

DATABASE ASSIGNMENT

1. To list all records with sal > 2000 and comm>200

```
mysql> select * from emp where
-> sal>2000 and comm>200;
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|---------|----------|------|------------|---------|---------|--------|
| 1000 | Krishna | Manager | NULL | 2024-03-03 | 5095.82 | 400.00 | 30 |
| 7499 | ALLEN | SALESMAN | 7698 | 1981-02-20 | 3656.26 | 300.00 | 30 |
| 7521 | WARD | SALESMAN | 7698 | 1981-02-22 | 3638.67 | 500.00 | 30 |
| 7654 | MARTIN | SALESMAN | 7698 | 1981-09-28 | 3638.67 | 1400.00 | 30 |

4 rows in set (0.02 sec)

2. To list all record with job='Clerk' or sal>2000

```
mysql> select * from emp
-> where job="Clerk" or sal>2000;
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|---------|-----------|------|------------|----------|---------|--------|
| 1000 | Krishna | Manager | NULL | 2024-03-03 | 5095.82 | 400.00 | 30 |
| 1001 | Raj | Clerk | NULL | 2024-01-07 | 6738.29 | 100.00 | 30 |
| 1002 | Aditya | Clerk | NULL | 2021-06-01 | 6738.29 | 200.00 | 30 |
| 1111 | Vihan | Manager | NULL | 2019-08-23 | 5371.10 | NULL | NULL |
| 7369 | SMITH | SenClerk | 7902 | 1980-12-17 | 2250.00 | NULL | 20 |
| 7499 | ALLEN | SALESMAN | 7698 | 1981-02-20 | 3656.26 | 300.00 | 30 |
| 7521 | WARD | SALESMAN | 7698 | 1981-02-22 | 3638.67 | 500.00 | 30 |
| 7566 | JONES | MANAGER | 7839 | 1981-04-02 | 7669.92 | NULL | 20 |
| 7654 | MARTIN | SALESMAN | 7698 | 1981-09-28 | 3638.67 | 1400.00 | 30 |
| 7698 | BLAKE | MANAGER | 7839 | 1981-05-01 | 7347.66 | NULL | 30 |
| 7782 | CLARK | MANAGER | 7839 | 1981-06-09 | 6316.40 | NULL | 10 |
| 7788 | SCOTT | ANALYST | 7566 | 1982-12-09 | 8789.06 | NULL | 20 |
| 7839 | KING | PRESIDENT | NULL | 1981-11-17 | 12656.26 | NULL | 10 |
| 7844 | TURNER | SALESMAN | 7698 | 1981-09-08 | 4366.40 | 0.00 | 30 |
| 7876 | ADAMS | CLERK | 7788 | 1983-01-12 | 3093.74 | NULL | 20 |
| 7900 | JAMES | CLERK | 7698 | 1981-12-03 | 2671.87 | NULL | 30 |
| 7902 | FORD | ANALYST | 7566 | 1981-12-03 | 8789.06 | NULL | 20 |
| 7934 | MILLER | CLERK | 7782 | 1982-01-23 | 3656.26 | NULL | 10 |

18 rows in set (0.00 sec)

3. To list all the record with sal=1250 or 1100 or 2850

```
MySQL 8.0 Command Line Client - Unicode
18 rows in set (0.00 sec)

mysql> select * from emp where sal in (1250,1100,2850);
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|--------|----------|------|------------|---------|---------|--------|
| 7521 | WARD | SALESMAN | 7698 | 1981-02-22 | 1250.00 | 500.00 | 30 |
| 7654 | MARTIN | SALESMAN | 7698 | 1981-09-28 | 1250.00 | 1400.00 | 30 |
| 7698 | BLAKE | MANAGER | 7839 | 1981-05-01 | 2850.00 | NULL | 30 |
| 7876 | ADAMS | CLERK | 7788 | 1983-01-12 | 1100.00 | NULL | 20 |

4 rows in set (0.00 sec)

4. To list all employees with sal>1250 and <2850

```
mysql> select * from emp
-> where sal between 1250 and 2850;
+-----+-----+-----+-----+-----+-----+-----+-----+
| EMPNO | ENAME | JOB      | MGR | HIREDATE   | SAL      | COMM | DEPTNO |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 7369 | SMITH | SenClerk | 7902 | 1980-12-17 | 2250.00 | NULL | 20 |
| 7900 | JAMES | CLERK    | 7698 | 1981-12-03 | 2671.87 | NULL | 30 |
+-----+-----+-----+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)
```

5. To list all employees with name ends with AS

```
mysql> select * from emp where ename REGEXP '.*AS$';
Empty set (0.00 sec)

mysql> select * from emp where ename like '%AS';
Empty set (0.00 sec)
```

6. 6. To list all employees with job starts with C and ends with K

```
mysql> select * from emp
-> where job REGEXP "^C.*K$";
+-----+-----+-----+-----+-----+-----+-----+-----+
| EMPNO | ENAME | JOB      | MGR | HIREDATE   | SAL      | COMM | DEPTNO |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 1001 | Raj   | Clerk    | NULL | 2024-01-07 | 6738.29 | 100.00 | 30 |
| 1002 | Aditya | Clerk    | NULL | 2021-06-01 | 6738.29 | 200.00 | 30 |
| 7876 | ADAMS | CLERK    | 7788 | 1983-01-12 | 3093.74 | NULL | 20 |
| 7900 | JAMES | CLERK    | 7698 | 1981-12-03 | 2671.87 | NULL | 30 |
| 7934 | MILLER | CLERK    | 7782 | 1982-01-23 | 3656.26 | NULL | 10 |
+-----+-----+-----+-----+-----+-----+-----+-----+
5 rows in set (0.04 sec)
```

7. To list all employees with job contains L at third position and M at third last position

```
mysql> select * from emp where job REGEXP '^C.*K$';
+-----+-----+-----+-----+-----+-----+-----+-----+
| EMPNO | ENAME | JOB      | MGR | HIREDATE   | SAL      | COMM | DEPTNO |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 7369 | SMITH | CLERK    | 7902 | 1980-12-17 | 800.00 | NULL | 20 |
| 7876 | ADAMS | CLERK    | 7788 | 1983-01-12 | 1100.00 | NULL | 20 |
| 7900 | JAMES | CLERK    | 7698 | 1981-12-03 | 950.00 | NULL | 30 |
| 7934 | MILLER | CLERK    | 7782 | 1982-01-23 | 1300.00 | NULL | 10 |
+-----+-----+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

8. To list all the record with sal not equal to 1250 or 1100 or 2850

```
7 rows in set (0.00 sec)

mysql> select * from emp where sal not in (1250,1100,2850);
+-----+-----+-----+-----+-----+-----+-----+-----+
| EMPNO | ENAME  | JOB      | MGR  | HIREDATE | SAL    | COMM  | DEPTNO |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 7369  | SMITH  | CLERK    | 7902 | 1980-12-17 | 800.00 | NULL  | 20     |
| 7499  | ALLEN  | SALESMAN | 7698 | 1981-02-20 | 1600.00 | 300.00 | 30     |
| 7566  | JONES  | MANAGER  | 7839 | 1981-04-02 | 2975.00 | NULL  | 20     |
| 7782  | CLARK  | MANAGER  | 7839 | 1981-06-09 | 2450.00 | NULL  | 10     |
| 7788  | SCOTT  | ANALYST  | 7566 | 1982-12-09 | 3000.00 | NULL  | 20     |
| 7839  | KING   | PRESIDENT | NULL | 1981-11-17 | 5000.00 | NULL  | 10     |
| 7844  | TURNER | SALESMAN | 7698 | 1981-09-08 | 1500.00 | 0.00  | 30     |
| 7900  | JAMES  | CLERK    | 7698 | 1981-12-03 | 950.00  | NULL  | 30     |
| 7902  | FORD   | ANALYST  | 7566 | 1981-12-03 | 3000.00 | NULL  | 20     |
| 7934  | MILLER | CLERK    | 7782 | 1982-01-23 | 1300.00 | NULL  | 10     |
+-----+-----+-----+-----+-----+-----+-----+-----+
10 rows in set (0.00 sec)
```

Activate Windows
Go to Settings to activate Windows.

9. To list all employees with salnot >1250 and <2850

```
mysql> select * from emp where sal not between 1250 and 2850;
+-----+-----+-----+-----+-----+-----+-----+-----+
| EMPNO | ENAME  | JOB      | MGR  | HIREDATE | SAL    | COMM  | DEPTNO |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 7369  | SMITH  | CLERK    | 7902 | 1980-12-17 | 800.00 | NULL  | 20     |
| 7566  | JONES  | MANAGER  | 7839 | 1981-04-02 | 2975.00 | NULL  | 20     |
| 7788  | SCOTT  | ANALYST  | 7566 | 1982-12-09 | 3000.00 | NULL  | 20     |
| 7839  | KING   | PRESIDENT | NULL | 1981-11-17 | 5000.00 | NULL  | 10     |
| 7876  | ADAMS  | CLERK    | 7788 | 1983-01-12 | 1100.00 | NULL  | 20     |
| 7900  | JAMES  | CLERK    | 7698 | 1981-12-03 | 950.00  | NULL  | 30     |
| 7902  | FORD   | ANALYST  | 7566 | 1981-12-03 | 3000.00 | NULL  | 20     |
+-----+-----+-----+-----+-----+-----+-----+-----+
7 rows in set (0.00 sec)
```

10. To list all employees with job starts with C , E at 3rd position and ends with K

```
mysql> select * from emp where job REGEXP '^C.E.*K$';
+-----+-----+-----+-----+-----+-----+-----+-----+
| EMPNO | ENAME  | JOB      | MGR  | HIREDATE | SAL    | COMM  | DEPTNO |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 7369  | SMITH  | CLERK    | 7902 | 1980-12-17 | 800.00 | NULL  | 20     |
| 7876  | ADAMS  | CLERK    | 7788 | 1983-01-12 | 1100.00 | NULL  | 20     |
| 7900  | JAMES  | CLERK    | 7698 | 1981-12-03 | 950.00  | NULL  | 30     |
| 7934  | MILLER | CLERK    | 7782 | 1982-01-23 | 1300.00 | NULL  | 10     |
+-----+-----+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

11. To list all rows with comm is null.

```
mysql> select * from emp where comm is null;
+-----+-----+-----+-----+-----+-----+-----+-----+
| EMPNO | ENAME  | JOB      | MGR  | HIREDATE | SAL    | COMM  | DEPTNO |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 7369  | SMITH  | CLERK    | 7902 | 1980-12-17 | 800.00 | NULL  | 20     |
| 7566  | JONES  | MANAGER  | 7839 | 1981-04-02 | 2975.00 | NULL  | 20     |
| 7698  | BLAKE  | MANAGER  | 7839 | 1981-05-01 | 2850.00 | NULL  | 30     |
| 7782  | CLARK  | MANAGER  | 7839 | 1981-06-09 | 2450.00 | NULL  | 10     |
| 7788  | SCOTT  | ANALYST  | 7566 | 1982-12-09 | 3000.00 | NULL  | 20     |
| 7839  | KING   | PRESIDENT | NULL | 1981-11-17 | 5000.00 | NULL  | 10     |
| 7876  | ADAMS  | CLERK    | 7788 | 1983-01-12 | 1100.00 | NULL  | 20     |
| 7900  | JAMES  | CLERK    | 7698 | 1981-12-03 | 950.00  | NULL  | 30     |
| 7902  | FORD   | ANALYST  | 7566 | 1981-12-03 | 3000.00 | NULL  | 20     |
| 7934  | MILLER | CLERK    | 7782 | 1982-01-23 | 1300.00 | NULL  | 10     |
+-----+-----+-----+-----+-----+-----+-----+-----+
10 rows in set (0.00 sec)
```

mysql>

Activate Windows
Go to Settings to activate Windows.

12. To list all employees with sal is null and name starts with 'S'

```
mysql> select sal,ename from emp where sal is null and ename like 'S%';
Empty set (0.00 sec)

mysql> select * from emp where length(job)=5;
```

13. To list all employees with job contains 5 characters.

```
mysql> select * from emp where length(job)=5;
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|--------|-------|------|------------|---------|------|--------|
| 7369 | SMITH | CLERK | 7902 | 1980-12-17 | 800.00 | NULL | 20 |
| 7876 | ADAMS | CLERK | 7788 | 1983-01-12 | 1100.00 | NULL | 20 |
| 7900 | JAMES | CLERK | 7698 | 1981-12-03 | 950.00 | NULL | 30 |
| 7934 | MILLER | CLERK | 7782 | 1982-01-23 | 1300.00 | NULL | 10 |

```
4 rows in set (0.00 sec)
```

14. To list all employees with name contain 'A' at 1 position and job Contains 5 characters.

```
mysql> select * from emp where ename like 'A%' and length(job)=5;
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|-------|-------|------|------------|---------|------|--------|
| 7876 | ADAMS | CLERK | 7788 | 1983-01-12 | 1100.00 | NULL | 20 |

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

Q2. Solve the following

1. Retrieve the details (Name, Salary and dept no) of the emp who are working in department code 20, 30 and 40.

```
mysql> select ename,sal,deptno from emp where deptno in (20,30,40);
```

| ename | sal | deptno |
|--------|---------|--------|
| SMITH | 800.00 | 20 |
| ALLEN | 1600.00 | 30 |
| WARD | 1250.00 | 30 |
| JONES | 2975.00 | 20 |
| MARTIN | 1250.00 | 30 |
| BLAKE | 2850.00 | 30 |
| SCOTT | 3000.00 | 20 |
| TURNER | 1500.00 | 30 |
| ADAMS | 1100.00 | 20 |
| JAMES | 950.00 | 30 |
| FORD | 3000.00 | 20 |

```
11 rows in set (0.00 sec)
```

2. Display the total salary of all employees . Total salary will be calculated as $sal + comm + sal * 0.10$

```
mysql> select empno,ename,sal,comm,sal+ifnull(comm,0)+sal*0.10 from emp;
```

| empno | ename | sal | comm | sal+ifnull(comm,0)+sal*0.10 |
|-------|--------|---------|---------|-----------------------------|
| 7369 | SMITH | 800.00 | NULL | 880.0000 |
| 7499 | ALLEN | 1600.00 | 300.00 | 2060.0000 |
| 7521 | WARD | 1250.00 | 500.00 | 1875.0000 |
| 7566 | JONES | 2975.00 | NULL | 3272.5000 |
| 7654 | MARTIN | 1250.00 | 1400.00 | 2775.0000 |
| 7698 | BLAKE | 2850.00 | NULL | 3135.0000 |
| 7782 | CLARK | 2450.00 | NULL | 2695.0000 |
| 7788 | SCOTT | 3000.00 | NULL | 3300.0000 |
| 7839 | KING | 5000.00 | NULL | 5500.0000 |
| 7844 | TURNER | 1500.00 | 0.00 | 1650.0000 |
| 7876 | ADAMS | 1100.00 | NULL | 1210.0000 |
| 7900 | JAMES | 950.00 | NULL | 1045.0000 |
| 7902 | FORD | 3000.00 | NULL | 3300.0000 |
| 7934 | MILLER | 1300.00 | NULL | 1430.0000 |

14 rows in set (0.00 sec)

4. List the empno, name, and department number of the emp works under manager with id 7698

```
mysql> select empno,ename,deptno from emp where mgr=7698;
```

| empno | ename | deptno |
|-------|--------|--------|
| 7499 | ALLEN | 30 |
| 7521 | WARD | 30 |
| 7654 | MARTIN | 30 |
| 7844 | TURNER | 30 |
| 7900 | JAMES | 30 |

5 rows in set (0.00 sec)

5. List the name, job, and salary of the emp who are working in departments 10 and 30.

```
mysql> select job,ename,sal from emp where deptno in (10,30);
```

| job | ename | sal |
|-----------|--------|---------|
| SALESMAN | ALLEN | 1600.00 |
| SALESMAN | WARD | 1250.00 |
| SALESMAN | MARTIN | 1250.00 |
| MANAGER | BLAKE | 2850.00 |
| MANAGER | CLARK | 2450.00 |
| PRESIDENT | KING | 5000.00 |
| SALESMAN | TURNER | 1500.00 |
| CLERK | JAMES | 950.00 |
| CLERK | MILLER | 1300.00 |

9 rows in set (0.00 sec)

6. Display name concatenated with dept code separated by comma and space. Name the column as 'Emp info'.

```
mysql> select concat(ename,", ",deptno) EmpInf from emp;
+-----+
| EmpInf |
+-----+
| SMITH, 20 |
| ALLEN, 30 |
| WARD, 30 |
| JONES, 20 |
| MARTIN, 30 |
| BLAKE, 30 |
| CLARK, 10 |
| SCOTT, 20 |
| KING, 10 |
| TURNER, 30 |
| ADAMS, 20 |
| JAMES, 30 |
| FORD, 20 |
| MILLER, 10 |
+-----+
14 rows in set (0.01 sec)
```

7. Display the emp details who do not have manager.

```
mysql> select * from emp where mgr is null;
+-----+-----+-----+-----+-----+-----+-----+-----+
| EMPNO | ENAME | JOB      | MGR  | HIREDATE | SAL      | COMM | DEPTNO |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 7839  | KING  | PRESIDENT | NULL | 1981-11-17 | 5000.00 | NULL | 10 |
+-----+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

8. Write a query which will display name, department no and date of joining of all employee who were joined January 1, 1981 and March 31, 1983. Sort it based on date of joining (ascending).

```
mysql> select ename,deptno,hiredate
-> from emp
-> where hiredate between '1981-1-1' and '1983-3-31'
-> order by hiredate;
```

| ename | deptno | hiredate |
|--------|--------|------------|
| ALLEN | 30 | 1981-02-20 |
| WARD | 30 | 1981-02-22 |
| JONES | 20 | 1981-04-02 |
| BLAKE | 30 | 1981-05-01 |
| CLARK | 10 | 1981-06-09 |
| TURNER | 30 | 1981-09-08 |
| MARTIN | 30 | 1981-09-28 |
| KING | 10 | 1981-11-17 |
| JAMES | 30 | 1981-12-03 |
| FORD | 20 | 1981-12-03 |
| MILLER | 10 | 1982-01-23 |
| SCOTT | 20 | 1982-12-09 |
| ADAMS | 20 | 1983-01-12 |

13 rows in set (0.00 sec)

9. Display the employee details where the job contains word 'AGE' anywhere in the Job.

```
mysql> select * from emp where job like '%AGE%';
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|-------|---------|------|------------|---------|------|--------|
| 7566 | JONES | MANAGER | 7839 | 1981-04-02 | 2975.00 | NULL | 20 |
| 7698 | BLAKE | MANAGER | 7839 | 1981-05-01 | 2850.00 | NULL | 30 |
| 7782 | CLARK | MANAGER | 7839 | 1981-06-09 | 2450.00 | NULL | 10 |

3 rows in set (0.00 sec)

11. List the details of the employee , whose names start with 'A' and end with 'S' or whose names contains N as the second or third character, and ending with either 'N' or 'S'.

```
mysql> select * from emp where ename like 'A%S' or ename like '_N%' or ename like '_N%S' or ename like '__N%' or ename like '__N%S';
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|-------|---------|------|------------|---------|------|--------|
| 7566 | JONES | MANAGER | 7839 | 1981-04-02 | 2975.00 | NULL | 20 |
| 7876 | ADAMS | CLERK | 7788 | 1983-01-12 | 1100.00 | NULL | 20 |

2 rows in set (0.00 sec)

12. List the names of the emp having '_' character in their name.

```
mysql> select * from emp
-> where ename regexp ".*_.*";
Empty set (0.02 sec)
```

Single Row functions

1. To list all employees and their email, to generate email use 2 to 5 characters from ename
Concat it with 2 to 4 characters in job and then concat it with '@mycompany.com'

```
mysql> select empno,ename,concat(left(ename,4),left(job,4),'@mycompany.com') emailid from emp;
```

| empno | ename | emailid |
|-------|--------|------------------------|
| 7369 | SMITH | SMITCLER@mycompany.com |
| 7499 | ALLEN | ALLESAL@mycompany.com |
| 7521 | WARD | WARDSAL@mycompany.com |
| 7566 | JONES | JONEMANA@mycompany.com |
| 7654 | MARTIN | MARTSAL@mycompany.com |
| 7698 | BLAKE | BLAKMANA@mycompany.com |
| 7782 | CLARK | CLARMANA@mycompany.com |
| 7788 | SCOTT | SCOTANAL@mycompany.com |
| 7839 | KING | KINGPRES@mycompany.com |
| 7844 | TURNER | TURNAL@mycompany.com |
| 7876 | ADAMS | ADAMCLER@mycompany.com |
| 7900 | JAMES | JAMECLER@mycompany.com |
| 7902 | FORD | FORDANAL@mycompany.com |
| 7934 | MILLER | MILLCLER@mycompany.com |

14 rows in set (0.01 sec)

2. List all employees who joined in September.

```
mysql> select * from emp  
-> where month(hiredate)=9;
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|--------|----------|------|------------|---------|---------|--------|
| 7654 | MARTIN | SALESMAN | 7698 | 1981-09-28 | 1953.12 | 1400.00 | 30 |
| 7844 | TURNER | SALESMAN | 7698 | 1981-09-08 | 2343.75 | 0.00 | 30 |

2 rows in set (0.01 sec)

3. List the empno, name, and department number of the emp who have experience of 18 or more years and sort them based on their experience.


```
mysql> select empno,ename,deptno,floor(datediff(curdate(),hiredate)/365) experiance
-> from emp
-> where floor(datediff(curdate(),hiredate)/365)>=18
-> order by experiance;
```

| empno | ename | deptno | experiance |
|-------|--------|--------|------------|
| 7788 | SCOTT | 20 | 41 |
| 7876 | ADAMS | 20 | 41 |
| 7654 | MARTIN | 30 | 42 |
| 7698 | BLAKE | 30 | 42 |
| 7782 | CLARK | 10 | 42 |
| 7839 | KING | 10 | 42 |
| 7844 | TURNER | 30 | 42 |
| 7900 | JAMES | 30 | 42 |
| 7902 | FORD | 20 | 42 |
| 7934 | MILLER | 10 | 42 |
| 7369 | SMITH | 20 | 43 |
| 7499 | ALLEN | 30 | 43 |
| 7521 | WARD | 30 | 43 |
| 7566 | JONES | 20 | 43 |

14 rows in set (0.01 sec)

4. Display the employee details who joined on 3rd of any month or any year

```
mysql> select * from emp
-> where day(hiredate)=03;
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|-------|---------|------|------------|---------|------|--------|
| 7900 | JAMES | CLERK | 7698 | 1981-12-03 | 1484.38 | NULL | 30 |
| 7902 | FORD | ANALYST | 7566 | 1981-12-03 | 4687.50 | NULL | 20 |

2 rows in set (0.00 sec)

5. display all employees who joined between years 1981 to 1983.

```
mysql> select * from emp
-> where year(hiredate) between 1981 and 1983;
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|--------|-----------|------|------------|---------|---------|--------|
| 7499 | ALLEN | SALESMAN | 7698 | 1981-02-20 | 2500.00 | 300.00 | 30 |
| 7521 | WARD | SALESMAN | 7698 | 1981-02-22 | 1953.12 | 500.00 | 30 |
| 7566 | JONES | MANAGER | 7839 | 1981-04-02 | 4648.44 | NULL | 20 |
| 7654 | MARTIN | SALESMAN | 7698 | 1981-09-28 | 1953.12 | 1400.00 | 30 |
| 7698 | BLAKE | MANAGER | 7839 | 1981-05-01 | 4453.12 | NULL | 30 |
| 7782 | CLARK | MANAGER | 7839 | 1981-06-09 | 3828.12 | NULL | 10 |
| 7788 | SCOTT | ANALYST | 7566 | 1982-12-09 | 4687.50 | NULL | 20 |
| 7839 | KING | PRESIDENT | NULL | 1981-11-17 | 7812.50 | NULL | 10 |
| 7844 | TURNER | SALESMAN | 7698 | 1981-09-08 | 2343.75 | 0.00 | 30 |
| 7876 | ADAMS | CLERK | 7788 | 1983-01-12 | 1718.75 | NULL | 20 |
| 7900 | JAMES | CLERK | 7698 | 1981-12-03 | 1484.38 | NULL | 30 |
| 7902 | FORD | ANALYST | 7566 | 1981-12-03 | 4687.50 | NULL | 20 |
| 7934 | MILLER | CLERK | 7782 | 1982-01-23 | 2031.25 | NULL | 10 |

13 rows in set (0.00 sec)

Group functions

6. Display the Highest, Lowest, Total & Average salary of all employee. Label the columns Maximum, Minimum, Total and Average respectively for each Department. Also round the result to the nearest whole number.

```
mysql> select deptno, floor(max(sal)) Maximum, floor(min(sal)) Minimum, floor(avg(sal))
Average from emp
-> group by deptno;
```

| deptno | Maximum | Minimum | Average |
|--------|---------|---------|---------|
| 20 | 4687 | 1250 | 3398 |
| 30 | 4453 | 1484 | 2447 |
| 10 | 7812 | 2031 | 4557 |

3 rows in set (0.02 sec)

7. Display Department no and number of managers working in that department. Label the column as 'Total Number of Managers' for each department.

```
mysql> select e.deptno, e.empno, count(*) TotalNumberOfManager from emp e
-> inner join emp m on m.empno=e.empno
-> group by e.deptno;
```

| deptno | empno | TotalNumberOfManager |
|--------|-------|----------------------|
| 20 | 7369 | 5 |
| 30 | 7499 | 6 |
| 10 | 7782 | 3 |

3 rows in set (0.01 sec)

ASSIGNMENT 1(Supplier)

1. display all suppliers who statys in state either in california or Texas or Arkansas.

```
mysql> select * from suppliers where state in('California','Texas','Arkansas');
+-----+-----+-----+-----+
| supplier_id | supplier_name | city | state |
+-----+-----+-----+-----+
| 200 | Google | Mountain View | California |
| 300 | Oracle | Redwood City | California |
| 400 | Kimberly-Clark | Irving | Texas |
| 500 | Tyson Foods | Springdale | Arkansas |
| 700 | Dole Food Company | Westlake Village | California |
| 900 | Electronic Arts | Redwood City | California |
+-----+-----+-----+-----+
6 rows in set (0.00 sec)
```

2. list all suppliers who does not stay in Springdale.

```
mysql> select * from suppliers where city not like 'Springdale';
+-----+-----+-----+-----+
| supplier_id | supplier_name | city | state |
+-----+-----+-----+-----+
| 100 | Microsoft | Redmond | Washington |
| 200 | Google | Mountain View | California |
| 300 | Oracle | Redwood City | California |
| 400 | Kimberly-Clark | Irving | Texas |
| 600 | SC Johnson | Racine | Wisconsin |
| 700 | Dole Food Company | Westlake Village | California |
| 800 | Flowers Foods | Thomasville | Georgia |
| 900 | Electronic Arts | Redwood City | California |
+-----+-----+-----+-----+
8 rows in set (0.00 sec)
```

3. find orderid and customerid for orders place on date 18-feb-16.

```
mysql> select order_id,customer_id from orders where order_date='2016-02-18';
+-----+-----+
| order_id | customer_id |
+-----+-----+
| 4 | 4000 |
| 5 | NULL |
+-----+-----+
2 rows in set (0.00 sec)
```

4. find orderid and customerid for orders place on feb 2016.

```
mysql> select * from orders where order_date between '2016-02-01' and '2016-02-29';
+-----+-----+-----+
| order_id | customer_id | order_date |
+-----+-----+-----+
| 4 | 4000 | 2016-02-18 |
| 5 | NULL | 2016-02-18 |
+-----+-----+-----+
2 rows in set (0.00 sec)
```

5. find all customers with name 'Reynolds', or Anderson.

```
mysql> select * from customers where last_name in('Reynolds','Anderson');
+-----+-----+-----+-----+
| customer_id | last_name | first_name | favorite_website |
+-----+-----+-----+-----+
|          7000 | Reynolds | Allen      | checkyourmath.com |
|          8000 | Anderson | Paige      | NULL              |
+-----+-----+-----+-----+
2 rows in set (0.00 sec)
```

6. find all suppliers with supplierid >=200 and <=700.

```
mysql> select * from suppliers where supplier_id between 200 and 700;
+-----+-----+-----+-----+
| supplier_id | supplier_name | city          | state |
+-----+-----+-----+-----+
|          200 | Google        | Mountain View | California |
|          300 | Oracle        | Redwood City  | California |
|          400 | Kimberly-Clark | Irving        | Texas   |
|          500 | Tyson Foods   | Springdale    | Arkansas |
|          600 | SC Johnson    | Racine        | Wisconsin |
|          700 | Dole Food Company | Westlake Village | California |
+-----+-----+-----+-----+
6 rows in set (0.00 sec)
```

7. find all customers for whome favorite_website is not given

```
mysql> select * from customers where favorite_website is null;
+-----+-----+-----+-----+
| customer_id | last_name | first_name | favorite_website |
+-----+-----+-----+-----+
|          8000 | Anderson | Paige      | NULL              |
+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

8. find all customers for whome favorite_website is given.

```
mysql> select * from customers where favorite_website is not null;
+-----+-----+-----+-----+
| customer_id | last_name | first_name | favorite_website |
+-----+-----+-----+-----+
|          4000 | Jackson   | Joe        | techonthenet.com |
|          5000 | Smith     | Jane       | digminecraft.com |
|          6000 | Ferguson  | Samantha   | bigactivities.com |
|          7000 | Reynolds  | Allen      | checkyourmath.com |
|          9000 | Johnson   | Derek      | KLFSpro.com       |
+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

9. find all suppliers with supplierid not >=200 and not <=700

```
mysql> select * from suppliers where supplier_id not between 200 and 700;
+-----+-----+-----+-----+
| supplier_id | supplier_name | city       | state    |
+-----+-----+-----+-----+
| 100         | Microsoft     | Redmond    | Washington |
| 800         | Flowers Foods | Thomasville | Georgia    |
| 900         | Electronic Arts | Redwood City | California |
+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

Date and Time functions Assignment day03

1. Write a query to display the last day of the month (in datetime format) three months before the current month.

```
mysql> select last_day(date_sub(curdate(),interval 3 month));
+-----+
| last_day(date_sub(curdate(),interval 3 month)) |
+-----+
| 2023-12-31                                     |
+-----+
1 row in set (0.00 sec)
```

2. Write a query to get the distinct Mondays from hiredate in emp tables.

```
mysql> select distinct hiredate,dayname(hiredate)
-> from emp
-> where dayname(hiredate)='Monday';
+-----+-----+
| hiredate | dayname(hiredate) |
+-----+-----+
| 1981-09-28 | Monday            |
+-----+-----+
1 row in set (0.00 sec)
```

3. Write a query to calculate your age in year.

```
mysql> select floor(datediff(curdate(),'2001-12-23')/365) AGE;
+-----+
| AGE |
+-----+
| 22 |
+-----+
1 row in set (0.00 sec)
```

12. Write a query to display the current date in the following format.

Sample output: 05/09/2014

```
mysql> select date_format(curdate(), "%d/%m/%Y");
+-----+
| date_format(curdate(), "%d/%m/%Y") |
+-----+
| 23/03/2024 |
+-----+
1 row in set (0.00 sec)
```

13. Write a query to display the current date in the following format.

Sample output: 12:00 AM Sep 5, 2014.

14. Write a query to get the employees who joined in the month of June.

```
mysql> select empno,ename,hiredate,extract(month from hiredate)
-> from emp
-> where extract(month from hiredate)=06;
+-----+-----+-----+-----+
| empno | ename | hiredate | extract(month from hiredate) |
+-----+-----+-----+-----+
| 7782 | CLARK | 1981-06-09 | 6 |
+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

16. Write a query to get first name of employees who joined in 1987.

```
mysql> select ename,hiredate,extract(year from hiredate)
-> from emp
-> where extract(year from hiredate)=1987;
Empty set (0.00 sec)
```

17. Write a query to get employees whose experience is more than 5 years.

```
mysql> select ename,hiredate,floor(datediff(curdate(),hiredate)/365) experience from emp
-> where floor(datediff(curdate(),hiredate)/365)>=5;
+-----+-----+-----+
| ename | hiredate | experience |
+-----+-----+-----+
| SMITH | 1980-12-17 | 43 |
| ALLEN | 1981-02-20 | 43 |
| WARD | 1981-02-22 | 43 |
| JONES | 1981-04-02 | 43 |
| MARTIN | 1981-09-28 | 42 |
| BLAKE | 1981-05-01 | 42 |
| CLARK | 1981-06-09 | 42 |
| SCOTT | 1982-12-09 | 41 |
| KING | 1981-11-17 | 42 |
| TURNER | 1981-09-08 | 42 |
| ADAMS | 1983-01-12 | 41 |
| JAMES | 1981-12-03 | 42 |
| FORD | 1981-12-03 | 42 |
| MILLER | 1982-01-23 | 42 |
+-----+-----+-----+
14 rows in set (0.00 sec)
```

18. Write a query to get employee ID, last name, and date of first salary of the employees.

19. Write a query to get first name, hire date and experience of the employees.

```
mysql> select ename,hiredate,floor(datediff(curdate(),hiredate)/365) experience
-> from emp;
+-----+-----+-----+
| ename | hiredate | experience |
+-----+-----+-----+
| SMITH | 1980-12-17 | 43 |
| ALLEN | 1981-02-20 | 43 |
| WARD | 1981-02-22 | 43 |
| JONES | 1981-04-02 | 43 |
| MARTIN | 1981-09-28 | 42 |
| BLAKE | 1981-05-01 | 42 |
| CLARK | 1981-06-09 | 42 |
| SCOTT | 1982-12-09 | 41 |
| KING | 1981-11-17 | 42 |
| TURNER | 1981-09-08 | 42 |
| ADAMS | 1983-01-12 | 41 |
| JAMES | 1981-12-03 | 42 |
| FORD | 1981-12-03 | 42 |
| MILLER | 1982-01-23 | 42 |
+-----+-----+-----+
14 rows in set (0.00 sec)
```

Sample table: employees

20. Write a query to get the department ID, year, and number of employees joined.

Day05

player (player_id, pname,speciality,date_of joining,num_matches,team_id)

team(team_id, tname,player_num)

matches(match_id, team1_id,team2_id,match_date,winner,man_of_the match)

simple query

1. display all players playerid, player name and experience

```
mysql> select player_id,pname,date_of_joining,floor(datediff(curdate(),date_of_joining)/365) from player;
```

| player_id | pname | date_of_joining | floor(datediff(curdate(),date_of_joining)/365) |
|-----------|--------------|-----------------|--|
| 3 | Maxwell | 2020-06-03 | 3 |
| 7 | MS Dhoni | 2004-08-04 | 19 |
| 11 | Anil kumble | 1995-03-06 | 29 |
| 12 | Bumrah | 2022-08-04 | 1 |
| 17 | Rachin | 2023-06-03 | 0 |
| 18 | Virat Kohli | 2008-03-01 | 16 |
| 20 | Ishan Kishan | 2022-12-04 | 1 |
| 34 | Shivam Dube | 2017-08-14 | 6 |
| 45 | Rohit sharma | 2009-08-04 | 14 |

```
rows in set (0.00 sec)
```

2. display all player with experience > 5 years

```
mysql> select player_id,pname,date_of_joining,floor(datediff(curdate(),date_of_joining)/365) Experience from player
-> where floor(datediff(curdate(),date_of_joining)/365) >5;
```

| player_id | pname | date_of_joining | Experience |
|-----------|--------------|-----------------|------------|
| 7 | MS Dhoni | 2004-08-04 | 19 |
| 11 | Anil kumble | 1995-03-06 | 29 |
| 18 | Virat Kohli | 2008-03-01 | 16 |
| 34 | Shivam Dube | 2017-08-14 | 6 |
| 45 | Rohit sharma | 2009-08-04 | 14 |

```
5 rows in set (0.00 sec)
```

3. display all players joined in march month, any year

```
mysql> select * from player
-> where month(date_of_joining)=03;
```

| player_id | pname | speciality | date_of_joining | num_matches | team_id |
|-----------|-------------|------------|-----------------|-------------|---------|
| 11 | Anil kumble | bowler | 1995-03-06 | 212 | 10 |
| 18 | Virat Kohli | batsman | 2008-03-01 | 264 | 10 |

```
2 rows in set (0.01 sec)
```

4. display all players joined in march 1995

```
mysql> select * from player
-> where year(date_of_joining)=1995 and month(date_of_joining)=03;
```

| player_id | pname | speciality | date_of_joining | num_matches | team_id |
|-----------|-------------|------------|-----------------|-------------|---------|
| 11 | Anil kumble | bowler | 1995-03-06 | 212 | 10 |

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

5. display all player with number of matches played are either 5 or 10 or 8


```
mysql> select * from player
-> where num_matches in(5,10,8);
```

| player_id | pname | speciality | date_of_joining | num_matches | team_id |
|-----------|---------|-------------|-----------------|-------------|---------|
| 3 | Maxwell | all_rounder | 2020-06-03 | 10 | 10 |
| 17 | Rachin | batsman | 2023-06-03 | 8 | 11 |

```
2 rows in set (0.00 sec)
```

6. display all employees who are either batsman, bowler

```
mysql> select * from player
-> where speciality in('batsman','bowler');
```

| player_id | pname | speciality | date_of_joining | num_matches | team_id |
|-----------|--------------|------------|-----------------|-------------|---------|
| 11 | Anil kumble | bowler | 1995-03-06 | 212 | 10 |
| 12 | Bumrah | bowler | 2022-08-04 | 22 | 13 |
| 17 | Rachin | batsman | 2023-06-03 | 8 | 11 |
| 18 | Virat Kohli | batsman | 2008-03-01 | 264 | 10 |
| 34 | Shivam Dube | batsman | 2017-08-14 | 82 | 11 |
| 45 | Rohit sharma | batsman | 2009-08-04 | 212 | 13 |

```
6 rows in set (0.00 sec)
```

7. display all employees who joined in year 1995 or 1996 or 1997

```
mysql> select * from player
-> where date_of_joining in(1995,1996,1997);
```

Empty set, 3 warnings (0.00 sec)

```
mysql> select * from player
-> where year(date_of_joining) in(1995,1996,1997);
```

| player_id | pname | speciality | date_of_joining | num_matches | team_id |
|-----------|-------------|------------|-----------------|-------------|---------|
| 11 | Anil kumble | bowler | 1995-03-06 | 212 | 10 |

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

Nested query

1. list all player who plays in virat kohalis team

```
mysql> select * from player
-> where team_id=(select team_id from player where pname="Virat Kohli");
```

| player_id | pname | speciality | date_of_joining | num_matches | team_id |
|-----------|-------------|-------------|-----------------|-------------|---------|
| 3 | Maxwell | all_rounder | 2020-06-03 | 10 | 10 |
| 11 | Anil kumble | bowler | 1995-03-06 | 212 | 10 |
| 18 | Virat Kohli | batsman | 2008-03-01 | 264 | 10 |

```
3 rows in set (0.12 sec)
```

- list all players who played matches in year 2024
- list all matches in which man of the match is from team1
- list all matches in which man of the match is from team1

5. list all matches in which csk team win.

```
mysql> select * from matches where
-> winner =(select team_id from team where tname="CSK");
```

| match_id | team1_id | team2_id | match_date | winner | man_of_the_match |
|----------|----------|----------|------------|--------|------------------|
| 102 | 10 | 11 | 2024-03-22 | 11 | 7 |
| 105 | 11 | 13 | 2022-08-09 | 11 | 34 |

2 rows in set (0.01 sec)

6. list all matches in which either csk or rcb team won the match

```
mysql> select * from matches
-> where winner in (select team_id from team where tname in ("RCB","CSK"));
```

| match_id | team1_id | team2_id | match_date | winner | man_of_the_match |
|----------|----------|----------|------------|--------|------------------|
| 102 | 10 | 11 | 2024-03-22 | 11 | 7 |
| 103 | 10 | 11 | 2024-03-27 | 10 | 18 |
| 105 | 11 | 13 | 2022-08-09 | 11 | 34 |
| 106 | 13 | 10 | 2023-08-09 | 10 | 3 |

4 rows in set (0.01 sec)

7. list all teams who one atleast one match

8. list all teams who does not played any match

9. list all team name, in which no players are their

10. list all players who are in dhoni's team

```
mysql> select * from player
-> where team_id=(select team_id from player where pname="MS Dhoni");
```

| player_id | pname | speciality | date_of_joining | num_matches | team_id |
|-----------|-------------|---------------|-----------------|-------------|---------|
| 7 | MS Dhoni | wicket_keeper | 2004-08-04 | 122 | 11 |
| 17 | Rachin | batsman | 2023-06-03 | 8 | 11 |
| 34 | Shivam Dube | batsman | 2017-08-14 | 82 | 11 |

3 rows in set (0.01 sec)

10. list all players who are in either dhoni's or virat's team

```
mysql> select * from player
-> where team_id in (select team_id from player where pname in ("MS Dhoni","Virat Kohli"));
```

| player_id | pname | speciality | date_of_joining | num_matches | team_id |
|-----------|-------------|---------------|-----------------|-------------|---------|
| 7 | MS Dhoni | wicket_keeper | 2004-08-04 | 122 | 11 |
| 17 | Rachin | batsman | 2023-06-03 | 8 | 11 |
| 34 | Shivam Dube | batsman | 2017-08-14 | 82 | 11 |
| 3 | Maxwell | all_rounder | 2020-06-03 | 10 | 10 |
| 11 | Anil kumble | bowler | 1995-03-06 | 212 | 10 |
| 18 | Virat Kohli | batsman | 2008-03-01 | 264 | 10 |

6 rows in set (0.00 sec)

PLSQL ASSIGNMENT

Solve the following

1. write a procedure to insert record into employee table.

the procedure should accept empno, ename, sal, job, hiredate as input parameter

write insert statement inside procedure insert_rec to add one record into table

create procedure insert_rec(peno int,pnm varchar(20),psal decimal(9,2),pjob

varchar(20),phiredate date)

begin

insert into emp(empno,ename,sal,job,hiredate)

values(peno,pnm,psal,pjob,phiredate)

end//

```
Database changed
mysql> delimiter //
mysql> create procedure insertrecord(dempno int,dname varchar(20),dsal float(9,2),djob varchar(20),dhiredate date)
-> begin
-> insert into emp(empno,ename,sal,job,hiredate)
-> values(dempno,dname,dsal,djob,dhiredate);
-> end//
Query OK, 0 rows affected, 1 warning (0.21 sec)

mysql> delimiter ;
mysql> call insertrecord(1111,"Vihan",2500,"Manager","2019-08-23");
Query OK, 1 row affected (0.03 sec)

mysql> select * from emp;
+-----+-----+-----+-----+-----+-----+-----+-----+
| EMPNO | ENAME | JOB   | MGR | HIREDATE | SAL   | COMM | DEPTNO |
+-----+-----+-----+-----+-----+-----+-----+
| 1000 | Radha | Manager | NULL | 2024-03-03 | 1500.00 | 400.00 | 30 |
| 1001 | Raj   | Clerk   | NULL | 2024-01-07 | 2500.00 | 100.00 | 30 |
| 1002 | Aditya | Clerk   | NULL | 2021-06-01 | 2500.00 | 200.00 | 30 |
| 1111 | Vihan | Manager | NULL | 2019-08-23 | 2500.00 | NULL | NULL |
| 7369 | SMITH | CLERK   | 7902 | 1980-12-17 | 800.00 | NULL | 20 |
| 7499 | ALLEN | SALESMAN | 7698 | 1981-02-20 | 1600.00 | 300.00 | 30 |
| 7521 | WARD  | SALESMAN | 7698 | 1981-02-22 | 1250.00 | 500.00 | 30 |
| 7566 | JONES | MANAGER | 7839 | 1981-04-02 | 2975.00 | NULL | 20 |
| 7654 | MARTIN | SALESMAN | 7698 | 1981-09-28 | 1250.00 | 1400.00 | 30 |
| 7698 | BLAKE | MANAGER | 7839 | 1981-05-01 | 2850.00 | NULL | 30 |
| 7782 | CLARK | MANAGER | 7839 | 1981-06-09 | 2450.00 | NULL | 10 |
| 7788 | SCOTT | ANALYST | 7566 | 1982-12-09 | 3000.00 | NULL | 20 |
| 7839 | KING  | PRESIDENT | NULL | 1981-11-17 | 5000.00 | NULL | 10 |
| 7844 | TURNER | SALESMAN | 7698 | 1981-09-08 | 1500.00 | 0.00 | 30 |
| 7876 | ADAMS | CLERK   | 7788 | 1983-01-12 | 1100.00 | NULL | 20 |
| 7900 | JAMES | CLERK   | 7698 | 1981-12-03 | 950.00 | NULL | 30 |
| 7902 | FORD  | ANALYST | 7566 | 1981-12-03 | 3000.00 | NULL | 20 |
| 7934 | MILLER | CLERK   | 7782 | 1982-01-23 | 1300.00 | NULL | 10 |
+-----+-----+-----+-----+-----+-----+-----+
18 rows in set (0.00 sec)
```

2. write a procedure to delete record from employee table.

the procedure should accept empno as input parameter.

write delete statement inside procedure delete_emp to delete one record from emp

table

3. write a procedure to display empno,ename,deptno,dname for all employees with sal
> given salary. pass salary as a parameter to procedure

```
mysql> delimiter //
mysql> create procedure displayempdetail4(dsal float(9,2))
-> begin
-> select e.empno,e.ename,e.deptno,d.dname,d.deptno
-> from emp e inner join dept d
-> where e.deptno=d.deptno and sal>dsal;
-> end //
Query OK, 0 rows affected, 1 warning (0.12 sec)

mysql> delimiter ;
mysql> call displayempdetail4(2000);
```

| empno | ename | deptno | dname | deptno |
|-------|--------|--------|------------|--------|
| 1001 | Raj | 30 | SALES | 30 |
| 1002 | Aditya | 30 | SALES | 30 |
| 7566 | JONES | 20 | RESEARCH | 20 |
| 7698 | BLAKE | 30 | SALES | 30 |
| 7782 | CLARK | 10 | ACCOUNTING | 10 |
| 7788 | SCOTT | 20 | RESEARCH | 20 |
| 7839 | KING | 10 | ACCOUNTING | 10 |
| 7902 | FORD | 20 | RESEARCH | 20 |

```
8 rows in set (0.00 sec)
```

4. write a procedure to find min,max,avg of salary and number of employees in the
given deptno.

deptno --> in parameter

min,max,avg and count ---> out type parameter

execute procedure and then display values min,max,avg and count

```
mysql> delimiter //
mysql> create procedure aggregatepr(in pdeptno int ,out pmax float(9,2),out pmin float(9,2),out pavg float(9,2),out cnt int)
-> begin
-> select deptno,max(sal),min(sal),avg(sal),count(*) into pdeptno,pmax,pmin,pavg,cnt
-> from emp
-> where deptno=pdeptno;
-> end //
Query OK, 0 rows affected, 3 warnings (0.16 sec)

mysql> delimiter ;
mysql> call aggregatepr(10,@m,@i,@a,@c);
Query OK, 1 row affected (0.03 sec)

mysql> select @m,@i,@a,@c;
```

| @m | @i | @a | @c |
|------|------|----------------|----|
| 5000 | 1300 | 2916.669921875 | 3 |

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

5. write a procedure to display all pid,pname,cid,cname and salesman name(use product,category and salesman table).

```
mysql> delimiter //
mysql> create procedure getproductdetail(in vpid int)
-> begin
-> select p.pid,p.pname,p.cid,p.sid,c.cid,c.cname,s.sid,s.sname from product p inner join category c on p.cid=c.cid inner join salesman s on s.sid=p.sid
-> where p.pid=vpid;
-> end //
Query OK, 0 rows affected (0.03 sec)

mysql> delimiter ;
mysql> call getproductdetail(3);
+-----+-----+-----+-----+-----+-----+
| pid | pname | cid | sid | cid | cname | sid | sname |
+-----+-----+-----+-----+-----+-----+
| 3 | Maggie | 30 | 1003 | 30 | Snack | 1003 | Ved |
+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

6. write a procedure to display all vehicles bought by a customer. pass customer name as a parameter.(use vehicle,salesman,customer and relation table)

```
mysql> delimiter //
mysql> create procedure displayvehicle1(in vcname varchar(20))
-> begin
-> select c.Cname,v.vname from cust_vehicle cv inner join vehicle v on cv.vid=v.vid
-> inner join customer1 c on c.custid=cv.custid
-> where c.cname=vcname;
-> end //
Query OK, 0 rows affected (0.03 sec)

mysql> delimiter ;
mysql> call displayvehicle1("Nilima");
+-----+-----+
| Cname | vname |
+-----+-----+
| Nilima | Aactiva |
| Nilima | Santro |
+-----+-----+
2 rows in set (0.01 sec)

Query OK, 0 rows affected (0.03 sec)
```

7. Write a procedure that displays the following information of all emp

Empno,Name,job,Salary,Status,deptno

Note: - Status will be (Greater, Lesser or Equal) respective to average salary of their own department. Display an error message Emp table is empty if there is no matching record.

```
mysql> create procedure displayemp3()
-> begin
-> declare vfinished int default 0;
-> declare vempno,vdeptno int;
-> declare vjob,vename,vstatus varchar(20);
-> declare vsal,vavgsal float(9,2);
->
-> declare empcur cursor for select empno,ename,job,sal,deptno from emp;
-> declare continue handler for NOT FOUND
-> set vfinished=1;
->
-> open empcur;
-> label1:loop
-> fetch empcur into vempno,vename,vjob,vsal,vdeptno;
-> if vfinished=1 then
-> leave label1;
-> end if;
-> select avg(sal) into vavgsal
-> from emp
-> where deptno=vdeptno;
->
-> if vsal<vavgsal then
-> set vstatus="Lesser";
-> elseif vsal>vavgsal then
-> set vstatus="Greater";
-> else
-> set vstatus="Eqaul";
-> end if;
-> select vempno,vename,vjob,vdeptno,vstatus;
-> end loop;
-> close empcur;
-> end //
```

Query OK, 0 rows affected, 1 warning (0.03 sec)

```
mysql> delimiter ;
```

```
mysql> call displayemp3();
```

```
+-----+-----+-----+-----+-----+
| vempno | vename | vjob   | vdeptno | vstatus |
+-----+-----+-----+-----+-----+
| 1000   | Krishna | Manager | 30      | Lesser  |
+-----+-----+-----+-----+-----+
```

```

+-----+-----+-----+-----+-----+
| vempno | vename | vjob   | vdeptno | vstatus |
+-----+-----+-----+-----+-----+
| 1001   | Raj    | Clerk | 30      | Greater |
+-----+-----+-----+-----+-----+
1 row in set (0.02 sec)

+-----+-----+-----+-----+-----+
| vempno | vename | vjob   | vdeptno | vstatus |
+-----+-----+-----+-----+-----+
| 1002   | Aditya | Clerk | 30      | Greater |
+-----+-----+-----+-----+-----+
1 row in set (0.04 sec)

+-----+-----+-----+-----+-----+
| vempno | vename | vjob   | vdeptno | vstatus |
+-----+-----+-----+-----+-----+
| 1111   | Vihan  | Manager | NULL    | Equal   |
+-----+-----+-----+-----+-----+
1 row in set (0.06 sec)

+-----+-----+-----+-----+-----+
| vempno | vename | vjob   | vdeptno | vstatus |
+-----+-----+-----+-----+-----+
| 7369   | SMITH  | CLERK | 20      | Lesser  |
+-----+-----+-----+-----+-----+
1 row in set (0.08 sec)

```

8. Write a procedure to update salary in emp table based on following rules.

Exp <= 35 then no Update

Exp > 35 and <= 38 then 20% of salary

Exp > 38 then 25% of salary

9. Write a procedure and a function.

Function: write a function to calculate number of years of experience of employee. (note: pass hiredate as a parameter)

Procedure: Capture the value returned by the above function to calculate the additional allowance for the emp based on the experience.

Additional Allowance = Year of experience x 3000

Calculate the additional allowance

and store Empno, ename, Date of Joining, and Experience in years and additional allowance in Emp_Allowance table.

```
create table emp_allowance(  
empno int,  
ename varchar(20),  
hiredate date,  
experience int,  
allowance decimal(9,2));
```

```
mysql> delimiter //  
mysql> create function calexp(fhiredate date) returns int  
-> begin  
-> declare vexp int;  
-> set vexp=floor(datediff(curdate(),fhiredate)/365);  
-> return vexp;  
-> end //  
Query OK, 0 rows affected (0.02 sec)
```

```
mysql> create table emp_allowance(  
-> nempno int,  
-> nename varchar(20),  
-> nhiredate date,  
-> nexp int,  
-> nallow float(9,2)  
-> );  
Query OK, 0 rows affected, 1 warning (0.19 sec)
```



```

mysql> create procedure calallowance2(in vempno int)
-> begin
-> declare vfinished int default 0;
-> declare vename varchar(20);
-> declare allowance float(9,2);
-> declare vempno,experiance int;
-> declare vhiredate date;
-> declare allcure cursor for select empno,ename,hiredate from emp;
-> declare continue handler for NOT FOUND
-> set vfinished=1;
->
-> open allcure;
-> label1:loop
-> fetch allcure into vempno,vename,vhiredate;
-> if vfinished=1 then
-> leave label1;
-> end if;
-> set experiance=calexp(vhiredate);
-> set allowance=experiance*3000;
-> insert into emp_allowance(nempno,nename,nhiredate,nexp,nallow)
-> values(vempno,vename,vhiredate,experiance,allowance);
-> end loop;
-> close allcure;
-> end //
Query OK, 0 rows affected, 1 warning (0.03 sec)

mysql> call calallowance2(7902);

```

```

mysql> select * from emp_allowance;

```

| nempno | nename | nhiredate | nexp | nallow |
|--------|---------|------------|------|-----------|
| 1000 | Krishna | 2024-03-03 | 0 | 0.00 |
| 1001 | Raj | 2024-01-07 | 0 | 0.00 |
| 1002 | Aditya | 2021-06-01 | 2 | 6000.00 |
| 1111 | Vihan | 2019-08-23 | 4 | 12000.00 |
| 7369 | SMITH | 1980-12-17 | 43 | 129000.00 |
| 7499 | ALLEN | 1981-02-20 | 43 | 129000.00 |
| 7521 | WARD | 1981-02-22 | 43 | 129000.00 |
| 7566 | JONES | 1981-04-02 | 43 | 129000.00 |
| 7654 | MARTIN | 1981-09-28 | 42 | 126000.00 |
| 7698 | BLAKE | 1981-05-01 | 42 | 126000.00 |
| 7782 | CLARK | 1981-06-09 | 42 | 126000.00 |
| 7788 | SCOTT | 1982-12-09 | 41 | 123000.00 |
| 7839 | KING | 1981-11-17 | 42 | 126000.00 |
| 7844 | TURNER | 1981-09-08 | 42 | 126000.00 |
| 7876 | ADAMS | 1983-01-12 | 41 | 123000.00 |
| 7900 | JAMES | 1981-12-03 | 42 | 126000.00 |
| 7902 | FORD | 1981-12-03 | 42 | 126000.00 |
| 7934 | MILLER | 1982-01-23 | 42 | 126000.00 |

```

18 rows in set (0.00 sec)

```

Q2. Write trigger

1. Write a trigger to store the old salary details in Emp _Back (Emp _Back has the same structure as emp table without any constraint) table.

(note :create emp_back table before writing trigger)

----- to create emp_back table

```
create table emp_back(  
empno int,  
ename varchar(20),  
oldsal decimal(9,2),  
newsal decimal(9,2)  
)
```

(note :

execute procedure written in Q8 and

check the entries in EMP_back table after execution of the procedure)

```
mysql> create table emp_back(  
-> empno int,  
-> ename varchar(30),  
-> oldsal float(9,2),  
-> newsal float(9,2),  
-> uname varchar(30),  
-> changedt datetime,  
-> action varchar(30));  
Query OK, 0 rows affected, 2 warnings (0.29 sec)  
  
mysql> create trigger oldsalary after update on emp for each row  
-> insert into emp_back values(empno,ename,old.sal,new.sal,user(),now(),"UPDATE");  
Query OK, 0 rows affected (0.07 sec)  
  
mysql> update emp  
-> set sal=5000  
-> where empno=1000;  
Query OK, 1 row affected (0.05 sec)  
Rows matched: 1 Changed: 1 Warnings: 0  
  
mysql> select *from emp_back;  
+-----+-----+-----+-----+-----+-----+-----+  
| empno | ename | oldsal | newsal | uname          | changedt          | action |  
+-----+-----+-----+-----+-----+-----+-----+  
| NULL  | NULL  | 1900.00 | 5000.00 | root@localhost | 2024-04-02 17:59:06 | UPDATE |  
+-----+-----+-----+-----+-----+-----+-----+  
1 row in set (0.00 sec)
```

2. Write a trigger which add entry in audit table when user tries to insert or delete records in employee table store empno,name,username and date on which operation performed and which action is done insert or delete. in emp_audit table.
create table before writing trigger.

create table empaudit(

```
empno int;
ename varchar(20),
username varchar(20);
chdate date;
action varchar(20)
);
```

```
mysql> create table emp_audit(
-> empno int,
-> oldname varchar(20),
-> newname varchar(20),
-> oldjob varchar(20),
-> newjob varchar(20),
-> oldsal float(9,2),
-> newsal float(9,2),
-> uname varchar(20),
-> changdt datetime,
-> action varchar(20)
-> );
Query OK, 0 rows affected, 2 warnings (0.29 sec)
```

```
mysql> create trigger insertemp after insert on emp for each row
-> insert into emp_audit values(new.empno,null,new.ename,null,new.job,null,new.sal,user(),now(),"INSERT");
Query OK, 0 rows affected (0.08 sec)
```

```
mysql> select * from emp_audit;
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| empno | oldname | newname | oldjob | newjob | oldsal | newsal | uname | changdt | action |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1111 | Rajan | Shilpa | Clerk | Manager | 1500.00 | 20000.00 | root@localhost | 2024-04-02 16:38:42 | UPDATE |
| 1111 | NULL | Bhagyashri | NULL | Developer | NULL | 50000.00 | root@localhost | 2024-04-02 17:11:00 | INSERT |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)
```

```
mysql> create trigger deleteemp after delete on emp for each row
-> insert into emp_audit values(old.empno,old.ename,null,old.job,null,old.sal,null,user(),now(),"DELETE");
Query OK, 0 rows affected (0.06 sec)
```

```
mysql> delete from emp
-> where empno=1111;
Query OK, 1 row affected (0.05 sec)
```

```
mysql> select *from emp_audit;
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| empno | oldname | newname | oldjob | newjob | oldsal | newsal | uname | changdt | action |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1111 | Rajan | Shilpa | Clerk | Manager | 1500.00 | 20000.00 | root@localhost | 2024-04-02 16:38:42 | UPDATE |
| 1111 | NULL | Bhagyashri | NULL | Developer | NULL | 50000.00 | root@localhost | 2024-04-02 17:11:00 | INSERT |
| 1111 | Bhagyashri | NULL | Developer | NULL | 50000.00 | NULL | root@localhost | 2024-04-02 17:43:14 | DELETE |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

3. Create table vehicle_history. Write a trigger to store old vehicleprice and new vehicle price in history table before you update price in vehicle table (note: use vehicle table).

```
create table vehicle_history(
vno int,
vname varchar(20),
oldprice decimal(9,2),
newprice decimal(9,2),
chdate date,
username varchar(20)
```

```
mysql> create table vehicle_history(
-> vno int,
-> vname varchar(20),
-> oldprice float(9,2),
-> newpeice float(9,2),
-> chdate date,
-> username varchar(20)
-> );
Query OK, 0 rows affected, 2 warnings (0.26 sec)
```

```
mysql> update vehicle
-> set price=40000
-> where vid=1;
Query OK, 1 row affected (0.05 sec)
Rows matched: 1 Changed: 1 Warnings: 0
```

```
mysql> select * from vehicle_history;
+-----+-----+-----+-----+-----+-----+
| vno | vname | oldprice | newpeice | chdate | username |
+-----+-----+-----+-----+-----+-----+
| 1 | Acura | 20000.00 | 40000.00 | 2024-04-02 | root@localhost |
+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

MAIN ASSIGNMENT

practice DQL statement

Write SQL statement for the following

1. To find all managers with salary >1500

```
mysql> select mgr,sal from emp
-> where sal>1500;
+-----+-----+
| mgr | sal |
+-----+-----+
| 7698 | 1600.00 |
| 7839 | 2450.00 |
| 7839 | 2850.00 |
| 7839 | 2975.00 |
| 7566 | 3000.00 |
| 7566 | 3000.00 |
| NULL | 5000.00 |
| NULL | 5000.00 |
+-----+-----+
8 rows in set (0.01 sec)
```

2. list all employees with sal >1200 and < 2000

```
mysql> select * from emp
-> where sal between 1200 and 2000;
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|--------|----------|------|------------|---------|---------|--------|
| 7521 | WARD | SALESMAN | 7698 | 1981-02-22 | 1250.00 | 500.00 | 30 |
| 7654 | MARTIN | SALESMAN | 7698 | 1981-09-28 | 1250.00 | 1400.00 | 30 |
| 7934 | MILLER | CLERK | 7782 | 1982-01-23 | 1300.00 | NULL | 10 |
| 7844 | TURNER | SALESMAN | 7698 | 1981-09-08 | 1500.00 | 0.00 | 30 |
| 7499 | ALLEN | SALESMAN | 7698 | 1981-02-20 | 1600.00 | 300.00 | 30 |

5 rows in set (0.00 sec)

3. list all employees with sal is 1600 or sal is 800 or sal is 1900

```
mysql> select * from emp
-> where sal in (1600,800,1900);
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|-------|----------|------|------------|---------|--------|--------|
| 7369 | SMITH | CLERK | 7902 | 1980-12-17 | 800.00 | NULL | 20 |
| 7499 | ALLEN | SALESMAN | 7698 | 1981-02-20 | 1600.00 | 300.00 | 30 |

2 rows in set (0.01 sec)

4. list all employees with R at second last position in name

```
mysql> select * from emp
-> where ename regexp "R.$";
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|-------|----------|------|------------|---------|--------|--------|
| 7521 | WARD | SALESMAN | 7698 | 1981-02-22 | 1250.00 | 500.00 | 30 |
| 7782 | CLARK | MANAGER | 7839 | 1981-06-09 | 2450.00 | NULL | 10 |
| 7902 | FORD | ANALYST | 7566 | 1981-12-03 | 3000.00 | NULL | 20 |

3 rows in set (0.03 sec)

5. List all employees with name starts with A and ends with N

```
mysql> select * from emp
-> where ename regexp "^A.*N$";
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|-------|----------|------|------------|---------|--------|--------|
| 7499 | ALLEN | SALESMAN | 7698 | 1981-02-20 | 1600.00 | 300.00 | 30 |

1 row in set (0.00 sec)

Q2. Solve following

1. list all employees with salary > 1250 and dept no=30

```
mysql> select * from emp
-> where sal>1250 and deptno=30;
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|--------|----------|------|------------|---------|--------|--------|
| 7844 | TURNER | SALESMAN | 7698 | 1981-09-08 | 1500.00 | 0.00 | 30 |
| 7499 | ALLEN | SALESMAN | 7698 | 1981-02-20 | 1600.00 | 300.00 | 30 |
| 7698 | BLAKE | MANAGER | 7839 | 1981-05-01 | 2850.00 | NULL | 30 |
| 1000 | Tinku | Manager | NULL | 2021-03-25 | 5000.00 | 200.00 | 30 |

4 rows in set (0.00 sec)

- list all employees with salary ≥ 1250 and ≤ 3000

```
mysql> select * from emp
-> where sal between 1250 and 3000;
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|--------|----------|------|------------|---------|---------|--------|
| 7521 | WARD | SALESMAN | 7698 | 1981-02-22 | 1250.00 | 500.00 | 30 |
| 7654 | MARTIN | SALESMAN | 7698 | 1981-09-28 | 1250.00 | 1400.00 | 30 |
| 7934 | MILLER | CLERK | 7782 | 1982-01-23 | 1300.00 | NULL | 10 |
| 7844 | TURNER | SALESMAN | 7698 | 1981-09-08 | 1500.00 | 0.00 | 30 |
| 7499 | ALLEN | SALESMAN | 7698 | 1981-02-20 | 1600.00 | 300.00 | 30 |
| 7782 | CLARK | MANAGER | 7839 | 1981-06-09 | 2450.00 | NULL | 10 |
| 7698 | BLAKE | MANAGER | 7839 | 1981-05-01 | 2850.00 | NULL | 30 |
| 7566 | JONES | MANAGER | 7839 | 1981-04-02 | 2975.00 | NULL | 20 |
| 7788 | SCOTT | ANALYST | 7566 | 1982-12-09 | 3000.00 | NULL | 20 |
| 7902 | FORD | ANALYST | 7566 | 1981-12-03 | 3000.00 | NULL | 20 |

10 rows in set (0.00 sec)

- list all employees with salary >1250 and < 3000

```
mysql> select * from emp
-> where sal>1250 and sal<3000;
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|--------|----------|------|------------|---------|--------|--------|
| 7934 | MILLER | CLERK | 7782 | 1982-01-23 | 1300.00 | NULL | 10 |
| 7844 | TURNER | SALESMAN | 7698 | 1981-09-08 | 1500.00 | 0.00 | 30 |
| 7499 | ALLEN | SALESMAN | 7698 | 1981-02-20 | 1600.00 | 300.00 | 30 |
| 7782 | CLARK | MANAGER | 7839 | 1981-06-09 | 2450.00 | NULL | 10 |
| 7698 | BLAKE | MANAGER | 7839 | 1981-05-01 | 2850.00 | NULL | 30 |
| 7566 | JONES | MANAGER | 7839 | 1981-04-02 | 2975.00 | NULL | 20 |

6 rows in set (0.00 sec)

- list all employees with salary either equal to 3000 or 1250 or 2500

```
mysql> select * from emp
-> where sal in(3000,1250,2500);
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|--------|----------|------|------------|---------|---------|--------|
| 7521 | WARD | SALESMAN | 7698 | 1981-02-22 | 1250.00 | 500.00 | 30 |
| 7654 | MARTIN | SALESMAN | 7698 | 1981-09-28 | 1250.00 | 1400.00 | 30 |
| 7788 | SCOTT | ANALYST | 7566 | 1982-12-09 | 3000.00 | NULL | 20 |
| 7902 | FORD | ANALYST | 7566 | 1981-12-03 | 3000.00 | NULL | 20 |

4 rows in set (0.00 sec)

5. list all employee with name=SMITH

```
mysql> select * from emp
-> where ename="SMITH";
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|-------|-------|------|------------|--------|------|--------|
| 7369 | SMITH | CLERK | 7902 | 1980-12-17 | 800.00 | NULL | 20 |

1 row in set (0.00 sec)

6. list all employees with name starting with S

```
mysql> select * from emp
-> where ename like 'S%';
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|-------|---------|------|------------|---------|------|--------|
| 7369 | SMITH | CLERK | 7902 | 1980-12-17 | 800.00 | NULL | 20 |
| 7788 | SCOTT | ANALYST | 7566 | 1982-12-09 | 3000.00 | NULL | 20 |

2 rows in set (0.00 sec)

7. list all employees with name ending with S

```
mysql> select * from emp
-> where ename like '%S';
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|-------|---------|------|------------|---------|------|--------|
| 7566 | JONES | MANAGER | 7839 | 1981-04-02 | 2975.00 | NULL | 20 |
| 7876 | ADAMS | CLERK | 7788 | 1983-01-12 | 1100.00 | NULL | 20 |
| 7900 | JAMES | CLERK | 7698 | 1981-12-03 | 950.00 | NULL | 30 |

3 rows in set (0.00 sec)

8. list all employees with name contains l at 2nd position

```
mysql> select * from emp
-> where ename like '_L%';
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|-------|----------|------|------------|---------|--------|--------|
| 7499 | ALLEN | SALESMAN | 7698 | 1981-02-20 | 1600.00 | 300.00 | 30 |
| 7698 | BLAKE | MANAGER | 7839 | 1981-05-01 | 2850.00 | NULL | 30 |
| 7782 | CLARK | MANAGER | 7839 | 1981-06-09 | 2450.00 | NULL | 10 |

```
3 rows in set (0.00 sec)
```

9. list all employees with name starts with A ends with N and somewhere in between L is there

```
mysql> select * from emp
-> where ename regexp "^A.*L.*N$";
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|-------|----------|------|------------|---------|--------|--------|
| 7499 | ALLEN | SALESMAN | 7698 | 1981-02-20 | 1600.00 | 300.00 | 30 |

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

10. list all employees with name starts with A and B at 3 rd position and P at second last position

```
mysql> select * from emp
-> where ename regexp "^A.B.*P.$";
Empty set (0.00 sec)
```

11. List all employees with name starts with either A or starts with S or starts with W
practice

```
mysql> select * from emp
-> where ename regexp "^A.*|^S.*|^W.*";
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|-------|----------|------|------------|---------|--------|--------|
| 7369 | SMITH | CLERK | 7902 | 1980-12-17 | 800.00 | NULL | 20 |
| 7499 | ALLEN | SALESMAN | 7698 | 1981-02-20 | 1600.00 | 300.00 | 30 |
| 7521 | WARD | SALESMAN | 7698 | 1981-02-22 | 1250.00 | 500.00 | 30 |
| 7788 | SCOTT | ANALYST | 7566 | 1982-12-09 | 3000.00 | NULL | 20 |
| 7876 | ADAMS | CLERK | 7788 | 1983-01-12 | 1100.00 | NULL | 20 |

```
5 rows in set (0.00 sec)
```

Aggregate functions

12. find max sal and min sal for each job

```
mysql> select job,max(sal),min(sal) from emp
-> group by job;
```

| job | max(sal) | min(sal) |
|-----------|----------|----------|
| Manager | 5000.00 | 2450.00 |
| CLERK | 1300.00 | 800.00 |
| SALESMAN | 1600.00 | 1250.00 |
| ANALYST | 3000.00 | 3000.00 |
| PRESIDENT | 5000.00 | 5000.00 |

```
5 rows in set (0.00 sec)
```

13. find how many employees have not received commission

```
mysql> select comm,count(*) from emp
-> group by comm
-> having comm is null;
```

| comm | count(*) |
|------|----------|
| NULL | 10 |

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

14. find sum of sal of all employees working in dept no 10

```
mysql> select deptno,sum(sal) from emp
-> group by deptno
-> having deptno=10;
```

| deptno | sum(sal) |
|--------|----------|
| 10 | 8750.00 |

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

15. find maximum salary,average sal for each job in every department

```
mysql> select job,deptno,max(sal),avg(sal) from emp
-> group by job,deptno;
```

| job | deptno | max(sal) | avg(sal) |
|-----------|--------|----------|-------------|
| Manager | 30 | 5000.00 | 3925.000000 |
| CLERK | 20 | 1100.00 | 950.000000 |
| SALESMAN | 30 | 1600.00 | 1400.000000 |
| MANAGER | 20 | 2975.00 | 2975.000000 |
| MANAGER | 10 | 2450.00 | 2450.000000 |
| ANALYST | 20 | 3000.00 | 3000.000000 |
| PRESIDENT | 10 | 5000.00 | 5000.000000 |
| CLERK | 30 | 950.00 | 950.000000 |
| CLERK | 10 | 1300.00 | 1300.000000 |

9 rows in set (0.00 sec)

16. find max salary for every department if deptno is > 15 and arrange data in deptno order.

```
mysql> select deptno,max(sal) from emp
-> group by deptno
-> having deptno>15
-> order by deptno;
```

| deptno | max(sal) |
|--------|----------|
| 20 | 3000.00 |
| 30 | 5000.00 |

2 rows in set (0.00 sec)

17. find sum salary for every department if sum is > 3000

```
mysql> select deptno,sum(sal) from emp
-> group by deptno
-> having sum(sal)>3000;
```

| deptno | sum(sal) |
|--------|----------|
| 30 | 14400.00 |
| 20 | 10875.00 |
| 10 | 8750.00 |

3 rows in set (0.00 sec)

18. list all department which has minimum 5 employees

```
mysql> select deptno,count(*) from emp
-> group by deptno
-> having count(*)>=5;
+-----+-----+
| deptno | count(*) |
+-----+-----+
|      30 |         7 |
|      20 |         5 |
+-----+-----+
2 rows in set (0.00 sec)
```

19. count how many employees earn salary more than 2000 in each job

```
mysql> select job,sal,count(*) from emp
-> group by job,sal
-> having sal>2000;
+-----+-----+-----+
| job      | sal      | count(*) |
+-----+-----+-----+
| Manager  | 5000.00  |         1 |
| MANAGER  | 2975.00  |         1 |
| MANAGER  | 2850.00  |         1 |
| MANAGER  | 2450.00  |         1 |
| ANALYST  | 3000.00  |         2 |
| PRESIDENT | 5000.00  |         1 |
+-----+-----+-----+
6 rows in set (0.00 sec)
```

20. list all enames and jobs in small case letter

```
mysql> select lower(ename),lower(job) from emp;
+-----+-----+
| lower(ename) | lower(job) |
+-----+-----+
| tinku        | manager    |
| smith        | clerk      |
| allen        | salesman   |
| ward         | salesman   |
| jones        | manager    |
| martin       | salesman   |
| blake        | manager    |
| clark        | manager    |
| scott        | analyst    |
| king         | president  |
| turner       | salesman   |
| adams        | clerk      |
| james        | clerk      |
| ford         | analyst    |
| miller       | clerk      |
+-----+-----+
15 rows in set (0.02 sec)
```

21. list all names and jobs so that the length of name should be 15 if it is smaller then add spaces to left

```
mysql> select lpad(ename,15," "),lpad(job,15," ") from emp;
```

| lpad(ename,15," ") | lpad(job,15," ") |
|--------------------|------------------|
| Tinku | Manager |
| SMITH | CLERK |
| ALLEN | SALESMAN |
| WARD | SALESMAN |
| JONES | MANAGER |
| MARTIN | SALESMAN |
| BLAKE | MANAGER |
| CLARK | MANAGER |
| SCOTT | ANALYST |
| KING | PRESIDENT |
| TURNER | SALESMAN |
| ADAMS | CLERK |
| JAMES | CLERK |
| FORD | ANALYST |
| MILLER | CLERK |

```
15 rows in set (0.00 sec)
```

22. display min sal,max sal, average sal for all employees working under same manager

```
mysql> select mgr,min(sal),max(sal),avg(sal) from emp
-> group by mgr;
```

| mgr | min(sal) | max(sal) | avg(sal) |
|------|----------|----------|-------------|
| NULL | 5000.00 | 5000.00 | 5000.000000 |
| 7902 | 800.00 | 800.00 | 800.000000 |
| 7698 | 950.00 | 1600.00 | 1310.000000 |
| 7839 | 2450.00 | 2975.00 | 2758.333333 |
| 7566 | 3000.00 | 3000.00 | 3000.000000 |
| 7788 | 1100.00 | 1100.00 | 1100.000000 |
| 7782 | 1300.00 | 1300.00 | 1300.000000 |

```
7 rows in set (0.00 sec)
```

23. find sum of total earnings(sal+comm), average of sal+comm, for all employees who earn sal > 2000 and work in either dept no 10 or 20

```
mysql> select ename,deptno,sum(sal+ifnull(comm,0)),avg(sal+ifnull(comm,0)) from emp
-> group by ename,deptno
-> having sum(sal+ifnull(comm,0))>2000 and deptno in (10,20);
```

| ename | deptno | sum(sal+ifnull(comm,0)) | avg(sal+ifnull(comm,0)) |
|-------|--------|-------------------------|-------------------------|
| JONES | 20 | 2975.00 | 2975.000000 |
| CLARK | 10 | 2450.00 | 2450.000000 |
| SCOTT | 20 | 3000.00 | 3000.000000 |
| KING | 10 | 5000.00 | 5000.000000 |
| FORD | 20 | 3000.00 | 3000.000000 |

```
5 rows in set (0.00 sec)
```

24. list all employees who joined in Aug 1980 and salary is >1500 and < 2500

```
mysql> select * from emp
-> where hiredate between "1981-12-01" and "1981-12-31" and sal>500 and sal<4000;
+-----+-----+-----+-----+-----+-----+-----+-----+
| EMPNO | ENAME | JOB      | MGR | HIREDATE | SAL      | COMM | DEPTNO |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 7900   | JAMES | CLERK    | 7698 | 1981-12-03 | 950.00   | NULL | 30     |
| 7902   | FORD  | ANALYST  | 7566 | 1981-12-03 | 3000.00  | NULL | 20     |
+-----+-----+-----+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)
```

25. list all employees joined in either aug or may or dec

```
mysql> select * from emp
-> where month(hiredate) in (08,05,12);
+-----+-----+-----+-----+-----+-----+-----+-----+
| EMPNO | ENAME | JOB      | MGR | HIREDATE | SAL      | COMM | DEPTNO |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 7369   | SMITH | CLERK    | 7902 | 1980-12-17 | 800.00   | NULL | 20     |
| 7698   | BLAKE | MANAGER  | 7839 | 1981-05-01 | 2850.00  | NULL | 30     |
| 7788   | SCOTT | ANALYST  | 7566 | 1982-12-09 | 3000.00  | NULL | 20     |
| 7900   | JAMES | CLERK    | 7698 | 1981-12-03 | 950.00   | NULL | 30     |
| 7902   | FORD  | ANALYST  | 7566 | 1981-12-03 | 3000.00  | NULL | 20     |
+-----+-----+-----+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

26. display name and hiredate in dd/mm/yy format for all employees whose job is clerk and they earn some commission

```
mysql> select ename,date_format(hiredate,"%d/%m/%y") from emp
-> where job="Clerk" and comm is null;
+-----+-----+
| ename | date_format(hiredate,"%d/%m/%y") |
+-----+-----+
| SMITH | 17/12/80                          |
| ADAMS | 12/01/83                          |
| JAMES | 03/12/81                          |
| MILLER | 23/01/82                         |
+-----+-----+
4 rows in set (0.00 sec)
```

27. list empcode,empno,name and job for each employee. (note :empcode is 3 to 5 characters from name and last 2 characters of job)

```
mysql> select empno,ename,job,concat(left(ename,3),right(job,2)) empcode from emp;
```

| empno | ename | job | empcode |
|-------|--------|-----------|---------|
| 1000 | Tinku | Manager | Tiner |
| 7369 | SMITH | CLERK | SMIRK |
| 7499 | ALLEN | SALESMAN | ALLAN |
| 7521 | WARD | SALESMAN | WARAN |
| 7566 | JONES | MANAGER | JONER |
| 7654 | MARTIN | SALESMAN | MARAN |
| 7698 | BLAKE | MANAGER | BLAER |
| 7782 | CLARK | MANAGER | CLAER |
| 7788 | SCOTT | ANALYST | SCOST |
| 7839 | KING | PRESIDENT | KINNT |
| 7844 | TURNER | SALESMAN | TURAN |
| 7876 | ADAMS | CLERK | ADARK |
| 7900 | JAMES | CLERK | JAMRK |
| 7902 | FORD | ANALYST | FORST |
| 7934 | MILLER | CLERK | MILRK |

```
15 rows in set (0.00 sec)
```

29. Display empid,name,sal,comm,remark Remark should base on following conditions

comm >= 600 "excellent Keep it up"

if it < 600 or not null "good"

otherwise "Need improvement"

```
mysql> select empno,ename,sal,comm
-> ,case when comm>=600 then "Excellent Keep it up"
-> when comm<600 or comm is not null then "Good"
-> else
-> "Need improvement"
-> end Remark
-> from emp;
```

| empno | ename | sal | comm | Remark |
|-------|---------|----------|---------|----------------------|
| 1000 | Krishna | 4431.15 | 400.00 | Good |
| 1001 | Raj | 5859.38 | 100.00 | Good |
| 1002 | Aditya | 5859.38 | 200.00 | Good |
| 1111 | Vihan | 5371.10 | NULL | Need improvement |
| 7369 | SMITH | 2250.00 | NULL | Need improvement |
| 7499 | ALLEN | 4050.00 | 300.00 | Good |
| 7521 | WARD | 3164.06 | 500.00 | Good |
| 7566 | JONES | 7669.92 | NULL | Need improvement |
| 7654 | MARTIN | 3164.06 | 1400.00 | Excellent Keep it up |
| 7698 | BLAKE | 7347.66 | NULL | Need improvement |
| 7782 | CLARK | 6316.40 | NULL | Need improvement |
| 7788 | SCOTT | 8789.06 | NULL | Need improvement |
| 7839 | KING | 12656.26 | NULL | Need improvement |
| 7844 | TURNER | 3796.87 | 0.00 | Good |
| 7876 | ADAMS | 3093.74 | NULL | Need improvement |
| 7900 | JAMES | 2671.87 | NULL | Need improvement |
| 7902 | FORD | 8789.06 | NULL | Need improvement |
| 7934 | MILLER | 3656.26 | NULL | Need improvement |

```
18 rows in set (0.00 sec)
```

30. Display empid, name, deptno and department name by using following conditions.

```
dept 10 then "Hr"
if 20 then "Admin"
if 30 then "accounts"
otherwise purchase
```

```
mysql> select empno,ename,deptno,
-> case when deptno=10 then "HR"
-> when deptno=20 then "ADMIN"
-> when deptno=30 then "ACCOUNTS"
-> else "PURCHASE"
-> end Department
-> from emp;
```

| empno | ename | deptno | Department |
|-------|---------|--------|------------|
| 1000 | Krishna | 30 | ACCOUNTS |
| 1001 | Raj | 30 | ACCOUNTS |
| 1002 | Aditya | 30 | ACCOUNTS |
| 1111 | Vihan | NULL | PURCHASE |
| 7369 | SMITH | 20 | ADMIN |
| 7499 | ALLEN | 30 | ACCOUNTS |
| 7521 | WARD | 30 | ACCOUNTS |
| 7566 | JONES | 20 | ADMIN |
| 7654 | MARTIN | 30 | ACCOUNTS |
| 7698 | BLAKE | 30 | ACCOUNTS |
| 7782 | CLARK | 10 | HR |
| 7788 | SCOTT | 20 | ADMIN |
| 7839 | KING | 10 | HR |
| 7844 | TURNER | 30 | ACCOUNTS |
| 7876 | ADAMS | 20 | ADMIN |
| 7900 | JAMES | 30 | ACCOUNTS |
| 7902 | FORD | 20 | ADMIN |
| 7934 | MILLER | 10 | HR |

```
18 rows in set (0.00 sec)
```

Topic ----- create Table, DML , subquery and joins

31. Practice creating following tables

MySQL syntax:

create table mydept_DBDA

(

deptid int primary key,

dname varchar(20) not null unique,

dloc varchar(20)

);


```
mysql> create table mydept_DBDA
-> (
-> deptid int primary key,
-> dname varchar(20) not null unique,
-> dloc varchar(20)
-> )
-> ;
Query OK, 0 rows affected (0.40 sec)

mysql> insert into mydept_DBDA values(30,'Purchase','Mumbai');
Query OK, 1 row affected (0.07 sec)

mysql> select * from mydept_DBDA;
+-----+-----+-----+
| deptid | dname   | dloc   |
+-----+-----+-----+
|      30 | Purchase | Mumbai |
+-----+-----+-----+
1 row in set (0.00 sec)
```

Oracle syntax:

```
create table mydept_DBDA
(
deptid number primary key,
dname varchar2(20) not null unique,
dloc varchar2(20)
)
```

```
insert into mydept_DBDA values(30,'Purchase','Mumbai');
```

MySql syntax:

```
create table myemployee
(
empno int primary key,
fname varchar(15) not null,
mname varchar(15),
lname varchar(15) not null,
sal float(9,2) check(sal >=1000),
doj date,
passportnum varchar(15) unique,
deptno int,
constraint fk_deptno foreign key(deptno) references mydept_DBDA(deptid) on
delete set null
on update cascade
)
```

Oracle syntax:

```
create table myemployee
```

```
(
empno number(5) primary key,
fname varchar2(15) not null,
mname varchar2(15),
lname varchar2(15) not null,
sal number(9,2) check(sal >=1000),
doj date default sysdate,
passportnum varchar2(15) unique,
deptno number constraint fk_deptno references mydept_DBDA(deptid) on delete
cascade
)
```

```
mysql> create table myemployee
-> (
-> empno int primary key,
-> fname varchar(15) not null,
-> mname varchar(15),
-> lname varchar(15) not null,
-> sal float(9,2) check(sal >=1000),
-> doj date,
-> passportnum varchar(15) unique,
-> deptno int,
-> constraint fk_deptno foreign key(deptno) references mydept_DBDA(deptid) on
-> delete set null
-> on update cascade
-> )
-> ;
Query OK, 0 rows affected, 1 warning (0.15 sec)
```

32. Create following tables Student, Course

Student (sid,sname) ----- sid ---primary key

Course(cid,cname)----- cid ---primary key

Marks(studid,courseid,marks)

Sample data for marks table

studid,courseid,marks

1 1 99

1 3 98

2 1 95

2 2 97

create table marks(

studid number,

courseid number,

marks number,

constraint pk primary key(studid,courseid),

constraint fk_sid1 foreign key (studid) references student1(sid) on delete cascade,

constraint fk_cid1 foreign key (courseid) references course1(cid)

)

```
mysql> select * from student1;
+-----+-----+
| sid | sname |
+-----+-----+
| 1 | Raju |
| 2 | Sham |
+-----+-----+
2 rows in set (0.04 sec)

mysql> select * from course1;
+-----+-----+
| cid | cname |
+-----+-----+
| 1 | DAC |
| 2 | DBDA |
| 3 | DITIIS |
+-----+-----+
3 rows in set (0.01 sec)

mysql> select * from marks;
Empty set (0.01 sec)

mysql> insert into marks values(1,1,99);
Query OK, 1 row affected (0.02 sec)

mysql> insert into marks values(1,3,98);
Query OK, 1 row affected (0.01 sec)

mysql> insert into marks values(2,1,95);
Query OK, 1 row affected (0.01 sec)

mysql> insert into marks values(2,2,97);
Query OK, 1 row affected (0.01 sec)
```

```
mysql> select * from marks;
+-----+-----+-----+
| sid | cid | marks |
+-----+-----+-----+
| 1 | 1 | 99 |
| 1 | 3 | 98 |
| 2 | 1 | 95 |
| 2 | 2 | 97 |
+-----+-----+-----+
4 rows in set (0.00 sec)
```

33. Create empty table emp10 with table structure same as emp table.

create table emp10 as

(

```
select *
from emp
where 1=2;
)
```

34. Solve following using alter table
add primary key constraint on emp,dept,salgrade
emp ----> empno

dept---> deptno
salgrade---> grade
add foreign key constarint in emp
deptno --->> dept(deptno)
add new column in emp table netsal with constraint default 1000

35. Update employee sal ---- increase sal of each employee by 15 % sal +comm, change the job to

manager and mgr to 7777 for all employees in deptno 10.

36. change job of smith to senior clerk

```
mysql> update emp
-> set job="SenClerk"
-> where ename="Smith";
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

```
mysql> select * from emp;
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|---------|-----------|------|------------|----------|---------|--------|
| 1000 | Krishna | Manager | NULL | 2024-03-03 | 5095.82 | 400.00 | 30 |
| 1001 | Raj | Clerk | NULL | 2024-01-07 | 6738.29 | 100.00 | 30 |
| 1002 | Aditya | Clerk | NULL | 2021-06-01 | 6738.29 | 200.00 | 30 |
| 1111 | Vihan | Manager | NULL | 2019-08-23 | 5371.10 | NULL | NULL |
| 7369 | SMITH | SenClerk | 7902 | 1980-12-17 | 2250.00 | NULL | 20 |
| 7499 | ALLEN | SALESMAN | 7698 | 1981-02-20 | 3656.26 | 300.00 | 30 |
| 7521 | WARD | SALESMAN | 7698 | 1981-02-22 | 3638.67 | 500.00 | 30 |
| 7566 | JONES | MANAGER | 7839 | 1981-04-02 | 7669.92 | NULL | 20 |
| 7654 | MARTIN | SALESMAN | 7698 | 1981-09-28 | 3638.67 | 1400.00 | 30 |
| 7698 | BLAKE | MANAGER | 7839 | 1981-05-01 | 7347.66 | NULL | 30 |
| 7782 | CLARK | MANAGER | 7839 | 1981-06-09 | 6316.40 | NULL | 10 |
| 7788 | SCOTT | ANALYST | 7566 | 1982-12-09 | 8789.06 | NULL | 20 |
| 7839 | KING | PRESIDENT | NULL | 1981-11-17 | 12656.26 | NULL | 10 |
| 7844 | TURNER | SALESMAN | 7698 | 1981-09-08 | 4366.40 | 0.00 | 30 |
| 7876 | ADAMS | CLERK | 7788 | 1983-01-12 | 3093.74 | NULL | 20 |
| 7900 | JAMES | CLERK | 7698 | 1981-12-03 | 2671.87 | NULL | 30 |
| 7902 | FORD | ANALYST | 7566 | 1981-12-03 | 8789.06 | NULL | 20 |
| 7934 | MILLER | CLERK | 7782 | 1982-01-23 | 3656.26 | NULL | 10 |

```
18 rows in set (0.00 sec)
```

37. increase salary of all employees by 15% if they are earning some commission

```
mysql> update emp
-> set sal=0.15*sal+sal
-> where comm is not null;
Query OK, 7 rows affected, 6 warnings (0.02 sec)
Rows matched: 7 Changed: 7 Warnings: 6
```

```
mysql> select * from emp;
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|---------|-----------|------|------------|----------|---------|--------|
| 1000 | Krishna | Manager | NULL | 2024-03-03 | 5095.82 | 400.00 | 30 |
| 1001 | Raj | Clerk | NULL | 2024-01-07 | 6738.29 | 100.00 | 30 |
| 1002 | Aditya | Clerk | NULL | 2021-06-01 | 6738.29 | 200.00 | 30 |
| 1111 | Vihan | Manager | NULL | 2019-08-23 | 5371.10 | NULL | NULL |
| 7369 | SMITH | CLERK | 7902 | 1980-12-17 | 2250.00 | NULL | 20 |
| 7499 | ALLEN | SALESMAN | 7698 | 1981-02-20 | 4657.50 | 300.00 | 30 |
| 7521 | WARD | SALESMAN | 7698 | 1981-02-22 | 3638.67 | 500.00 | 30 |
| 7566 | JONES | MANAGER | 7839 | 1981-04-02 | 7669.92 | NULL | 20 |
| 7654 | MARTIN | SALESMAN | 7698 | 1981-09-28 | 3638.67 | 1400.00 | 30 |
| 7698 | BLAKE | MANAGER | 7839 | 1981-05-01 | 7347.66 | NULL | 30 |
| 7782 | CLARK | MANAGER | 7839 | 1981-06-09 | 6316.40 | NULL | 10 |
| 7788 | SCOTT | ANALYST | 7566 | 1982-12-09 | 8789.06 | NULL | 20 |
| 7839 | KING | PRESIDENT | NULL | 1981-11-17 | 12656.26 | NULL | 10 |
| 7844 | TURNER | SALESMAN | 7698 | 1981-09-08 | 4366.40 | 0.00 | 30 |
| 7876 | ADAMS | CLERK | 7788 | 1983-01-12 | 3093.74 | NULL | 20 |
| 7900 | JAMES | CLERK | 7698 | 1981-12-03 | 2671.87 | NULL | 30 |
| 7902 | FORD | ANALYST | 7566 | 1981-12-03 | 8789.06 | NULL | 20 |
| 7934 | MILLER | CLERK | 7782 | 1982-01-23 | 3656.26 | NULL | 10 |

```
18 rows in set (0.00 sec)
```

38. list all employees with sal>smith's sal

```
mysql> select * from emp
-> where sal>(select sal from emp where ename="Smith");
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|---------|-----------|------|------------|----------|---------|--------|
| 1000 | Krishna | Manager | NULL | 2024-03-03 | 5095.82 | 400.00 | 30 |
| 1001 | Raj | Clerk | NULL | 2024-01-07 | 6738.29 | 100.00 | 30 |
| 1002 | Aditya | Clerk | NULL | 2021-06-01 | 6738.29 | 200.00 | 30 |
| 1111 | Vihan | Manager | NULL | 2019-08-23 | 5371.10 | NULL | NULL |
| 7499 | ALLEN | SALESMAN | 7698 | 1981-02-20 | 4657.50 | 300.00 | 30 |
| 7521 | WARD | SALESMAN | 7698 | 1981-02-22 | 3638.67 | 500.00 | 30 |
| 7566 | JONES | MANAGER | 7839 | 1981-04-02 | 7669.92 | NULL | 20 |
| 7654 | MARTIN | SALESMAN | 7698 | 1981-09-28 | 3638.67 | 1400.00 | 30 |
| 7698 | BLAKE | MANAGER | 7839 | 1981-05-01 | 7347.66 | NULL | 30 |
| 7782 | CLARK | MANAGER | 7839 | 1981-06-09 | 6316.40 | NULL | 10 |
| 7788 | SCOTT | ANALYST | 7566 | 1982-12-09 | 8789.06 | NULL | 20 |
| 7839 | KING | PRESIDENT | NULL | 1981-11-17 | 12656.26 | NULL | 10 |
| 7844 | TURNER | SALESMAN | 7698 | 1981-09-08 | 4366.40 | 0.00 | 30 |
| 7876 | ADAMS | CLERK | 7788 | 1983-01-12 | 3093.74 | NULL | 20 |
| 7900 | JAMES | CLERK | 7698 | 1981-12-03 | 2671.87 | NULL | 30 |
| 7902 | FORD | ANALYST | 7566 | 1981-12-03 | 8789.06 | NULL | 20 |
| 7934 | MILLER | CLERK | 7782 | 1982-01-23 | 3656.26 | NULL | 10 |

```
17 rows in set (0.00 sec)
```

39. list all employees who are working in smith's department

```
mysql> select * from emp
-> where deptno=(select deptno from emp where ename="SMITH");
+-----+-----+-----+-----+-----+-----+-----+-----+
| EMPNO | ENAME | JOB      | MGR  | HIREDATE   | SAL      | COMM | DEPTNO |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 7369 | SMITH | CLERK    | 7902 | 1980-12-17 | 2250.00 | NULL | 20     |
| 7566 | JONES | MANAGER  | 7839 | 1981-04-02 | 7669.92 | NULL | 20     |
| 7788 | SCOTT | ANALYST  | 7566 | 1982-12-09 | 8789.06 | NULL | 20     |
| 7876 | ADAMS | CLERK    | 7788 | 1983-01-12 | 3093.74 | NULL | 20     |
| 7902 | FORD  | ANALYST  | 7566 | 1981-12-03 | 8789.06 | NULL | 20     |
+-----+-----+-----+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

40. list all employees with sal < Blake's sal and salary > krishna's sal

```
mysql> select * from emp
-> where sal<(select sal from emp where ename="BLAKE") and sal>(select sal from emp
where ename="KRISHNA");
+-----+-----+-----+-----+-----+-----+-----+-----+
| EMPNO | ENAME | JOB      | MGR  | HIREDATE   | SAL      | COMM | DEPTNO |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 1001 | Raj   | Clerk    | NULL | 2024-01-07 | 6738.29 | 100.00 | 30     |
| 1002 | Aditya | Clerk    | NULL | 2021-06-01 | 6738.29 | 200.00 | 30     |
| 1111 | Vihan | Manager  | NULL | 2019-08-23 | 5371.10 | NULL | NULL   |
| 7782 | CLARK | MANAGER  | 7839 | 1981-06-09 | 6316.40 | NULL | 10     |
+-----+-----+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

41. delete all employees working in alan's department

42. change salary of Alan to the salary of Miller.

```
mysql> update emp e
-> set sal=(select sal from (select sal from emp p where p.ename="MILLER") p)
-> where e.ename="Allen";
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select ename,sal from emp
-> where ename in ("ALLEN","MILLER");
+-----+-----+
| ename | sal      |
+-----+-----+
| ALLEN | 3656.26 |
| MILLER | 3656.26 |
+-----+-----+
2 rows in set (0.00 sec)
```

44. list all employees with salary > either Smith's salary or alan's sal

```
mysql> select * from emp
-> where sal>any(select sal from emp where ename in ("SMITH","ALLEN"));
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|---------|-----------|------|------------|----------|---------|--------|
| 1000 | Krishna | Manager | NULL | 2024-03-03 | 5095.82 | 400.00 | 30 |
| 1001 | Raj | Clerk | NULL | 2024-01-07 | 6738.29 | 100.00 | 30 |
| 1002 | Aditya | Clerk | NULL | 2021-06-01 | 6738.29 | 200.00 | 30 |
| 1111 | Vihan | Manager | NULL | 2019-08-23 | 5371.10 | NULL | NULL |
| 7499 | ALLEN | SALESMAN | 7698 | 1981-02-20 | 3656.26 | 300.00 | 30 |
| 7521 | WARD | SALESMAN | 7698 | 1981-02-22 | 3638.67 | 500.00 | 30 |
| 7566 | JONES | MANAGER | 7839 | 1981-04-02 | 7669.92 | NULL | 20 |
| 7654 | MARTIN | SALESMAN | 7698 | 1981-09-28 | 3638.67 | 1400.00 | 30 |
| 7698 | BLAKE | MANAGER | 7839 | 1981-05-01 | 7347.66 | NULL | 30 |
| 7782 | CLARK | MANAGER | 7839 | 1981-06-09 | 6316.40 | NULL | 10 |
| 7788 | SCOTT | ANALYST | 7566 | 1982-12-09 | 8789.06 | NULL | 20 |
| 7839 | KING | PRESIDENT | NULL | 1981-11-17 | 12656.26 | NULL | 10 |
| 7844 | TURNER | SALESMAN | 7698 | 1981-09-08 | 4366.40 | 0.00 | 30 |
| 7876 | ADAMS | CLERK | 7788 | 1983-01-12 | 3093.74 | NULL | 20 |
| 7900 | JAMES | CLERK | 7698 | 1981-12-03 | 2671.87 | NULL | 30 |
| 7902 | FORD | ANALYST | 7566 | 1981-12-03 | 8789.06 | NULL | 20 |
| 7934 | MILLER | CLERK | 7782 | 1982-01-23 | 3656.26 | NULL | 10 |

17 rows in set (0.00 sec)

45. list all employees who earn more than average sal of dept 10

```
mysql> select * from emp
-> where sal>(select avg(sal) from emp where deptno=10);
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|-------|-----------|------|------------|----------|------|--------|
| 7566 | JONES | MANAGER | 7839 | 1981-04-02 | 7669.92 | NULL | 20 |
| 7788 | SCOTT | ANALYST | 7566 | 1982-12-09 | 8789.06 | NULL | 20 |
| 7839 | KING | PRESIDENT | NULL | 1981-11-17 | 12656.26 | NULL | 10 |
| 7902 | FORD | ANALYST | 7566 | 1981-12-03 | 8789.06 | NULL | 20 |

4 rows in set (0.00 sec)

46. list all employees who earn more than average sal of Alan's department

```
mysql> select * from emp
-> where sal > (select avg(sal) from emp where deptno = (select deptno from emp where ename = "Allen"));
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|---------|-----------|------|------------|----------|--------|--------|
| 1000 | Krishna | Manager | NULL | 2024-03-03 | 5095.82 | 400.00 | 30 |
| 1001 | Raj | Clerk | NULL | 2024-01-07 | 6738.29 | 100.00 | 30 |
| 1002 | Aditya | Clerk | NULL | 2021-06-01 | 6738.29 | 200.00 | 30 |
| 1111 | Vihan | Manager | NULL | 2019-08-23 | 5371.10 | NULL | NULL |
| 7566 | JONES | MANAGER | 7839 | 1981-04-02 | 7669.92 | NULL | 20 |
| 7698 | BLAKE | MANAGER | 7839 | 1981-05-01 | 7347.66 | NULL | 30 |
| 7782 | CLARK | MANAGER | 7839 | 1981-06-09 | 6316.40 | NULL | 10 |
| 7788 | SCOTT | ANALYST | 7566 | 1982-12-09 | 8789.06 | NULL | 20 |
| 7839 | KING | PRESIDENT | NULL | 1981-11-17 | 12656.26 | NULL | 10 |
| 7902 | FORD | ANALYST | 7566 | 1981-12-03 | 8789.06 | NULL | 20 |

10 rows in set (0.00 sec)

47. list all employees who are working in accounting department

```
mysql> select * from emp
-> where deptno = (select deptno from dept where dname = "Accounting");
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|--------|-----------|------|------------|----------|------|--------|
| 7782 | CLARK | MANAGER | 7839 | 1981-06-09 | 6316.40 | NULL | 10 |
| 7839 | KING | PRESIDENT | NULL | 1981-11-17 | 12656.26 | NULL | 10 |
| 7934 | MILLER | CLERK | 7782 | 1982-01-23 | 3656.26 | NULL | 10 |

3 rows in set (0.00 sec)

48. list all employees who earn more than average salary of their own department

```
mysql> select * from emp e
-> where sal > (select avg(sal) from emp p where p.deptno = e.deptno);
```

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|-------|---------|-----------|------|------------|----------|--------|--------|
| 1000 | Krishna | Manager | NULL | 2024-03-03 | 5095.82 | 400.00 | 30 |
| 1001 | Raj | Clerk | NULL | 2024-01-07 | 6738.29 | 100.00 | 30 |
| 1002 | Aditya | Clerk | NULL | 2021-06-01 | 6738.29 | 200.00 | 30 |
| 7566 | JONES | MANAGER | 7839 | 1981-04-02 | 7669.92 | NULL | 20 |
| 7698 | BLAKE | MANAGER | 7839 | 1981-05-01 | 7347.66 | NULL | 30 |
| 7788 | SCOTT | ANALYST | 7566 | 1982-12-09 | 8789.06 | NULL | 20 |
| 7839 | KING | PRESIDENT | NULL | 1981-11-17 | 12656.26 | NULL | 10 |
| 7902 | FORD | ANALYST | 7566 | 1981-12-03 | 8789.06 | NULL | 20 |

8 rows in set (0.00 sec)

49. list all employees who earn sal < than their managers salary

50. list all employees who are earning more than average salary of their job

51. display employee name and department


```
mysql> select e.ename,d.dname from emp e inner join dept d on e.deptno=d.deptno;
```

| ename | dname |
|---------|------------|
| Krishna | SALES |
| Raj | SALES |
| Aditya | SALES |
| SMITH | RESEARCH |
| ALLEN | SALES |
| WARD | SALES |
| JONES | RESEARCH |
| MARTIN | SALES |
| BLAKE | SALES |
| CLARK | ACCOUNTING |
| SCOTT | RESEARCH |
| KING | ACCOUNTING |
| TURNER | SALES |
| ADAMS | RESEARCH |
| JAMES | SALES |
| FORD | RESEARCH |
| MILLER | ACCOUNTING |

```
17 rows in set (0.00 sec)
```

52. display empno,name,department name and grade (use emp,dept and salgrade table)

53. list all employees number,name, mgrno and manager name

```
mysql> select e.empno,e.ename,e.mgr,m.ename
-> from emp e,emp m
-> where e.mgr=m.empno;
```

| empno | ename | mgr | ename |
|-------|--------|------|-------|
| 7369 | SMITH | 7902 | FORD |
| 7499 | ALLEN | 7698 | BLAKE |
| 7521 | WARD | 7698 | BLAKE |
| 7566 | JONES | 7839 | KING |
| 7654 | MARTIN | 7698 | BLAKE |
| 7698 | BLAKE | 7839 | KING |
| 7782 | CLARK | 7839 | KING |
| 7788 | SCOTT | 7566 | JONES |
| 7844 | TURNER | 7698 | BLAKE |
| 7876 | ADAMS | 7788 | SCOTT |
| 7900 | JAMES | 7698 | BLAKE |
| 7902 | FORD | 7566 | JONES |
| 7934 | MILLER | 7782 | CLARK |

```
13 rows in set (0.00 sec)
```

54. create following tables and solve following questions(primary keys are marked in yellow)

foreign keys are marked in green

product(pid,pname,price,qty,cid,sid)

salesman (sid,sname,address)

category(cid,cnam,description)

1. list all product name,their category name and name of a person, who sold that product

```
mysql> select p.pname,c.cname,s.sname
-> from product p inner join category c on c.cid=p.cid
-> inner join salesman s on s.sid=p.sid;
+-----+-----+
| pname | cname | sname |
+-----+-----+
| Biscuits | Snack | Rahul |
| Dairy milk | Candy | Aditya |
| LAYS | Chips | Ved |
| Maggie | Snack | Ved |
+-----+-----+
4 rows in set (0.00 sec)
```

2. list all product name and salesman name for all salesman who stays in pune

```
mysql> select p.pname,s.sname from
-> product p inner join salesman s on s.sid=p.sid
-> where s.address="PUNE";
+-----+-----+
| pname | sname |
+-----+-----+
| Biscuits | Rahul |
| Dairy milk | Aditya |
+-----+-----+
2 rows in set (0.00 sec)
```

3. list all product name and category name

```
mysql> select p.pname,c.cname
-> from product p inner join category c on c.cid=p.cid;
+-----+-----+
| pname | cname |
+-----+-----+
| Dairy milk | Candy |
| LAYS | Chips |
| Maggie | Snack |
| Biscuits | Snack |
+-----+-----+
4 rows in set (0.00 sec)
```

55. create following tables and solve following questions(primary keys are marked in yellow)
foreign keys are marked in green

faculty(fid,fname,sp.skill1,sp.skill2)

courses(cid,cname,rld,fid)

room(roomid,rname,rloc)

faculty

fid fname spskill1 spskill2

10 kjzhcjh a b

11 sdd x z

12 lksjk a x
13 ksdjlkj a b

courses
cid cname rid fid
121 DBDA 100 10
131 DAC 101
141 DTISS
151 DIOT 105 12

Room
roomid rname rloc
100 jasmin 1st floor
101 Rose 2nd floor
105 Lotus 1st floor
103 Mogra 1st floor

```
mysql> select * from room;
+-----+-----+-----+
| roomid | rname  | loc      |
+-----+-----+-----+
| 100    | Jasmin | 1st floor |
| 101    | Rose   | 2nd floor |
| 102    | Lotus  | 1st floor |
| 103    | Mogra  | 1st floor |
+-----+-----+-----+
4 rows in set (0.01 sec)

mysql> select * from faculty;
+-----+-----+-----+-----+
| fid | fname    | skill1 | skill2 |
+-----+-----+-----+-----+
| 10  | Narendra | a       | b       |
| 11  | Nilima   | x       | z       |
| 12  | Shilpa   | a       | x       |
| 13  | Aditya   | a       | b       |
+-----+-----+-----+-----+
4 rows in set (0.01 sec)

mysql> select * from course;
+-----+-----+-----+-----+
| cid | cname   | rid  | fid  |
+-----+-----+-----+-----+
| 121 | DBDA    | 100  | 10   |
| 131 | DAC     | 101  | NULL |
| 141 | DTISS   | NULL | NULL |
| 151 | DIOT    | 102  | 12   |
+-----+-----+-----+-----+
4 rows in set (0.01 sec)
```

1. list all courses for which no room is assigned and all rooms for which are

Available

```
mysql> select cname,rname
-> from course c right join room r
-> on c.rid=r.roomid;
+-----+-----+
| cname | rname |
+-----+-----+
| DBDA  | Jasmin |
| DAC   | Rose   |
| DIOT  | Lotus  |
| NULL  | Mogra  |
+-----+-----+
4 rows in set (0.00 sec)
```

2. list all faculties who are not allocated to any course and rooms which are not allocated to any course

```
mysql> select * from course c right join faculty f on f.fid=c.fid right join room r
-> on r.roomid=c.rid;
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| cid | cname | rid | fid | fid | fname | skill1 | skill2 | roomid | rname | loc |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 121 | DBDA  | 100 | 10  | 10  | Narendra | a      | b      | 100  | Jasmin | 1st floor |
| NULL | NULL  | NULL | NULL | NULL | NULL    | NULL   | NULL   | 101  | Rose   | 2st floor |
| 151 | DIOT  | 102 | 12  | 12  | Shilpa  | a      | x      | 102  | Lotus  | 1st floor |
| NULL | NULL  | NULL | NULL | NULL | NULL    | NULL   | NULL   | 103  | Mogra  | 1st floor |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

3. list all rooms which are allocated or not allocated to any courses

```
mysql> select * from room r left join course c on c.rid=r.roomid;
+-----+-----+-----+-----+-----+-----+-----+
| roomid | rname | loc | cid | cname | rid | fid |
+-----+-----+-----+-----+-----+-----+-----+
| 100    | Jasmin | 1st floor | 121 | DBDA  | 100 | 10  |
| 101    | Rose   | 2st floor | 131 | DAC   | 101 | NULL |
| 102    | Lotus  | 1st floor | 151 | DIOT  | 102 | 12  |
| 103    | Mogra  | 1st floor | NULL | NULL  | NULL | NULL |
+-----+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

4. list all rooms which are not allocated to any courses

```
mysql> select * from room r
-> where not exists (select * from course c where c.rid=r.roomid);
+-----+-----+-----+
| roomid | rname | loc |
+-----+-----+-----+
| 103    | Mogra | 1st floor |
+-----+-----+-----+
1 row in set (0.01 sec)
```

5. display courses and faculty assigned to those courses whose special skill is

A

```
mysql> select * from faculty f inner join course c on c.fid=f.fid
-> where skill1="a";
```

| fid | fname | skill1 | skill2 | cid | cname | rid | fid |
|-----|----------|--------|--------|-----|-------|-----|-----|
| 10 | Narendra | a | b | 121 | DBDA | 100 | 10 |
| 12 | Shilpa | a | x | 151 | DIOT | 102 | 12 |

```
2 rows in set (0.00 sec)
```

56. create following tables with given constraints

product--- qty >0, default 20.00,pname not null and unique

prodid pname qty price catid sid

123 lays 30 30.00 1 12

111 pepsi 40 50.00 4 11

134 nachos 50 50.00 1 12

124 dairy milk 40 60.00 2 14

124 pringles 40 60.00 1 14

saleman ----- sname -----not null

sid sname city

11 Rahul Pune

12 Kirti Mumbai

13 Prasad Nashik

14 Arnav Amaravati

category ---- cname unique and not null

cid cname description

1 chips very crunchy

2 chocolate very chocolaty

3 snacks yummy

4 cold drinks thanda thanda cool cool

1. List all products with category chips

```
mysql> select * from product
-> where cid=(select cid from category where cname="CHIPS");
```

| pid | pname | qty | price | cid | sid |
|-----|-------|-----|-------|-----|------|
| 2 | LAYS | 2 | 24.00 | 10 | 1003 |

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

2. display all products sold by aditya

```
mysql> select * from product
      -> where sid=(select sid from salesman where sname="Aditya");
```

| pid | pname | qty | price | cid | sid |
|-----|------------|-----|--------|-----|------|
| 1 | Dairy milk | 21 | 234.00 | 20 | 1001 |

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

3. display all salesman who do not sold any product

```
mysql> select * from salesman s
      -> where not exists (select * from product p where p.sid=s.sid);
```

| sid | sname | address |
|------|-------|---------|
| 1005 | Ram | Nagpur |

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

4. display all category for which no product is there

```
mysql> select * from category c
      -> where not exists (select * from product p where p.cid=c.cid);
```

| cid | cname | description |
|-----|-------|-------------|
| 111 | Juice | Refreshing |

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

5. display all products with no category assigned

```
mysql> select * from product p
      -> where not exists(select * from category c where c.cid=p.cid);
```

| pid | pname | qty | price | cid | sid |
|-----|-------|-----|---------|------|------|
| 5 | Sugar | 23 | 2345.00 | NULL | 1003 |

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

6. list all salesman who stays in city with name starts with P or N

```
mysql> select * from salesman
      -> where address regexp "^P|^N";
```

| sid | sname | address |
|------|--------|---------|
| 1000 | Rahul | Pune |
| 1001 | Aditya | Pune |
| 1005 | Ram | Nagpur |

```
3 rows in set (0.03 sec)
```

7. add new column in salesman table by name credit limit

```
mysql> alter table salesman
-> add credit_limit float(9,2);
Query OK, 0 rows affected, 1 warning (0.17 sec)
Records: 0 Duplicates: 0 Warnings: 1

mysql> select * from salesman;
+-----+-----+-----+-----+
| sid  | sname | address | credit_limit |
+-----+-----+-----+-----+
| 1000 | Rahul | Pune    | NULL         |
| 1001 | Aditya | Pune    | NULL         |
| 1003 | Ved    | Mumbai  | NULL         |
| 1005 | Ram    | Nagpur   | NULL         |
+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

MONGODB ASSIGNMENT

1. Write a MongoDB query to display all the documents in the collection restaurants

```
iacsd0324> db.restaurent.find()
[
  {
    _id: ObjectId('660fbe23a8f1fabf8a9fa297'),
    address: {
      building: '2780',
      coord: [ -73.98241999999999, 40.579505 ],
      street: 'Stillwell Avenue',
      zipcode: '11224'
    },
    borough: 'Brooklyn',
    cuisine: 'American',
    grades: [
      {
        date: ISODate('2014-06-10T00:00:00.000Z'),
        grade: 'A',
        score: 5
      },
      {
        date: ISODate('2013-06-05T00:00:00.000Z'),
        grade: 'A',
        score: 7
      },
      {
        date: ISODate('2012-04-13T00:00:00.000Z'),
        grade: 'A',
        score: 12
      }
    ]
  }
]
```

2. Write a MongoDB query to display the fields restaurant_id, name, borough and cuisine for all the documents in the collection restaurant.


```

iacsd0324> db.restaurent.find({}, {restaurant_id:1, name:1, borough:1, cuisine:1})
[
  {
    _id: ObjectId('660fbe23a8f1fabf8a9fa297'),
    borough: 'Brooklyn',
    cuisine: 'American',
    name: 'Riviera Caterer',
    restaurant_id: '40356018'
  },
  {
    _id: ObjectId('660fbe23a8f1fabf8a9fa298'),
    borough: 'Bronx',
    cuisine: 'Bakery',
    name: 'Morris Park Bake Shop',
    restaurant_id: '30075445'
  },
  {
    _id: ObjectId('660fbe23a8f1fabf8a9fa299'),
    borough: 'Brooklyn',

```

3. Write a MongoDB query to display the fields restaurant_id, name, borough and cuisine, but exclude the field _id for all the documents in the collection restaurant.

```

iacsd0324> db.restaurent.find({}, {restaurant_id:1, name:1, borough:1, cuisine:1, _id:0})
[
  {
    borough: 'Brooklyn',
    cuisine: 'American',
    name: 'Riviera Caterer',
    restaurant_id: '40356018'
  },
  {
    borough: 'Bronx',
    cuisine: 'Bakery',
    name: 'Morris Park Bake Shop',
    restaurant_id: '30075445'
  },
  {
    borough: 'Brooklyn',
    cuisine: 'Hamburgers',
    name: "Wendy'S",
    restaurant_id: '30112340'
  },
  {
    borough: 'Queens',
    cuisine: 'Jewish/Kosher',
    name: 'Tov Kosher Kitchen',
    restaurant_id: '40356068'
  }
]

```

4. Write a MongoDB query to display the fields restaurant_id, name, borough and zip code, but exclude the field _id for all the documents in the collection restaurant.

```

iacsd0324> db.restaurent.find({}, {restaurant_id:1, name:1, borough:1, cuisine:1, 'address.zipcode':1, _id:0})
[
  {
    address: { zipcode: '11224' },
    borough: 'Brooklyn',
    cuisine: 'American',
    name: 'Riviera Caterer',
    restaurant_id: '40356018'
  },
  {
    address: { zipcode: '10462' },
    borough: 'Bronx',
    cuisine: 'Bakery',
    name: 'Morris Park Bake Shop',
    restaurant_id: '30075445'
  },
  {
    address: { zipcode: '10024' },
    borough: 'Bronx',
    cuisine: 'Hamburgers',
    name: 'Wendy's',
    restaurant_id: '30112340'
  }
]

```

5. Write a MongoDB query to display all the restaurant which is in the borough Bronx

```

iacsd0324> db.restaurent.find({borough:"Bronx"})
[
  {
    _id: ObjectId('660fbe23a8f1fabf8a9fa298'),
    address: {
      building: '1007',
      coord: [ -73.856077, 40.848447 ],
      street: 'Morris Park Ave',
      zipcode: '10462'
    },
    borough: 'Bronx',
    cuisine: 'Bakery',
    grades: [
      {
        date: ISODate('2014-03-03T00:00:00.000Z'),
        grade: 'A',
        score: 2
      },
      {
        date: ISODate('2013-09-11T00:00:00.000Z'),
        grade: 'A',
        score: 6
      },
      {
        date: ISODate('2013-01-24T00:00:00.000Z'),
        grade: 'A',
        score: 10
      },
      {
        date: ISODate('2014-11-03T00:00:00.000Z'),
        grade: 'A',
        score: 10
      }
    ]
  }
]

```

6. Write a MongoDB query to display the first 5 restaurant which is in the borough Bronx.

```

iacsd0324> db.restaurent.find({borough:"Bronx"}).limit(5)
[
  {
    _id: ObjectId('660fbe23a8f1fabf8a9fa298'),
    address: {
      building: '1007',
      coord: [ -73.856077, 40.848447 ],
      street: 'Morris Park Ave',
      zipcode: '10462'
    },
    borough: 'Bronx',
    cuisine: 'Bakery',
    grades: [
      {
        date: ISODate('2014-03-03T00:00:00.000Z'),
        grade: 'A',
        score: 2
      },
      {
        date: ISODate('2013-09-11T00:00:00.000Z'),
        grade: 'A',
        score: 6
      },
      {
        date: ISODate('2013-01-24T00:00:00.000Z'),
        grade: 'A',

```

7. Write a MongoDB query to display the next 5 restaurants after skipping first 5 which are in the borough Bronx.

```

acsd0324> db.restaurent.find({borough:"Bronx"}).skip(5).limit(5)

{
  _id: ObjectId('660fbe23a8f1fabf8a9fa2d7'),
  address: {
    building: '658',
    coord: [ -73.81363999999999, 40.829411000000001 ],
    street: 'Clarence Ave',
    zipcode: '10465'
  },
  borough: 'Bronx',
  cuisine: 'American',
  grades: [
    {
      date: ISODate('2014-06-21T00:00:00.000Z'),
      grade: 'A',
      score: 5
    },
    {
      date: ISODate('2012-07-11T00:00:00.000Z'),
      grade: 'A',
      score: 10
    }
  ],
  name: 'Manhem Club',
  restaurant_id: '40364363'
},

```

8. Write a MongoDB query to find the restaurants who achieved a score more than 90.

```

iacsd0324> db.restaurent.find({'grades.score':{$gt:90}})
[
  {
    _id: ObjectId('660fbe23a8f1fabf8a9fa3f3'),
    address: {
      building: '65',
      coord: [ -73.9782725, 40.7624022 ],
      street: 'West 54 Street',
      zipcode: '10019'
    },
    borough: 'Manhattan',
    cuisine: 'American',
    grades: [
      {
        date: ISODate('2014-08-22T00:00:00.000Z'),
        grade: 'A',
        score: 11
      },
      {
        date: ISODate('2014-03-28T00:00:00.000Z'),
        grade: 'C',
        score: 131
      },
      {

```

9. Write a MongoDB query to find the restaurants that achieved a score, more than 80 but less than 100.

```

iacsd0324> db.restaurent.find({'grades.score':{'$gt:80,$lt:100}})
[
  {
    _id: ObjectId('660fbe23a8f1fabf8a9fa3f3'),
    address: {
      building: '65',
      coord: [ -73.9782725, 40.7624022 ],
      street: 'West 54 Street',
      zipcode: '10019'
    },
    borough: 'Manhattan',
    cuisine: 'American',
    grades: [
      {
        date: ISODate('2014-08-22T00:00:00.000Z'),
        grade: 'A',
        score: 11
      },
      {
        date: ISODate('2014-03-28T00:00:00.000Z'),
        grade: 'C',
        score: 131
      },
      {
        date: ISODate('2013-09-25T00:00:00.000Z'),
        grade: 'A'
      }
    ]
  }
]

```

10. Write a MongoDB query to find the restaurants which locate in latitude value less than - 95.754168.

```

iacsd0324> db.restaurent.find({'address.coord.0':{'$lt:-95.754168'}})
[
  {
    _id: ObjectId('660fbe23a8f1fabf8a9fa8de'),
    address: {
      building: '3707',
      coord: [ -101.8945214, 33.5197474 ],
      street: '82 Street',
      zipcode: '11372'
    },
    borough: 'Queens',
    cuisine: 'American',
    grades: [
      {
        date: ISODate('2014-06-04T00:00:00.000Z'),
        grade: 'A',
        score: 12
      },
      {
        date: ISODate('2013-11-07T00:00:00.000Z'),
        grade: 'B',
        score: 19
      }
    ],
  },
]

```

11. Write a MongoDB query to find the restaurants that do not prepare any cuisine of 'American' and their grade score more than 70 and latitude less than -65.754168.

```

iacsd0324> db.restaurent.find({'cuisine':{'$ne: 'American'},'grades.score':{'$gt:70'}, 'address.coord':{'$lt:-65.754168'}})
[
  {
    _id: ObjectId('660fbe23a8f1fabf8a9fa495'),
    address: {
      building: '345',
      coord: [ -73.9864626, 40.7266739 ],
      street: 'East 6 Street',
      zipcode: '10003'
    },
    borough: 'Manhattan',
    cuisine: 'Indian',
    grades: [
      {
        date: ISODate('2014-09-15T00:00:00.000Z'),
        grade: 'A',
        score: 5
      }
    ],
  },
]

```

12. Write a MongoDB query to find the restaurants which do not prepare any cuisine of 'American' and achieved a score more than 70 and located in the longitude less than -

65.754168.

```
iacsd0324> db.restaurent.find({cuisine:{$ne: 'American'}, 'grades.score':{$gt:70}, address.coord:{$lt:-65.754168}})
[
  {
    _id: ObjectId('660fbe23a8f1fabf8a9fa495'),
    address: {
      building: '345',
      coord: [ -73.9864626, 40.7266739 ],
      street: 'East 6 Street',
      zipcode: '10003'
    },
    borough: 'Manhattan',
    cuisine: 'Indian',
    grades: [
      {
        date: ISODate('2014-09-15T00:00:00.000Z'),
        grade: 'A',
        score: 5
      }
    ]
  },
]
```

13. Write a MongoDB query to find the restaurants which do not prepare any cuisine of 'American ' and achieved a grade point 'A' not belongs to the borough Brooklyn. The document must be displayed according to the cuisine in descending order.

```
Type "it" for more
iacsd0324> db.restaurent.find({cuisine:{$ne:"American"}, 'grades.grade':"A",borough:{$ne:"Brooklyn"}}).sort({cuisine:-1})
[
  {
    _id: ObjectId('660fbe23a8f1fabf8a9fa9a1'),
    address: {
      building: '89',
      coord: [ -73.9995899, 40.7168015 ],
      street: 'Baxter Street',
      zipcode: '10013'
    },
    borough: 'Manhattan',
    cuisine: 'Vietnamese/Cambodian/Malaysia',
    grades: [
      {
        date: ISODate('2014-08-21T00:00:00.000Z'),
        grade: 'A',
        score: 13
      },
      {
        date: ISODate('2013-08-31T00:00:00.000Z'),
        grade: 'A',
        score: 13
      },
      {
        date: ISODate('2013-04-11T00:00:00.000Z'),
        grade: 'C',

```

14. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which contain 'Wil' as first three letters for its name.


```

iacsd0324> db.restaurent.find({name:/^Wil/i},{restaurant_id:1,name:1,borough:1,cuisine:1
,_id:0})
[
  {
    borough: 'Brooklyn',
    cuisine: 'Delicatessen',
    name: "Wilken'S Fine Food",
    restaurant_id: '40356483'
  },
  {
    borough: 'Bronx',
    cuisine: 'American',
    name: 'Wild Asia',
    restaurant_id: '40357217'
  },
  {
    borough: 'Bronx',
    cuisine: 'Pizza',
    name: 'Wilbel Pizza',
    restaurant_id: '40871979'
  },
  {
    borough: 'Manhattan',
    cuisine: 'Seafood',
    name: 'Wild Edibles',
    restaurant_id: '40928482'
  },
  {
    borough: 'Brooklyn',
    cuisine: 'Bagels/Pretzels',
    name: 'Wild Bagels',
    restaurant_id: '41225826'
  },
  {
    borough: 'Bronx',
    cuisine: 'Latin (Cuban, Dominican, Puerto Rican, South & Central American)',
    name: "Willie'S Steak House",
    restaurant_id: '41239497'
  }
]

```

15. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which contain 'ces' as last three letters for its name.

```

iacsd0324> db.restaurent.find({name:/ces$/},{restaurant_id:1,name:1,borough:1,cuisine:1,
_id:0})
[
  {
    borough: 'Manhattan',
    cuisine: 'American',
    name: 'Pieces',
    restaurant_id: '40399910'
  },
  {
    borough: 'Queens',
    cuisine: 'American',
    name: 'S.M.R Restaurant Services',
    restaurant_id: '40403857'
  },
  {
    borough: 'Manhattan',
    cuisine: 'American',
    name: 'Good Shepherd Services',
    restaurant_id: '40403989'
  },
  {
    borough: 'Queens',
    cuisine: 'Ice Cream, Gelato, Yogurt, Ices',
    name: 'The Ice Box-Ralph'S Famous Italian Ices',
    restaurant_id: '40600800'
  }
]

```

16. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which contain 'Reg' as three letters somewhere in its name.

```

iacsd0324> db.restaurent.find({name:/Reg/},{restaurant_id:1,name:1,borough:1,cuisine:1,
_id:0})
[
  {
    borough: 'Brooklyn',
    cuisine: 'American',
    name: 'Regina Caterers',
    restaurant_id: '40356649'
  },
  {
    borough: 'Manhattan',
    cuisine: 'Café/Coffee/Tea',
    name: 'Caffe Reggio',
    restaurant_id: '40369418'
  },
  {
    borough: 'Manhattan',
    cuisine: 'American',
    name: 'Regency Hotel',
    restaurant_id: '40382679'
  },
  {
    borough: 'Manhattan',
    cuisine: 'American',
    name: 'Regency Whist Club',
    restaurant_id: '40402377'
  }
]

```

17. Write a MongoDB query to find the restaurants which belong to the borough Bronx and

prepared either American or Chinese dish.

```
iacsd0324> db.restaurent.find({borough:'Bronx',cuisine:{$in:['American','Chinese']}})
[ncaught:
{taxError: Unexpected token, expected ",", (1:72)
  _id: ObjectId('660e81f1e9667c896323a281'),
  address: {urent.find({borough:'Bronx',cuisine:{$in:['American','Chinese']}})
    building: '2300',
    coord: [ -73.8786113, 40.8502883 ],
    street: 'Southern Boulevard',
    zipcode: '10460'
  },
  borough: 'Bronx',
  cuisine: 'American',
  grades: [
    {
      date: ISODate('2014-05-28T00:00:00.000Z'),
      grade: 'A',
      score: 11
    },
    {
      date: ISODate('2013-06-19T00:00:00.000Z'),
      grade: 'A',
      score: 4
    },
    {
      date: ISODate('2012-06-15T00:00:00.000Z'),
      grade: 'A',
      score: 3
    }
  ]
},
],
```

18. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which belong to the borough Staten Island or Queens or Bronx or Brooklyn.

```
iacsd0324> db.restaurent.find({borough:{$in:['Staten Island','Queens','Bronx','Brooklyn']},{restaurant_id:1,name:1,borough:1,cuisine:1,_id:0})
[
  {
    borough: 'Brooklyn',
    cuisine: 'American',
    name: 'Riviera Caterer',
    restaurant_id: '40356018'
  },
  {
    borough: 'Bronx',
    cuisine: 'Bakery',
    name: 'Morris Park Bake Shop',
    restaurant_id: '30075445'
  },
  {
    borough: 'Brooklyn',
    cuisine: 'Hamburgers',
    name: 'Wendy's',
    restaurant_id: '30112340'
  },
  {
    borough: 'Staten Island',
    cuisine: 'Jewish/Kosher',
    name: 'Kosher Island',
    restaurant_id: '40356442'
  }
]
```

19. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which are not belonging to the borough Staten Island or Queens or Bronx or Brooklyn.

Brooklyn.

```
iacsd0324> db.restaurent.find({'borough':{'$nin':['Staten Island','Queens','Bronx','Brooklyn']}},{'restaurant_id:1,name:1,borough:1,cuisine:1,_id:0'})
[
  {
    borough: 'Manhattan',
    cuisine: 'Irish',
    name: 'Dj Reynolds Pub And Restaurant',
    restaurant_id: '30191841'
  },
  {
    borough: 'Manhattan',
    cuisine: 'American',
    name: '1 East 66th Street Kitchen',
    restaurant_id: '40359480'
  },
  {
    borough: 'Manhattan',
    cuisine: 'American',
    name: 'Glorious Food',
    restaurant_id: '40361521'
  },
  {
    borough: 'Manhattan',
    cuisine: 'Delicatessen',
    name: 'Bully'S Deli',
    restaurant_id: '40361708'
  }
]
```

20. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which achieved a score which is not more than 10.

```
iacsd0324> db.restaurent.find({'grades.score':{'$lt:10}},{'restaurant_id:1,name:1,borough:1,cuisine:1,_id:0'})
[
  {
    borough: 'Manhattan',
    cuisine: 'Irish',
    name: 'Dj Reynolds Pub And Restaurant',
    restaurant_id: '30191841'
  },
  {
    borough: 'Brooklyn',
    cuisine: 'American',
    name: 'Riviera Caterer',
    restaurant_id: '40356018'
  },
  {
    borough: 'Bronx',
    cuisine: 'Bakery',
    name: 'Morris Park Bake Shop',
    restaurant_id: '30075445'
  },
  {
    borough: 'Brooklyn',
    cuisine: 'Hamburgers',
    name: 'Wendy'S',
    restaurant_id: '30112340'
  },
  {
    borough: 'Staten Island',
    cuisine: 'Jewish/Kosher',
    name: 'Kosher Island',
    restaurant_id: '40356442'
  },
  {
    borough: 'Queens',
    cuisine: 'American'
  }
]
```

21. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which prepared dish except 'American' and 'Chinees' or restaurant's name begins with letter 'Wil'.

```

iacs0324> db.restaurent.find({$or:[{cuisine:{$nin:['American','Chinese']}},{name:/^Wil/}],{restaurant_id:1,name:1,borough:1,cuisine:1,_id:0}})
{
  borough: 'Manhattan',
  cuisine: 'Irish',
  name: 'Dj Reynolds Pub And Restaurant',
  restaurant_id: '30191841'
},
{
  borough: 'Bronx',
  cuisine: 'Bakery',
  name: 'Morris Park Bake Shop',
  restaurant_id: '30075445'
},
{
  borough: 'Brooklyn',
  cuisine: 'Hamburgers',
  name: 'Wendy's',
  restaurant_id: '30112340'
},
{
  borough: 'Staten Island',
  cuisine: 'Jewish/Kosher',
  name: 'Kosher Island',
  restaurant_id: '40356442'
},
{
  borough: 'Queens',

```

22. Write a MongoDB query to find the restaurant Id, name, and grades for those restaurants which achieved a grade of "A" and scored 11 on an ISODate "2014-08-11T00:00:00Z" among many of survey dates

```

iacs0324> db.restaurent.find({grades:{$elemMatch:{grade:'A',score:11,date:ISODate("2014-08-11T00:00:00Z")}}},{restaurant_id:1,name:1,_id:0})
{ name: 'Don Filippo Restaurant', restaurant_id: '40372417' },
{ name: 'Cosi', restaurant_id: '40922983' },
{ name: 'Subway', restaurant_id: '41230741' },
{
  name: 'Smorgas Chef At Scandinavia House',
  restaurant_id: '41366873'
},
{ name: 'Schnitzel Express', restaurant_id: '41428543' },
{ name: 'Sweet Afton', restaurant_id: '41432714' },
{ name: 'Wen Ming Chinese Restaurant', restaurant_id: '41549803' },
{ name: 'Subway', restaurant_id: '41721364' },
{ name: 'China King', restaurant_id: '50005588' }
iacs0324>

```

23. Write a MongoDB query to find the restaurant Id, name and grades for those restaurants where the 2nd element of grades array contains a grade of "A" and score 9 on an ISODate "2014-08-11T00:00:00Z".

```

iacs0324> db.restaurent.find({'grades.1.grade':'A','grades.1.score':9,'grades.1.date':ISODate("2014-08-11T00:00:00Z")},{restaurant_id:1,grades:1,name:1,_id:0})
[
  {
    grades: [
      {
        date: ISODate('2015-01-12T00:00:00.000Z'),
        grade: 'A',
        score: 10
      },
      {
        date: ISODate('2014-08-11T00:00:00.000Z'),
        grade: 'A',
        score: 9
      }
    ]
  }
]

```

24. Write a MongoDB query to find the restaurant Id, name, address and geographical location for those restaurants where 2nd element of coord array contains a value which is more than 42 and upto 52

```
iacsd0324> db.restaurant.find({'address.coord.1':{'$gt:42,$lt:52}},{'restaurant_id:1,name:1,address:1,_id:0'})
[
  {
    address: {
      building: '47',
      coord: [ -78.877224, 42.89546199999999 ],
      street: 'Broadway @ Trinity Pl',
      zipcode: '10006'
    },
    name: "T.G.I. Friday'S",
    restaurant_id: '40387990'
  },
  {
    address: {
      building: '1',
      coord: [ -0.7119979, 51.6514664 ],
      street: 'Pennplaza E, Penn Sta',
      zipcode: '10001'
    },
    date: ISODate('2013-04-23T00:00:00.000Z'),
    name: 'T.G.I. Fridays',
    restaurant_id: '40388936'
  },
  {
    address: {ISODate('2012-04-24T00:00:00.000Z'),
```

25. Write a MongoDB query to arrange the name of the restaurants in ascending order along with all the columns.

26. Write a MongoDB query to arrange the name of the restaurants in descending order along with all the columns.

```

iacsd0324> db.restaurent.find().sort({name:-1})
[
  {
    _id: ObjectId('660e81f2e9667c896323e6e3'),
    address: {
      building: '1',
      coord: [ -74.073156, 40.6457369 ],
      street: 'Richmond Terrace',
      zipcode: '10301'
    },
    borough: 'Staten Island',
    cuisine: 'Pizza',
    grades: [
      {
        date: ISODate('2015-01-13T00:00:00.000Z'),
        grade: 'Z',
        score: 18
      },
      {
        date: ISODate('2014-07-24T00:00:00.000Z'),
        grade: 'A',
        score: 12
      },
      {
        date: ISODate('2013-11-08T00:00:00.000Z'),
        grade: 'B',
        score: 21
      },
      {
        date: ISODate('2013-04-17T00:00:00.000Z'),
        grade: 'A',
        score: 12
      }
    ],
    name: "Zz'S Pizza & Grill",
    restaurant_id: '41702705'
  }
]

```

27. Write a MongoDB query to arranged the name of the cuisine in ascending order and for that same cuisine borough should be in descending order.

```

iacsd0324> db.restaurent.find().sort({cuisine:1,borough:-1})
[
  {
    _id: ObjectId('660e81f2e9667c896323d2cb'),
    address: {
      building: '259-11',
      coord: [ -73.708831, 40.73748399999999 ],
      street: 'Hillside Avenue',
      zipcode: '11004'
    },
    borough: 'Queens',
    cuisine: 'Afghan',
    grades: [
      {
        date: ISODate('2014-09-15T00:00:00.000Z'),
        grade: 'A',
        score: 13
      },
      {
        date: ISODate('2013-09-18T00:00:00.000Z'),
        grade: 'A',
        score: 7
      },
      {
        date: ISODate('2013-04-18T00:00:00.000Z'),
        grade: 'B',
        score: 21
      },
      {

```

28. Write a MongoDB query to know whether all the addresses contains the street or not.


```

acsd0324> db.restaurent.find({'address.street':{'$exists:true'}})

{
  _id: ObjectId('660e81f1e9667c896323a271'),
  address: {
    building: '351',
    coord: [ -73.98513559999999, 40.7676919 ],
    street: 'West 57 Street',
    zipcode: '10019'
  },
  borough: 'Manhattan',
  cuisine: 'Irish',
  grades: [
    {
      date: ISODate('2014-09-06T00:00:00.000Z'),
      grade: 'A',
      score: 2
    },
    {
      date: ISODate('2013-07-22T00:00:00.000Z'),
      grade: 'A',
      score: 11
    }
  ]
}

```

30. Write a MongoDB query which will select the restaurant Id, name and grades for those restaurants which returns 0 as a remainder after dividing the score by 7.

```

iacsd0324> db.restaurent.find({'grades.score':{'$mod':[7,0]}},{restaurant_id:1,name:1,grades:1,_id:0})
[
  {
    score: 12
  },
  {
    grades: [
      {
        date: ISODate('2013-04-04T00:00:00.000Z'),
        date: ISODate('2014-06-10T00:00:00.000Z'),
        grade: 'A',
        score: 5
      },
      {
        date: ISODate('2012-01-24T00:00:00.000Z'),
        date: ISODate('2013-06-05T00:00:00.000Z'),
        grade: 'A',
        score: 7
      },
      {
        e: 'Kosher Island',
        date: ISODate('2012-04-13T00:00:00.000Z'),
        grade: 'A',
        score: 12
      },
      ObjectId('660e81f1e9667c896323a276'),
      {
        res: {
          date: ISODate('2011-10-12T00:00:00.000Z'),
          grade: 'A',
          score: 12
        },
        ipcode: '11369'
      }
    ]
  }
]

```

31. Write a MongoDB query to find the restaurant name, borough, longitude and attitude and cuisine for those restaurants which contains 'mon' as three letters somewhere in its name.

```

iacsd0324> db.restaurent.find({name:/mon/},{restaurant_id:1,name:1,borough:1,'address.coord':1,cuisine:1,_id:0})
[
  {
    address: { coord: [ -73.98306099999999, 40.7441419 ] },
    borough: 'Manhattan',
    cuisine: 'American',
    name: "Desmond'S Tavern",
    restaurant_id: '40366396'
  },
  {
    address: { coord: [ -73.8221418, 40.7272376 ] },
    borough: 'Queens',
    cuisine: 'Jewish/Kosher',
    name: 'Shimons Kosher Pizza',
    restaurant_id: '40366586'
  },
  {
    address: { coord: [ -74.10465599999999, 40.58834 ] },
    borough: 'Staten Island',
    cuisine: 'American',
    name: 'Richmond County Country Club',
    restaurant_id: '40366928'
  },
  {
    address: { coord: [ -73.9812843, 40.5947365 ] },
    borough: 'Brooklyn',
    cuisine: 'Pizza/Italian',
    name: 'Lb Spumoni Gardens',
    restaurant_id: '40367860'
  },
]

```

32. Write a MongoDB query to find the restaurant name, borough, longitude and latitude and cuisine for those restaurants which contain 'Mad' as first three letters of its name.

```

iacsd0324> db.restaurent.find({name:/^Mad/},{restaurant_id:1,name:1,borough:1,'address.coord':1,cuisine:1,_id:0})
[
  {
    address: { coord: [ -73.9860597, 40.7431194 ] },
    borough: 'Manhattan',
    cuisine: 'American',
    name: 'Madison Square',
    restaurant_id: '40402527'
  },
  {
    address: { coord: [ -73.98302199999999, 40.742313 ] },
    borough: 'Manhattan',
    cuisine: 'Indian',
    name: 'Madras Mahal',
    restaurant_id: '40519978'
  },
  {
    address: { coord: [ -74.000002, 40.72735 ] },
    borough: 'Manhattan',
    cuisine: 'American',
    name: 'Madame X',
    restaurant_id: '40611024'
  },
  {
    address: { coord: [ -73.98171959999999, 40.7499406 ] },
    borough: 'Manhattan',
    cuisine: 'French',
    name: 'Madison Bistro',
    restaurant_id: '40657588'
  },
  {
    address: { coord: [ -73.9717845, 40.6897199 ] },
    borough: 'Brooklyn',
    cuisine: 'African',
    name: 'Madiba',
    restaurant_id: '40684161'
  },
]

```

- ## Categories

2. List all the books with category Internet at first position in category array

3. Change the status of books “undergoing change” for books having more than 500

```
i@csd0324> db.book.updateMany({pageCount:{$gt:500}, publishedDate:{$gte:ISODate('2009-01-01'), $lt:ISODate('2009-12-31')}}),
i@csd0324>
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 8,
  modifiedCount: 8,
  upsertedCount: 0
}
```

```

iacsd0324> db.book.find({publishedDate:{$gte:ISODate('2009-01-01'),$lt:ISODate('2009-12-01')}})
[
  {
    _id: 4,
    title: 'Flex 3 in Action',
    isbn: '1933988746',
    pageCount: 576,
    publishedDate: ISODate('2009-02-02T08:00:00.000Z'),
    thumbnailUrl: 'https://s3.amazonaws.com/AKIAJC5RLADLUMVRPFDQ.book-thumb-images/ahmed.jpg',
    longDescription: "New web applications require engaging user-friendly interfaces and the cool,
    eat high-quality, effective, and interactive Rich Internet Applications (RIAs) quickly and easily.
    sophisticated tools and a straightforward programming language so you can focus on what you want.
    ex are free and open-source, the cost barrier is gone, as well! Flex 3 in Action is an easy-to-
    beyond feature coverage and helps you put Flex to work in real day-to-day tasks. You'll quickly ma
    ex applications stand out from the crowd. Interesting themes, styles, and skins It's in there.
    n You bet. Charting techniques to help you visualize data Bam! The expert authors of Flex 3 in
    Fast. Many Flex books are overwhelming to new users focusing on the complexities of the langua
    in Action filters out the noise and dives into the core topics you need every day. Using numerous
    ation that you can build on as the complexity of your projects increases.",
    status: 'undergoing change',
    authors: [ 'Tariq Ahmed with Jon Hirschi', 'Faisal Abid' ],
    categories: [ 'Internet' ],
    Genre: [ 'fiction', 'moral stories', 'adventurous' ]
  },
  {
    _id: 1,
    title: 'Unlocking Android',
    isbn: '1933988673',
    pageCount: 416,
    publishedDate: ISODate('2009-04-01T07:00:00.000Z'),
    thumbnailUrl: 'https://s3.amazonaws.com/AKIAJC5RLADLUMVRPFDQ.book-thumb-images/ablesen.jpg',
    shortDescription: "Unlocking Android: A Developer's Guide provides concise, hands-on instructi
    teaches important architectural concepts in a straightforward writing style and builds on this with
    longDescription: "Android is an open source mobile phone platform based on the Linux operating
    er 30 hardware, software and telecom companies that focus on open standards for mobile devices. Le
    more open and cost effective mobile experience. Unlocking Android: A Developer's Guide provide
    development tools. This book teaches important architectural concepts in a straightforward writing
    . Based on his mobile development experience and his deep knowledge of the arcane Android technica
    practical applications that build upon or replace any of Androids features, however small. Unloc
    platform in easy-to-understand language and builds on this foundation with re-usable Java code exam
  }
]

```

6. Find all books with pageCount is either 500 or 556 or 670

```
type it for more
iacsd0324> db.book.find({'pageCount':{'$in':[500,556,670]}})
[
  {
    _id: 54,
    title: 'Android in Practice',
    isbn: '1935182927',
    pageCount: 500,
    publishedDate: ISODate('2011-09-30T07:00:00.000Z'),
    thumbnailUrl: 'https://s3.amazonaws.com/AKIAJC5RLADLUMVRPFDQ.book-thumb-images/collins.jpg',
    shortDescription: 'Android in Practice is treasure trove of Android goodness, with over 100 tested, ready-to-use techniques including complete end-to-end example applications and practical tips for real world mobile application developers. Written by real world Android developers, this book addresses the trickiest questions raised in forums and mailing lists. Using an easy-to-follow problem/solution/discussion format, it dives into important topics not covered in other Android books, like advanced drawing and graphics, testing and instrumentation, building and deploying applications, using alternative languages, and native development.',
    longDescription: `Android, Google's platform for mobile application development, provides powerful features, a robust SDK, and almost limitless possibilities. It's not hard to find the information you need to build your first Android app, but then what? If you want to build real apps for real users, you have real questions and you need real answers. Android in Practice is treasure trove of Android goodness, with over 100 tested, ready-to-use techniques including complete end-to-end example applications and practical tips for real world mobile application developers. Written by real world Android developers, this book addresses the trickiest questions raised in forums and mailing lists. Using an easy-to-follow problem/solution/discussion format, it dives into important topics not covered in other Android books, like advanced drawing and graphics, testing and instrumentation, building and deploying applications, using alternative languages, and native development. If you're new to Android, or even if you have a few cycles under your belt, you'll love the quick "pre-flight check," where you'll review key platform details and tools and the building blocks of all Android applications. Then, you'll delve into Android use cases from visual elements and style, to saving and sharing data, networking, background tasks, concurrency, and more.`,
    status: 'PUBLISH',
```

7. Add 2 categories "kindle" and "hard bind" in all the books if its pageCount >200 and < 500 or number of pages >500

```

iacsd0324> db.book.updateMany({pageCount:{$gt:200,$lt:500}},{$push:{categories:{$each:["Kindle","Hard bind"]}}}
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 165,
  modifiedCount: 165,
  upsertedCount: 0
}
iacsd0324> db.book.find()
[
  {
    _id: 3,
    title: 'Specification by Example',
    isbn: '1617290084',
    pageCount: 0,
    publishedDate: ISODate('2011-06-03T07:00:00.000Z'),
    thumbnailUrl: 'https://s3.amazonaws.com/AKIAJCSRLADLUMVRPFDQ.book-thumb-images/adzic.jpg',
    status: 'PUBLISHED',
    authors: [ 'Gojko Adzic' ],
    categories: [ 'Software Engineering' ]
  },
  {
    _id: 4,
    title: 'Flex 3 in Action',
    isbn: '1933988746',
    pageCount: 576,
    publishedDate: ISODate('2009-02-02T08:00:00.000Z'),
    thumbnailUrl: 'https://s3.amazonaws.com/AKIAJCSRLADLUMVRPFDQ.book-thumb-images/ahmed.jpg',
    longDescription: "New web applications require engaging user-friendly interfaces and the cooler, the better. Flex 3 in Action shows you how to create high-quality, effective, and interactive Rich Internet Applications (RIAs) quickly and easily. Flex removes the need for complex and sophisticated tools and a straightforward programming language so you can focus on what you want to do instead of how to do it. Flex 3 in Action is an easy-to-follow, hands-on guide that goes beyond feature coverage and helps you put Flex to work in real day-to-day tasks. You'll quickly master the F
  }
]

```

8. List all the books which has thumbnailUrl key

```

iacsd0324> db.book.find({thumbnailUrl:{$exists:true}},{title:1,thumbnailUrl:1})
[
  {
    _id: 3,
    title: 'Specification by Example',
    thumbnailUrl: 'https://s3.amazonaws.com/AKIAJCSRLADLUMVRPFDQ.book-thumb-images/adzic.jpg'
  },
  {
    _id: 4,
    title: 'Flex 3 in Action',
    thumbnailUrl: 'https://s3.amazonaws.com/AKIAJCSRLADLUMVRPFDQ.book-thumb-images/ahmed.jpg'
  },
  {
    _id: 2,
    title: 'Android in Action, Second Edition',
    thumbnailUrl: 'https://s3.amazonaws.com/AKIAJCSRLADLUMVRPFDQ.book-thumb-images/ablesen2.jpg'
  },
  {
    _id: 1,
    title: 'Unlocking Android',
    thumbnailUrl: 'https://s3.amazonaws.com/AKIAJCSRLADLUMVRPFDQ.book-thumb-images/ablesen.jpg'
  },
  {
    _id: 6,
    title: 'Collective Intelligence in Action',
    thumbnailUrl: 'https://s3.amazonaws.com/AKIAJCSRLADLUMVRPFDQ.book-thumb-images/alag.jpg'
  },
  {
    _id: 5,
    title: 'Flex 4 in Action',
    thumbnailUrl: 'https://s3.amazonaws.com/AKIAJCSRLADLUMVRPFDQ.book-thumb-images/ahmed.jpg'
  }
]

```

9. Add key type with values ["fiction","moral stories","adventurous"] in all books

whose title starts with FI and contains a some where in the name

```

}
iacsd0324> db.book.updateMany({title:/^Fl.*a/i},{ $push:{Genre:{ $each:["fiction","moral stories","adventurous"]}}} )
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 5,
  modifiedCount: 5,
  upsertedCount: 0
}
iacsd0324> db.book.find()
[
  {
    _id: 3,
    title: 'Specification by Example',
    isbn: '1617290084',
    pageCount: 0,
    publishedDate: ISODate('2011-06-03T07:00:00.000Z'),
    thumbnailUrl: 'https://s3.amazonaws.com/AKIAJCSRLADLUMVRPFDQ.book-thumb-images/adzic.jpg',
    status: 'PUBLISH',
    authors: [ 'Gojko Adzic' ],
    categories: [ 'Software Engineering' ]
  },
  {
    _id: 4,
    title: 'Flex 3 in Action',
    isbn: '1933988746',
    pageCount: 576,
    publishedDate: ISODate('2009-02-02T08:00:00.000Z'),
    thumbnailUrl: 'https://s3.amazonaws.com/AKIAJCSRLADLUMVRPFDQ.book-thumb-images/ahmed.jpg',
    longDescription: "New web applications require engaging user-friendly interfaces and the cooler, the better. Use Flex 3 to create high-quality, effective, and interactive Rich Internet Applications (RIAs) quickly and easily. Flex removes the barrier to entry by providing sophisticated tools and a straightforward programming language so you can focus on what you want to do instead of how to do it. Flex 3 books are free and open-source, the cost barrier is gone, as well! Flex 3 in Action is an easy-to-follow, hands-on guide that goes beyond feature coverage and helps you put Flex to work in real day-to-day tasks. You'll quickly master the Flex API and how Flex applications stand out from the crowd. Interesting themes, styles, and skins It's in there. Working with data and charts You bet. Charting techniques to help you visualize data Bam! The expert authors of Flex 3 in Action have one thing in common: Fast. Many Flex books are overwhelming to new users focusing on the complexities of the language and the superfluous details. Flex 3 in Action filters out the noise and dives into the core topics you need every day. Using numerous easy-to-understand examples, Flex 3 in Action shows you how to build on as the complexity of your projects increases.",
    status: 'PUBLISH',
    authors: [ 'Tariq Ahmed with Jon Hirschi', 'Faisal Abid' ],
    categories: [ 'Internet' ],
    Genre: [ 'fiction', 'moral stories', 'adventurous' ]
  },
]

```

11. Add new author “myauthor” at position 2 for all books whose title starts with h or m and contains s at 2nd last position

```

iacsd0324> db.book.updateMany({title:/^[hm].*s.$/i},{ $push:{'authors.1':{$each:["Myauthor"]}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
iacsd0324> db.book.find({title:/^[hm].*s.$/i})
[
  {
    _id: 65,
    title: 'Hello! HTML5 & CSS3',
    isbn: '1935182897',
    pageCount: 325,
    publishedDate: ISODate('2012-10-17T07:00:00.000Z'),
    thumbnailUrl: 'https://s3.amazonaws.com/AKIAJCSRLADLUMVRPFDQ.book-thumb-images/crowther.jpg',
    shortDescription: "Quick and Easy HTML5 and CSS3 is written for the web designer or developer
5 features. After a quick review of the basics, you'll turn to what's new. Start by learning to ap
HTML5 pages. You'll then take a quick tour through the new APIs: Form Validation, Canvas, Drag & D
include video and audio on your pages without plug-ins, and how to draw interactive vector graphi
longDescription: "HTML and CSS are the foundation of the web, and HTML5 and CSS3 are the latest
velopment at all, you'll have to learn HTML5 and CSS3, so why not start now Quick and Easy HTML5
ng web sites with these really cool new tools. Quick and Easy HTML5 and CSS3 is written for the
tion to the new HTML and CSS features. After a quick review of the basics, you'll turn to what's n
y building your first real HTML5 pages. You'll then take a quick tour through the new APIs: Form V
You'll also discover how to include video and audio on your pages without plug-ins, and how to dra
ndamentals of HTML5, it's time to add some style to your pages with CSS3. New CSS features include
, you'll learn to layout your pages with the new flexible box and layout modules, and add the fini
c devices with media queries, and do all of it with less code thanks to the new selectors and pseu
ere you see all the features of HTML5 and CSS3 working together to produce responsive and lightwei
apps.",
    status: 'PUBLISH',
    authors: [ 'Rob Crowther', [ 'Myauthor' ] ],
    categories: [ 'Internet', 'Kindle', 'Hard bind' ]
  }
]

```

12. Increase pageCount by 100 for all books whose author at 1 st position is “Gal

Shachor”


```

iacsd0324> db.book.updateMany({'authors.0':'Gal Shachor'},{$inc:{pageCount:100}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
iacsd0324> db.book.find({'authors.0':'Gal Shachor'})
[
  {
    _id: 287,
    title: 'JSP Tag Libraries',
    isbn: '193011009X',
    pageCount: 756,
    publishedDate: ISODate('2001-05-01T07:00:00.000Z'),
    thumbnailUrl: 'https://s3.amazonaws.com/AKIAJC5RLADLUMVRPFDQ.book-thumb-images/shachor.jpg',
    longDescription: 'JSP Tag Libraries is a bible for serious JSP developers. The reader will learn how to use JSP tags that is beginning to have an enormous impact on the way people are developing JSP. JSP tags allow you to separate presentation and business logic. The book is fully loaded with many real-world tags and examples. The book also includes JavaBeans. To make the tag usage even more real, the book also offers two full-scale case studies: an e-Commerce applications WAP applications that work with current cellular phones. This book is available online content. It focuses on reusable component-centric design via JavaBeans and custom tags.',
    status: 'PUBLISH',
    authors: [ 'Gal Shachor', 'Adam Chace', 'Magnus Rydin' ],
    categories: [ 'Java', 'Internet' ]
  }
]

```

13. Overwrite "Magnus Rydin" with name "Fr"

```

iacsd0324> db.book.updateMany({'authors':'Magnus Rydin'},{$set:{"authors.$":"Fr"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}

```

14. List all books title, status, pageCount, comments for all books with pages > 300 or < 500 or title starts with a or isbn starts with 193

```
iacsd0324> db.book.find({$or:[{pageCount:{$gt:300,$lt:500}},{title:/^a/i},{isbn:/^19/}]},
... {title:1,status:1,pageCount:1,comments:1,isbn:1,_id:0})
[
  {
    title: 'Flex 3 in Action',
    isbn: '1933988746',
    pageCount: 576,
    status: 'undergoing change'
  },
  {
    title: 'Android in Action, Second Edition',
    isbn: '1935182722',
    pageCount: 592,
    status: 'PUBLISH'
  },
  {
    title: 'Unlocking Android',
    isbn: '1933988673',
    pageCount: 416,
    status: 'PUBLISH'
  },
  {
    title: 'Collective Intelligence in Action',
    isbn: '1933988312',
    pageCount: 425,
    status: 'PUBLISH'
  },
  {
    title: 'Flex on Java',
    isbn: '1933988797',
    pageCount: 265,
    status: 'PUBLISH'
  },
  {
    title: 'Flex 4 in Action',
    isbn: '1935182420',
    pageCount: 600,
    status: 'PUBLISH'
  },
  {
    title: 'Zend Framework in Action',
```