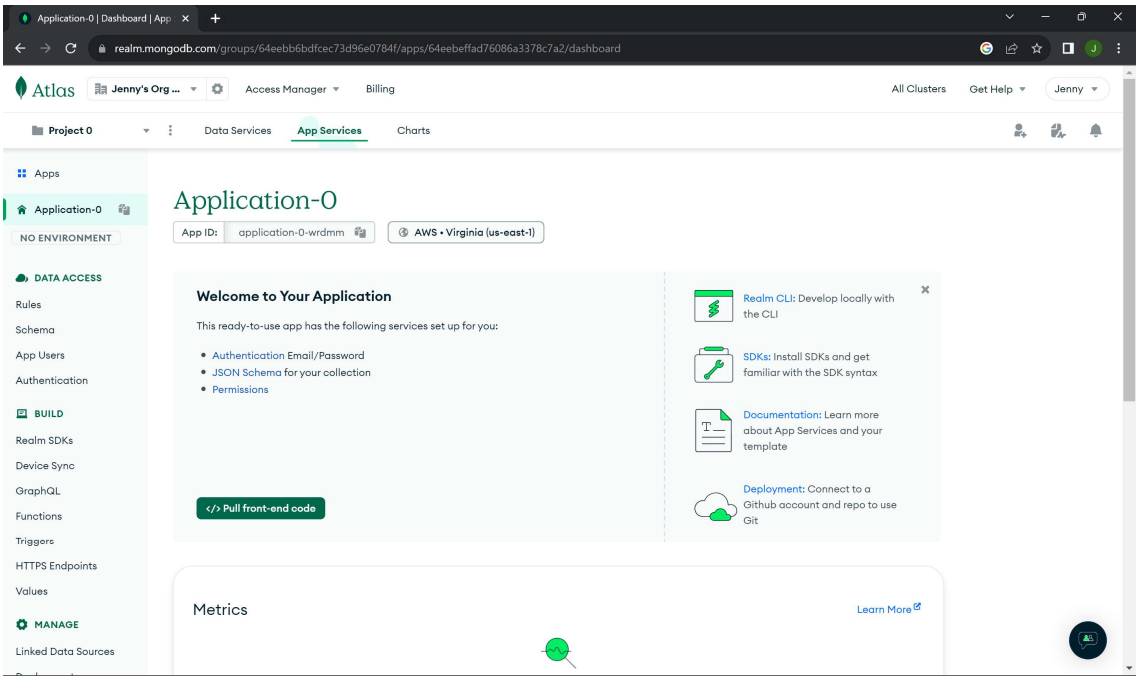

Database Deployments | Cloud:
cloud.mongodb.com/v2/64eebb6bdfcec73d96e0784f#/clusters
Atlas
Jenny's Org ...
Access Manager
Billing
All Clusters
Get Help
Jenny
Project 0
Data Services
App Services
Charts
Overview
DEPLOYMENT
Database
Data Lake
SERVICES
Device Sync
Triggers
Data API
Data Federation
Search
Stream Processing
SECURITY
Backup
Database Access
Network Access
Advanced
New On Atlas 4
Goto
JENNY'S ORG - 2023-08-30 > PROJECT 0
Database Deployments
Find a database deployment...
Edit Config
+ Create
Cluster0
Connect
View Monitoring
Browse Collections
FREE
SHARED
Visualize Your Data
Build dashboards and charts, and embed them in your apps with MongoDB Charts.
Dismiss
Explore Charts
R 0
W 0
Last 42 seconds
Connections 0
Last 42 seconds
In 0.0 B/s
Out 0.0 B/s
Last 42 seconds
Data Size
0.0 B / 512.0 MB (0%)
Last 42 seconds
VERSION
6.0.9
REGION
AWS / N. Virginia (us-east-1)
CLUSTER TIER
M0 Sandbox (General)
TYPE
Replica Set - 3 nodes
BACKUPS
Inactive
LINKED APP SERVICES
None Linked
ATLAS SQL
Connect
ATLAS SEARCH
Create Index
+ Add Tag

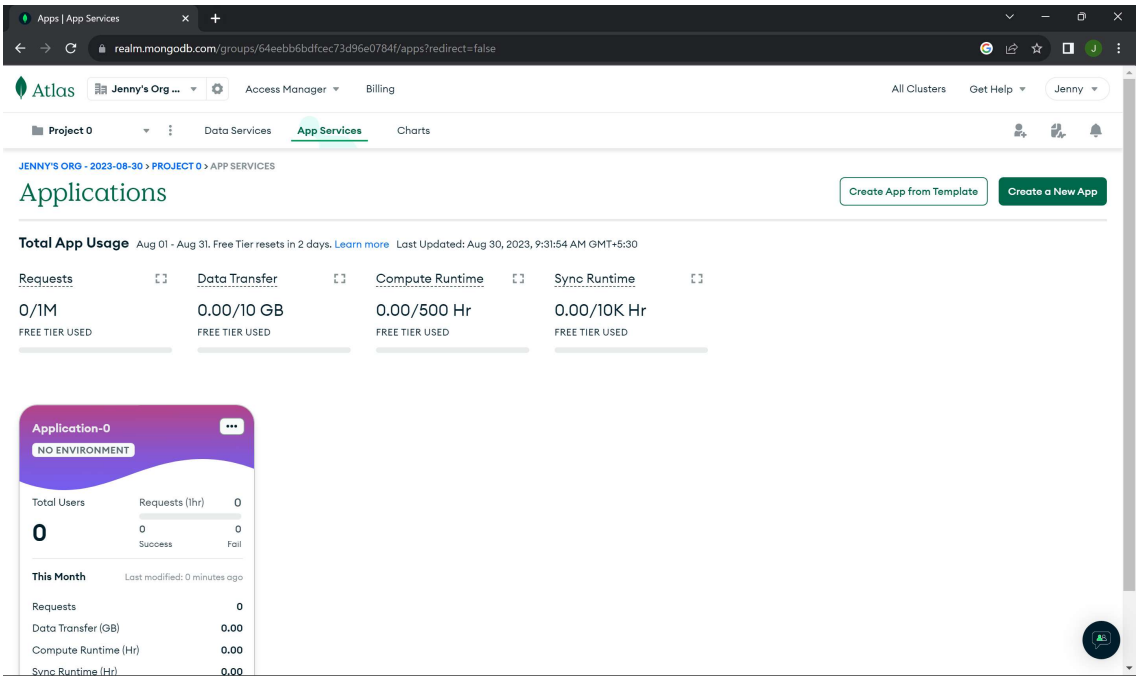

Database Deployments | Cloud:
cloud.mongodb.com/v2/64eebb6bdfcec73d96e0784f#/clusters/connect?clusterId=Cluster0
Atlas
Jenny's Org ...
All Clusters
Get Help
Jenny
Project 0
Overview
DEPLOYMENT
Database
Data Lake
SERVICES
Device Sync
Triggers
Data API
Data Federation
Search
Stream Processing
SECURITY
Backup
Database Access
Network Access
Advanced
New On Atlas
Goto
Edit Config
Create
FREE
SHARED
Data Size
0.0 B / 512.0 MB (0%)
Last 6 minutes
Connect to Cluster0
Set up connection security
Choose a connection method
Connect
Connect to your application
Drivers
Access your Atlas data using MongoDB's native drivers (e.g. Node.js, Go, etc.)
Access your data through tools
Compass
Explore, modify, and visualize your data with MongoDB's GUI
Shell
Quickly add & update data using MongoDB's Javascript command-line interface
MongoDB for VS Code
Work with your data in MongoDB directly from your VS Code environment
Atlas SQL
Easily connect SQL tools to Atlas for data analysis and visualization


Database Deployments | Cloud:
cloud.mongodb.com/v2/64eebb6bdfcec73d96e0784f#/clusters/connect?clusterId=Cluster0
Atlas
Jenny's Org ...
All Clusters
Get Help
Jenny
Project 0
Overview
DEPLOYMENT
Database
Data Lake
SERVICES
Device Sync
Triggers
Data API
Data Federation
Search
Stream Processing
SECURITY
Backup
Database Access
Network Access
Advanced
New On Atlas
Goto
Edit Config
Create
FREE
SHARED
Data Size
0.0 B / 512.0 MB (0%)
Last 6 minutes
Connect to Cluster0
Set up connection security
Choose a connection method
Connect
Connecting with MongoDB Driver
1. Select your driver and version
We recommend installing and using the latest driver version.
Driver
Node.js
Version
5.5 or later
2. Install your driver
Run the following on the command line
npm install mongodb
View MongoDB Node.js Driver installation instructions.
3. Add your connection string into your application code
View full code sample
mongodb+srv://jennyjohnson:<password>@cluster0.ifbwxzr.mongodb.net/?
retryWrites=true&w=majority
Replace <password> with the password for the jennyjohnson user. Ensure any option params are URL encoded.


App Services
realm.mongodb.com/groups/64eebb6bdfcec73d96e0784f/apps
AtlasApp
NO ENVIRONMENT
DATA ACCESS
Rules
Schema
App Users
Authentication
BUILD
Realm SDKs
Device Sync
GraphQL
Functions
Triggers
HTTPS Endpoints
Values
MANAGE
Linked Data Sources
Deployment
Hosting
Logs
Build better apps faster with App Services
Seamlessly connect your data to mobile apps, websites, and other services with our fully managed services and APIs.
Learn more about App Services
Start with an app template
Using App Services reduces the code you need to write. Get started quicker with Templates.
Build your own App
Set up your own app services to fit your development needs
Real-time Sync
Mobile app using a Realm SDK to sync data to your backend
Manage Database Views
Event-driven Database Trigger template to update a view in a separate collection
React.JS + Realm Web Boilerplate
Todo web app using the Realm Web SDK
GraphQL + React App Boilerplate
Todo web app using the GraphQL API


App Services
realm.mongodb.com/groups/64eebb6bdfcec73d96e0784f/apps
NO ENVIRONMENT
DATA ACCESS
Rules
Schema
App Users
Authentication
BUILD
Realm SDKs
Device Sync
GraphQL
Functions
Triggers
HTTPS Endpoints
Values
MANAGE
Linked Data Sources
Deployment
Hosting
Logs
App Settings
Build better apps faster with App Services
Seamlessly connect your data to mobile apps, websites, and other services with our fully managed services and APIs.
Learn more about App Services
1 Select a Template
Using App Services reduces the code you need to write. Get started quicker with Templates.
React.JS + Realm Web Boilerplate
2 Link your Data Source
Template Starter Applications let you build on top of data in a MongoDB Atlas cluster.
Cluster0
3 Name your Application
This name will be used internally and cannot be changed later.
Application-0
4 App Deployment Model
You can change this later in your Application Settings. View Docs
Single Region • Virginia (us-east-1) • AWS | Region chosen based on your cluster's region Change




## Result

The program was executed and the result was successfully obtained. Thus, CO6 was obtained.