```
Program No: 10
Date: 14-01-2022
                          AIM: FUNCTIONS
PROGRAMS
Numeric Functions
  1. Abs(-5)
Query:
mysql> select abs(-5);
Output:
| abs(-5) |
+----+
   5 |
  1 row in set (0.00 sec)
  2. Cos (0)
Query:
mysql> select cos(0);
Output:
+----+
| cos(0) |
+----+
       1 |
+----+
1 row in set (0.00 sec)
3. Sin(0)
Query:
mysql> select sin(0);
```

```
+----+
| sin(0) |
+----+
| 0 |
+----+
1 row in set (0.00 sec)
```

4. Ceiling(90.9)

Query:

mysql> select ceiling(90.9);

Output:

```
+-----+
| ceiling(90.9) |
+-----+
| 91 |
+-----+
1 row in set (0.00 sec)
```

5. Floor(90.9)

Query:

mysql> select floor(90.9);

Output:

```
+-----+
| floor(90.9) |
+-----+
| 90 |
+-----+
1 row in set (0.00 sec)
```

6. Truncate(1.83579,3)

Query:

mysql> select truncate(1.83579,3);

Output:

```
| truncate(1.83579,3) |

+-----+

| 1.835 |

+-----+

1 row in set (0.00 sec)
```

7. Mod(31,3)

Query:

mysql> select mod(31,3);

Output:

```
+----+
| mod(31,3) |
+----+
| 1 |
+----+
1 row in set (0.00 sec)
```

8. Power(2,3)

Query:

mysql> select power(2,3);

Output:

```
+-----+
| power(2,3) |
+-----+
| 8 |
+-----+
1 row in set (0.00 sec)
```

9. Exp(2)

Query:

mysql > select exp(2);

10. Round(5)

Query:

mysql> select round(5);

Output

```
+-----+
| round(5) |
+-----+
| 5 |
+-----+
1 row in set (0.00 sec)
```

11. Sqrt(25)

Query:

mysql> select sqrt(25);

Output:

```
+-----+
| sqrt(25) |
+-----+
| 5 |
+-----+
1 row in set (0.00 sec)
```

12. greatest(99,56)

Query:

mysql> select greatest(99,56);

13. least(99,56)

Query:

mysql> select least(99,56);

Output:

```
+-----+
| least(99,56) |
+-----+
| 56 |
+-----+
1 row in set (0.00 sec)
```

14. bin(12)

Query:

mysql> select bin(12);

Output:

```
+----+
| bin(12) |
+-----+
| 1100 |
+-----+
1 row in set (0.01 sec)
```

15. oct(8)

Query:

mysql> select oct(8);

16. hex(13)

Query:

mysql> select hex(13);

Output:

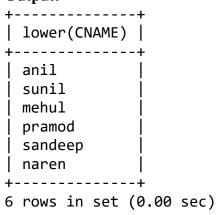
STRING FUNCTIONS

1. Display the customer names in deposit table in lower case

Query:

mysql> select lower(CNAME) from deposit;

Output:

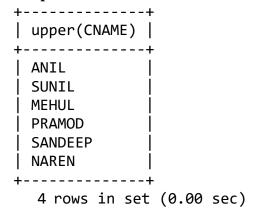


2. Display customer names in deposit table in uppercase

Query:

mysql> select upper(CNAME) from deposit;

Output:



3. Display the customer name and length of each name.

Query:

mysql> select CNAME,length(CNAME) from deposit;

Output:

+	++ length(CNAME)
Anil Sunil Mehul Pramod Sandeep Naren	4 5 5 6 7 5
5 rows	in set (0.00 sec)

4. Find the length of a string in bits.

Query

mysql> select CNAME,bit_length(CNAME) from deposit;

+	
CNAME	bit_length(CNAME)
+	++ 32 40 40 48 56
+	++

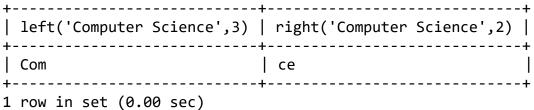
6 rows in set (0.00 sec)

5. Extract a specific no. of characters from the left and right of a particular string.

Query:

mysql> select left('Computer Science',3),right('Computer Science',2);

Output:



·

6. Concatenate two strings

Query:

mysql> select concat('Sun','Flower');

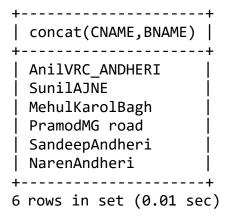
Output:

```
+-----+
| concat('Sun','Flower') |
+-----+
| SunFlower |
+-----+
1 row in set (0.01 sec)
```

7. Concatenate customer name and branch name from the table deposit

Query:

mysql> select concat(CNAME,BNAME) from deposit;



8. Concatenate two strings with comma as the separator.

Query:

```
mysql> select concat_ws(',','Sun','Flower');
```

Output:

9. Find the ascii value of a character.

Query:

```
mysql> select ascii('Anu');
```

Output:

```
+-----+
| ascii('Anu') |
+-----+
| 65 |
+-----+
1 row in set (0.01 sec)
```

10. Compare two strings.

Query:

mysql> select strcmp('computer','comp');

```
+-----+

| Cmp_Value |

+-----+

| 1 |

+-----+

1 row in set (0.00 sec)
```

11. Reverse a string.

Query:

mysql> select reverse('APPLE') as Reverse;

Output:

```
+-----+
| Reverse |
+-----+
| ELPPA |
+-----+
1 row in set (0.00 sec)
```

12. Extract 5 characters from the string "hello world" starting with 'l'.

Query:

mysql> select substring('hello world',1,5) as Extract;

Output:

```
+-----+

| Extract |

+-----+

| hello |

+-----+

1 row in set (0.00 sec)
```

13. Find the location of the substring "come" in the string "welcome".

Query:

mysql> select locate('come', 'welcome') as Location;

```
+-----+
| Location |
+-----+
| 4 |
+-----+
1 row in set (0.01 sec)
```

14. Find the position of a particular string in a set of strings.

Query:

mysql> select position('world' in 'welcome to the world of IT') as Position;

Output:

```
+-----+
| Position |
+-----+
| 16 |
+-----+
1 row in set (0.00 sec)
```

15. translate()

Query:

mysql> select replace('Monday','Mon','Tues');

Output:

16. Replace(), insert()

Query:

mysql> select replace('Two','w','o') as Translate,insert('Two fun',2,3,'oo') as Inserted;

```
| Translate | Inserted | +-----+ | Too | Toofun | +-----+ | Tow in set (0.00 sec)
```

17. rpad(), lpad()

Query:

mysql> select lpad('Flower',10,'Sun'),rpad('Flower',10,'Sun');

Output:

18. ltrim(),rtrim()

Query:

mysql> select ltrim(' Sun Flower '),rtrim(' Sun Flower ');

Output:

```
+------+
| ltrim(' Sun Flower ') | rtrim(' Sun Flower ') |
+------+
| Sun Flower | Sun Flower |
+-----+
1 row in set (0.00 sec)
```

19. Trim()

Query:

mysql> select trim(' Sun Flower ');

DATE FUNCTIONS

1. curtime(), curdate()

Query:

mysql> select curdate(),curtime();

Output:

```
+------+
| curdate() | curtime() |
+-----+
| 2022-02-1 | 12:23:43 |
+-----+
1 row in set (0.00 sec)
```

2. year()

Query:

mysql> select year('2022/2/7');

Output:

```
+-----+
| year('2022/2/7') |
+-----+
| 2022 |
+-----+
1 row in set (0.00 sec)
```

3. month(), monthname()

Query:

mysql> select month('2022/2/7'),monthname('2022/2/7');

```
+-----+
| month('2022/2/7') | monthname('2022/2/7') |
+-----+
| 2 | February |
+-----+
1 row in set (0.00 sec)
```

4. dayofyear(), dayofmonth(), dayofweek()

Query:

mysql> select dayofyear('2022/2/7'),dayofmonth('2022/2/7'),dayofweek('2022/2/7');

Output

```
+------+
| dayofyear('2022/2/7') | dayofmonth('2022/2/7') | dayofweek('2022/2/7')|
+------+
| 38 | 7 | 2|
+-----+
1 row in set (0.00 sec)
```

5. hour(), minute(), second()

Query:

mysql> select hour('12:23:43'),minute('12:23:43'),second('12:23:43');

Output:

6. to_days()

Query:

mysql> select to_days('2022/2/7');

```
| to_days('2022/2/7') |
+-----+
| 738558|
+-----+
1 row in set (0.00 sec)
```

7. from_days()

Query:

mysql> select from_days('2022/2/7');

Output:

```
+-----+

| from_days('2022/2/7') |

+-----+

| 0005-07-15 |

+-----+

1 row in set, 1 warning (0.00 sec)
```

8. date_add(), date_sub()

Query:

mysql> select date_add('2022/02/16',interval 10 day) as date_add,date_sub('2022/02/16',interval 10 day) as date_sub;

Output:

```
+------+
| date_add | date_sub |
+-----+
| 2022-02-26 | 2022-02-06 |
+-----+
1 row in set (0.00 sec)
```

9. extract()

Query:

mysql> select extract(month from '2022/02/16') as extract;

```
+----+

| extract |

+----+

| 2 |

+----+

1 row in set (0.00 sec)
```

10. period_diff()

Query

mysql> select period_diff('1999/09/25','2022/09/25') as period_diff;

Output:

```
+-----+
| period_diff |
+-----+
| 65 |
+-----+
1 row in set, 2 warnings (0.00 sec)
```

Program No: 11

Date: 16/02/2022

SUB-QUERIES

PROGRAMS

1. Insert into deposit a new record (107,Pradeep,Ajne,2000,12032011). Display the customers' names whose branch is same as that of the branch of Sandeep.

Query:

mysql>insertinto deposit(ACCNO,CNAME,BNAME,AMT,DDATE)values(107,'Pradeep','Ajne', 2000,'2011/03/12');

Query OK, 1 row affected (0.01 sec)

mysql> select * from deposit;

Output:

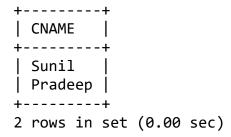
	++				
	ACCNO	CNAME	BNAME	AMT	DDATE
-	1001 1002 1003 1006 1007 1008 107	Anil Sunil Mehul Pramod Sandeep Naren Pradeep	VRC_ANDHERI AJNE KarolBagh MG road Andheri Andheri AJNE	20000 57500 40250 37950 25300 37950 2000	2014-03-01 2015-01-04 2014-11-17 2015-03-27 2015-03-31 2014-02-12 2011-03-12
-	++				+

7 rows in set (0.00 sec)

Query

 $mysql> \ select \ CNAME \ from \ deposit \ where \ BNAME \ in(select \ BNAME \ from \ deposit \ where \ CNAME='Pradeep');$

Output

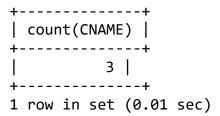


2. Give the number of customers who are depositors as well as borrowers.

Query:

mysql> select count(CNAME) from deposit where CNAME in(select CNAME from borrow);

Output:



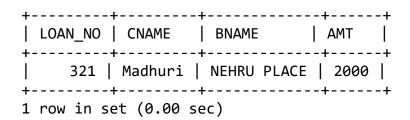
3. Delete the record from loans who have a deposit.

Query:

mysql> delete from borrow where CNAME in(select CNAME from deposit); Query OK, 3 rows affected (0.00 sec)

mysql> select * from borrow;

Output



Program No: 12

Date: 16/02/2022

VIEWS IN DBMS

PROGRAMS

1. Create a view for the table deposit with fields accno, cust name and branch name

Query:

mysql> create view view_deposit as select ACCNO,CNAME,BNAME from deposit; Query OK, 0 rows affected (0.01 sec)

Output:

mysql> desc view_deposit;

+ Field +	Туре	+ Null	+ Key	+ Default	+ Extra
ACCNO CNAME	int(15) varchar(50) varchar(50)	YES	j j	NULL NULL NULL	+

2 rows in set (0.03 sec)

mysql> select *from view deposit;

4			L -
į	ACCNO	CNAME	BNAME
+	1001 1002 1003 1006 1007 1008 107	Anil Sunil Mehul Pramod Sandeep Naren Pradeep	VRC_ANDHERI AJNE KarolBagh MG road Andheri Andheri AJNE
-			-

7 rows in set (0.00 sec)

2. Create a view that gives information about customer name, deposit amount and living city

Query:

mysql> create view view_customer as select deposit.CNAME,deposit.AMT,customer.LCITY from deposit,customer where deposit.CNAME=customer.CNAME;

Query OK, 0 rows affected (0.00 sec)

mysql> select * from view_customer;

+	+ AMT	++ LCITY
Anil Sunil Mehul Pramod Naren	20000 57500 40250 37950 37950	Calcutta Delhi Baroda Nagpur Bombay
5 rows in	set (0.0	02 sec)

3. Create a view to display the borrow details having loan no 206

Query:

mysql> create view view_borrow as select * from borrow where LOAN_NO='206'; Query OK, 0 rows affected (0.00 sec)

Output:

mysql> select * from view_borrow;

4. Drop a view

Query:

mysql> drop view view_borrow;

Output:

```
Query OK, 0 rows affected (0.02 sec)
```

```
mysql> select *from view_borrow;
ERROR 1146 (42S02): Table 'anuja.view_borrow' doesn't exist
```

Program No: 13

Date: 25/03/2022

PERFORM JOIN OPERATIONS

PROGRAMS

a. Create the tables and insert the following data into the tables.

<u>Salesman</u>

+	-+	+	-+	+
salesman_id	•	-	•	ommission
+	-+	+	-+	+
5001	james hoog	new york		0.15
5002	Nail knite	paris		0.13
5005	pit alex	london		0.11
5006	mc lyon	paris		0.14
5007	paul adam	rome		0.13
5003	lauson hen	san jose		0.12
+	-+	+	-+	+

<u>Customer</u>

+	 cust_name	, ,		•	+ sman_id
3002	nick rimando	new york	i I	100	5001
3007	brad davis	new york		200	5001
3005	graham zusi	california		200	5002
3008	julian green	london		300	5002
3004	fabian johnson	paris		300	5006
3009	geoff cameron	berlin		100	5003
3003	jozy altidor	moscow		200	5007
3001	brad guzan	london		300	5005
+		+	+	+-	+

8 rows in set (0.01 sec)

<u>Orders</u>			
ord_no purch	_amt	customer_id	salesman_id
70009 270.6	5 2012-09-10	3001	5005
70002 65.26	2012-10-05	3002	5001
70004 110.5	2012-08-17	3009	5003
70007 948.5	2012-09-10	3005	5002
70005 2400.	6 2012-07-27	3007	5001
70008 5760	2012-09-10	3002	5001
70010 1983.	43 2012-10-10	3004	5006
70003 2480.	4 2010-10-10	3009	5003
70012 250.4	5 2012-06-27	3008	5002
70011 75.29	2012-08-17	3003	5007
70013 3045.	6 2012-04-25	3002	5001
+	+	-+	+

a. Find the names of salesmen and customer who belongs to the same city.

mysql> select salesman.name,customer.cust_name from salesman inner join customer on salesman.city=customer.city;

+	+
name	cust_name
james hoog pit alex nail knite	nick rimando brad davis julian green fabian johns fabian johns brad guzan
6 rows in set	(0.00 sec)

b. Display the purchase details of customers (Customer name, orderno, amount and city) in which the order amount between 100 and 1000.

mysql> select customer.cust_name,orders.ord_no,orders.purch_amt,customer.city from customer inner join orders on customer_id=orders.customer_id where orders.purch_amt>100 and orders.purch_amt<1000;

+				
cust_name	ord_no	purch_amt	city	
graham zusi brad guzan	70001 70009	:	california london	
geoff camero	70004	110.5	berlin	
graham zusi julian green	70007 70012	250.45	california london	
++				
5 rows in set (0.00 sec)				

c. Find the salesman appear for all customers and vice versa. (Hint: Cross join)

mysql> select name,cust_name from salesman cross join customer;

+	
name	cust_name
<pre> james hoog nail knite pit alex mc lyon paul adam </pre>	nick rimando nick rimando nick rimando nick rimando nick rimando
lauson hen	nick rimando
james hoog	brad davis
nail knite	brad davis
pit alex	brad davis
mc lyon	brad davis
paul adam	brad davis
lauson hen	brad davis
james hoog	graham zusi
nail knite	graham zusi
pit alex	graham zusi
mc lyon	graham zusi
paul adam	graham zusi
lauson hen	graham zusi
james hoog	julian green
nail knite	julian green
pit alex	julian green
mc lyon	julian green
paul adam	julian green
lauson hen	julian green

```
| james hoog | fabian johns |
| nail knite | fabian johns |
| pit alex  | fabian johns |
| mc lyon | fabian johns |
| paul adam | fabian johns |
| lauson hen | fabian johns |
| james hoog | geoff camero |
| nail knite | geoff camero |
| pit alex | geoff camero |
mc lyon
             | geoff camero |
| paul adam | geoff camero |
| lauson hen | geoff camero |
| james hoog | jozy altidor |
| nail knite | jozy altidor |
| pit alex | jozy altidor |
mc lyon
             | jozy altidor |
| paul adam | jozy altidor |
| lauson hen | jozy altidor |
| james hoog | brad guzan
| nail knite | brad guzan
| pit alex | brad guzan
mc lyon | brad guzan
| paul adam | brad guzan
| lauson hen | brad guzan
48 rows in set (0.00 sec)
```

d. Display the details to know which salesman are working for which customer. (Hint: Inner Join)

mysql> select name as salesman,cust_name as customer from salesman inner join customer on salesman.salesman_id=customer.salesman_id;

++	+
salesman	:
james hoog james hoog nail knite nail knite mc lyon lauson hen	nick rimando brad davis graham zusi julian green fabian johns geoff camero jozy altidor

e. Find out the customers who holds a grade less than 200 and works either through a salesman or by own. Also display the customer names in sorted order. (Hint: Left Join)

mysql> select distinct cust_name as cust_name from salesman left join customer on salesman.salesman_id=customer.salesman_id where grade<200 order by cust_name;

f. Find the customer name, order number, order date, and order amount in descending order according to the order date to find that either any of the existing customers have placed no order or placed one or more orders.(Hint : left Outer Join)

mysql> select cust_name,ord_no,ord_date,purch_amt from orders left outer join customer on orders.customer_id=customer.customer_id order by ord_date desc;

cust_name	+			
fabian johns 70010 2012-10-10 1983.43 geoff camero 70003 2012-10-10 2480.4 graham zusi 70001 2012-10-05 150.5 nick rimando 70002 2012-10-05 65.26 brad guzan 70009 2012-09-10 270.65 graham zusi 70007 2012-09-10 948.5 nick rimando 70008 2012-09-10 5760 jozy altidor 70011 2012-08-17 75.29 geoff camero 70004 2012-08-17 110.5 brad davis 70005 2012-07-27 2400.6 julian green 70012 2012-06-27 250.45	cust_name	. – .		purch_amt
nick rimando 70013 2012-04-25 3045.6	geoff camero graham zusi nick rimando brad guzan graham zusi nick rimando jozy altidor geoff camero brad davis julian green	70010 70003 70001 70002 70009 70007 70008 700011 70004 70005 70012	2012-10-10 2012-10-10 2012-10-05 2012-10-05 2012-09-10 2012-09-10 2012-09-17 2012-08-17 2012-07-27 2012-06-27	2480.4 150.5 65.26 270.65 948.5 5760 75.29 110.5 2400.6 250.45
	+	, , , , , , , , , , , , , , , , , , ,	+	+

Program No: 14

Date: 25/03/2022

PERFORM THE FOLLOWING QUERIES IN MONGODB

PROGRAMS

a. Create a database "mydb" and display the current database that you are accessing

```
use mydbswitched to db mydbdbmydb
```

b. Display all the databases in the MongoDB shell

> show dbs local 0.03125GB whitecloud 0.0625GB

c. Create a collection Employee and insert the following document into the collection Employee

Key Value		
Name William		
Age 35		
Department Finance		

Salary 75000/-

```
> db.createCollection("Employee")
{ "ok" : 1 }
>db.Employee.insert({Name:"William",Age:35,Department:"Finance",Salary:75000})
> db.Employee.find()
{ "_id" : ObjectId("6231869a0bbcac00191f602a"), "Name" : "William", "Age" : 35,
```

"Department": "Finance", "Salary": 75000 }

d. To the same collection add the following multiple documents

Key Value
Name Adam
Age 29
Department Marketing
Salary 50,000/-

```
Name Jerry

Age 25

Department Finance

Salary 38,000/-
```

```
}
            {
                   "_id": ObjectId("623188f00bbcac00191f602b"),
                   "Name": "Adam",
                   "Age" : 29,
                   "Department": "Marketing",
                   "Salary" : 50000
            }
            {
                   "_id": ObjectId("623188f00bbcac00191f602c"),
                   "Name": "Jerry",
                   "Age": 25,
                   "Department": "Finance",
                   "Salary" : 38000
            }
e. Display a single document from the collection Employee.
> db.Employee.findOne()
      "_id": ObjectId("6231869a0bbcac00191f602a"),
      "Name": "William",
      "Age": 35,
      "Department": "Finance",
      "Salary": 75000
}
f. Display all the documents from the collection in a formatted way.
> db.Employee.find().pretty()
      "_id": ObjectId("6231869a0bbcac00191f602a"),
      "Name": "William",
      "Age": 35,
      "Department": "Finance",
      "Salary": 75000
}
      "_id": ObjectId("623188f00bbcac00191f602b"),
      "Name": "Adam",
      "Age": 29,
      "Department": "Marketing",
      "Salary": 50000
}
```

```
"_id": ObjectId("623188f00bbcac00191f602c"),
      "Name": "Jerry",
      "Age": 25,
      "Department": "Finance",
      "Salary": 38000
}
g. Update the Name "Adam" to "Robin" from the collection employee.
> db.Employee.update({Name:"Adam"},{$set:{Name:"Robin"}})
> db.Employee.find().pretty()
      "_id": ObjectId("6231869a0bbcac00191f602a"),
      "Name": "William",
      "Age": 35,
      "Department": "Finance",
      "Salary": 75000
}
      "Age": 29,
      "Department": "Marketing",
      "Name": "Robin",
      "Salary": 50000,
      "_id" : ObjectId("623188f00bbcac00191f602b")
}
      "_id": ObjectId("623188f00bbcac00191f602c"),
      "Name": "Jerry",
      "Age": 25,
      "Department": "Finance",
      "Salary" : 38000
}
h. Update the department of William & Jerry to "Operations" using
appropriate parameter
>db.Employee.update({Department:"Finance"},{$set:{Department:"Operations"}},{multi:true})
> db.Employee.find().pretty()
      "Age": 35,
```

```
"Department": "Operations",
      "Name": "William",
      "Salary": 75000,
      "_id" : ObjectId("6231869a0bbcac00191f602a")
}
      "Age": 29,
      "Department": "Marketing",
      "Name": "Robin",
      "Salary": 50000,
      "_id": ObjectId("623188f00bbcac00191f602b")
}
{
      "Age": 25,
      "Department": "Operations",
      "Name": "Jerry",
      "Salary": 38000,
      "_id" : ObjectId("623188f00bbcac00191f602c")
}
i. Update the Robin's Age to 30, Department to Sales.
>db.Employee.update({Name:"Robin"},{$set:{Age:30,Department:"Sales"}},{m
ulti:true})
> db.Employee.find().pretty()
      "Age": 35,
      "Department": "Operations",
      "Name": "William",
      "Salary": 75000,
      "_id": ObjectId("6231869a0bbcac00191f602a")
}
      "Age": 30,
      "Department": "Sales",
      "Name": "Robin",
      "Salary": 50000,
      "_id" : ObjectId("623188f00bbcac00191f602b")
      "Age": 25,
      "Department": "Operations",
      "Name": "Jerry",
```

```
"Salary" : 38000,
"_id" : ObjectId("623188f00bbcac00191f602c")
}
```

j. Without displaying ObjectId, display only the name and age of the employees in formatted way using projection

```
> db.Employee.find({},{Name:1,Age:1,_id:0}).pretty()
{ "Age" : 35, "Name" : "William" }
{ "Age" : 30, "Name" : "Robin" }
{ "Age" : 25, "Name" : "Jerry" }
```

k. Sort the name and salary of the employee in ascending and descending order

```
> db.Employee.find().sort({Name:1}).pretty()
      "Age": 25,
      "Department": "Operations",
      "Name": "Jerry",
      "Salary": 38000,
      "_id": ObjectId("623188f00bbcac00191f602c")
            "Age": 30,
            "Department": "Sales",
            "Name": "Robin",
            "Salary": 50000,
            "_id": ObjectId("623188f00bbcac00191f602b")
      }
         "Age": 35,
         "Department": "Operations",
        "Name": "William",
        "Salary": 75000,
        "_id": ObjectId("6231869a0bbcac00191f602a")
> db.Employee.find().sort({Salary:-1}).pretty()
         "Age": 35,
         "Department": "Operations",
```

```
"Name": "William",
        "Salary": 75000,
        "_id": ObjectId("6231869a0bbcac00191f602a")
}
        "Age": 30,
        "Department": "Sales",
        "Name": "Robin",
        "Salary": 50000,
        "_id": ObjectId("623188f00bbcac00191f602b")
}
        "Age": 25,
        "Department": "Operations",
        "Name": "Jerry",
        "Salary": 38000,
        "_id" : ObjectId("623188f00bbcac00191f602c")
}
```

$\label{lem:conding} \textbf{l. Create indexes for the fields name} (ascending) \ and \ salary (descending) \ of \ the employee$

```
> db.Employee.createIndex({Name:1,Salary:-1})
```

m. Display the description of all the indexes in the collection.

```
},
"ns": "mydb.Employee",
"name": "Name_1_Salary_-1"
}
```

n. Delete all the indexes from the collection employee.

```
> db.Employee.dropIndexes()
{
     "nIndexesWas" : 2,
     "msg" : "non-_id indexes dropped for collection",
     "ok" : 1
}
```

o. Remove the respective documents and collection from the MongoDB p. Delete the database 'mydb'.

```
> db.Employee.remove({})
```