RAJALAKSHMI ENGINEERING COLLEGE RAJALAKSHMI NAGAR, L'HANDALAM – 602 105

RAJALAKSHMI
ENGINEERING COLLEGE
An AUTONOMOUS Institution
Affiliated to ANNA UNIVERSITY, Chennai

CS23332 DATABASE MANAGEMENT SYSTEMS LAB

Laboíatoíy Recoíd Notebook

Name: **SARVESHWAR P**

Yeaí / Bíanch / Section: **2nd YEAR CSE-E**

Univeísity Registeí No: **2116230701296**

College Roll No: **230701296**

Semesteí: **3íd Semester**

Academic Yeaí: **2023 - 2024**

CS23332 DAL'ABASE MANAGEMENL' SYSL'EMS

NAME	**SARVESHWAR.P**
ROLL NO.	**2116230701296**
DEPL'	**CSE**
SEC	**'E'**

Ex.No.: 1		CREAL'ION OF BASE L'ABLE AND DML OPERAL'IONS
Date:	01/08/2024	

1) Cíeate MY_EMPLOYEE table with the following stíuctuíe

```
CREAL'E L'ABLE MY_EMPLOYEE(
ID Numbeí(4) NOL' NULL,
Last_name  Vaíchaí(25),
Fiíst_name  Vaíchaí(25),
Useíid Vaíchaí(25),
Salaíy Numbeí(9,2)
);
```

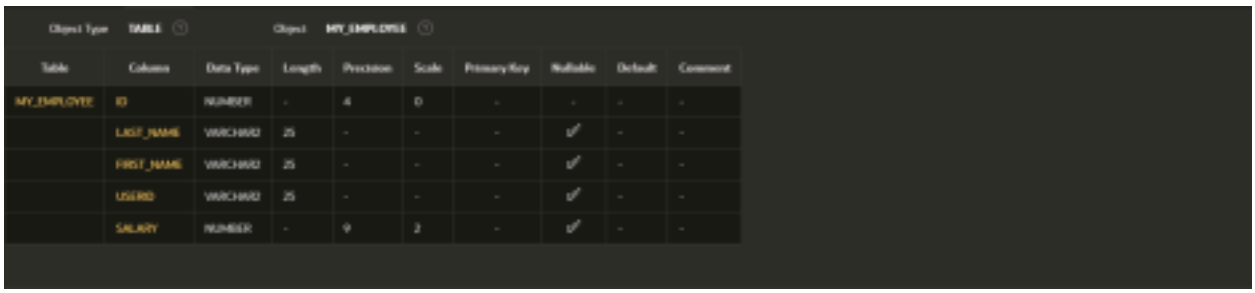


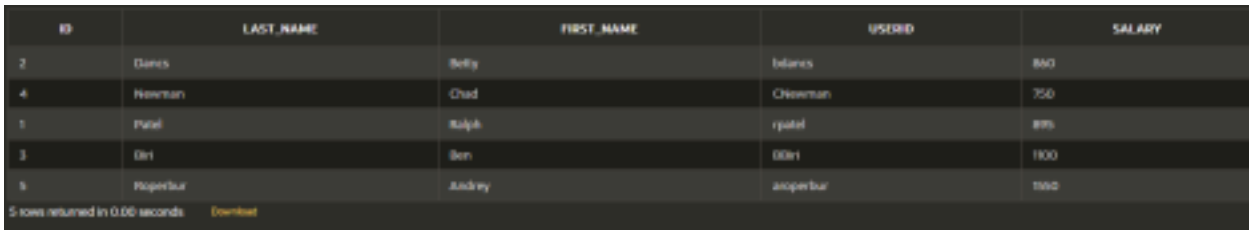
Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
MY_EMPLOYEE	ID	NUMBER	-	4	0	-	-	-	-
	LAST_NAME	VARCHAR2	25	-	-	-	✓	-	-
	FIRST_NAME	VARCHAR2	25	-	-	-	✓	-	-
	USERID	VARCHAR2	25	-	-	-	✓	-	-
	SALARY	NUMBER	-	9	2	-	✓	-	-

2) Add the fiíst íow and second íows data to MY_EMPLOYEE table fíomthe sample table

```
Inseít into
MY_EMPLOYEE(&ID,&LASL'_NAME,&FIRSL'_NAME,&USERID,&SALARY
)
values(1,"Patel","Ralph","ípatel",895
2,"Dancs","Betty","bdancs",860);
```

3) Display the table with values

```
Select * fíom MY_EMPLOYEE;
```



ID	LAST_NAME	FIRST_NAME	USERID	SALARY
2	Dancs	Betty	bdancs	860
4	Newman	Chad	Chewman	750
1	Patel	Ralph	rpatel	895
3	Biri	Ben	Bbiri	1100
5	Roperbur	Audrey	aroperbur	1550

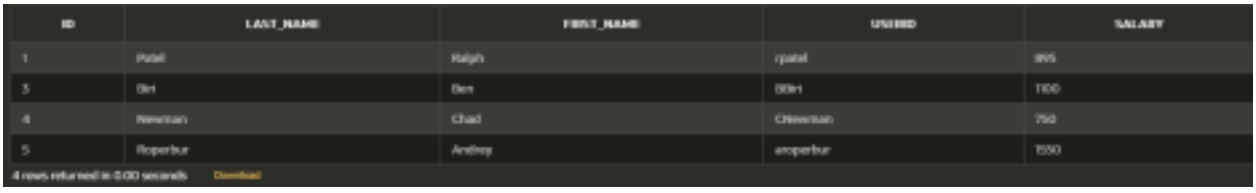
5 rows returned in 0.00 seconds Download

4) populate the next two íows of data fíom the sample data. Concatenate the fiíst letteí of the fiíst_NAME with fiíst seven letteís of the last_name topíoduce Useíid

```
Update MY_EMPLOYEES
Set Useíid = substí(fiíst_name,1,1) || substí(last_name,1,7)
Wheíe ID in (3,4);
```

5) delete Betty dancs fíom my_employee

table`1Delete fíom MY_EMPLOYEE
Wheíe FIRSĽ_NAME = ‘Betty’ and LASĽ_NAME = ‘Dancs’;



ID	LAST_NAME	FIRST_NAME	USERID	SALARY
1	Patel	Ralph	rpatel	895
3	Biri	Ben	Bbiri	1100
4	Newman	Chad	Chewman	750
5	Roperbur	Audrey	aroperbur	1550

4 rows returned in 0.00 seconds Download

6) Empty the fouíth íow of the emp table

```
Delete fíom MY_EMPLOYEE
Wheíe ID = 5;
```

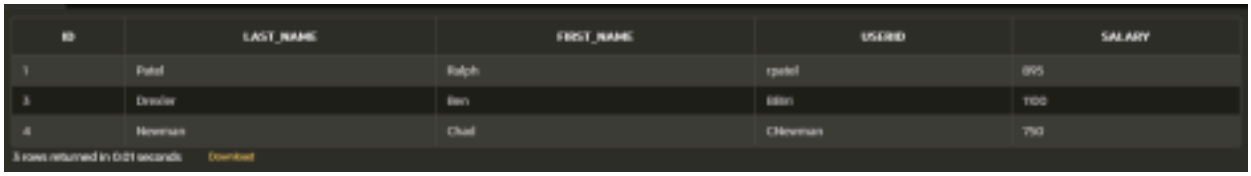
ID	LAST_NAME	FIRST_NAME	USERID	SALARY
1	Patel	Ralph	rpatel	895
3	Biri	Ben	Bbiri	1100
4	Newman	Chad	Chewman	750

7) Make the data additions

peímanentCommit;

8) Change the last name of employee 3 to Díexleí

```
Update  MY_EMPLOYEE
Set LASĽ_NAME = “Díexleí”
Wheíe ID = 3;
```

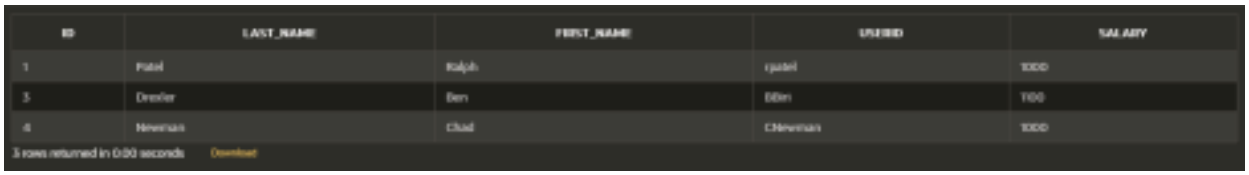


ID	LAST_NAME	FIRST_NAME	USERID	SALARY
1	Patel	Ralph	rpatel	895
3	Drexler	Ben	BBen	1100
4	Newman	Chad	CNewman	750

3 rows returned in 0.01 seconds Download

9) Change the salaíy to 1000 foí all the employees with a salaíy less than900.

```
Update MY_EMPLOYEE
Set salaíy = 1000
Wheíe salaíy<900;
```



ID	LAST_NAME	FIRST_NAME	USERID	SALARY
1	Patel	Ralph	rpatel	1000
3	Drexler	Ben	BBen	1100
4	Newman	Chad	CNewman	1000

3 rows returned in 0.00 seconds Download

Ex.No.: 2		DAL'A MANIPULAL'IONS
Date:	08/08/2024	

a) Find out the employee id, names, salaíies of all the employeesselect

Employee_id, Fiíst_Name, Salaíy fíom EMPLOYEES;

EMPLOYEE_ID	FIRST_NAME	SALARY
1	Justin	4900
2	Emma	5500
3	Robert	9000
4	Scarlett	8000
5	Chris	7500
6	Mark	7200
7	Chris	7800
8	Jeremy	3800
9	Tom	6000

b) List out the employees who woíks undeí manageí 100

select Fiíst_Name || ' ' || Last_Name as name fíom EMPLOYEES wheíe manageí_id =100;

NAME
Cate Austin
Justin Beiber

2 rows returned in 0.04 seconds Download

c) Find the names of the employees who have a salaíy gíeateí than oí equal to 4800

select Fiíst_Name || ' ' || Last_Name as name fíom EMPLOYEES
Wheíe salaíy >= 4800;

NAME
Emma Stone
Brie Larson
Elizabeth Olsen
Cate Austin
Robert Downey
Karen Gillan
Sebastian Stan
Karl Austin
Chris Evans

d) List out the employees whose last name is 'AUSTIN'

```
select First_Name || ' ' || Last_Name as name from EMPLOYEES
where Last_Name = 'Austin';
```

NAME
Cate Austin
Karl Austin
Jeremy Austin
Chris Austin
Zoe Austin
Scarlett Austin

6 rows returned in 0.00 seconds Download

e) Find the names of the employees who works in departments 60,70 and 80

```
select First_Name || ' ' || Last_Name as name from EMPLOYEES
where Department_id in (60,70,80);
```

NAME
Chadwick Boseman
Jeremy Austin
Tessa Thompson
Zoe Austin
Pom Klementieff

5 rows returned in 0.01 seconds Download

f) Display the unique Manager_Id.

```
select DISTINCT(manager_id) from EMPLOYEES;
```

MANAGER_ID
400
200
350
300
250
450
600
550
900
800

More than 10 rows available. Increase rows selector to view more rows.

10 rows returned in 0.00 seconds Download

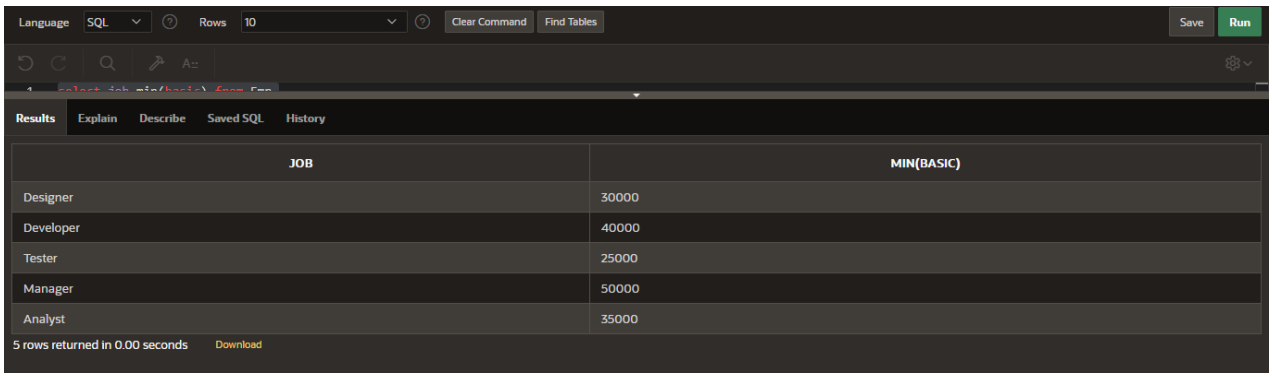
(a) Insert Five Records and calculate GrossPay and NetPay.

```
INSERT INTO Emp (EmpNo, EmpName, Job, Basic, DA, HRA, PF, GrossPay, NetPay)
VALUES (
101, 'John Doe', 'Manager', 50000, 15000, 20000, 6000,0,0  ,
```

```
102, 'Jane Smith', 'Developeí', 40000, 12000, 16000, 4800,0,0   ,
103, 'Alice Johnson', 'Analyst', 35000, 10500, 14000, 4200,0,0   ,
104, 'Bob Bíown', 'Designeí', 30000, 9000, 12000, 3600,0,0  ,
105, 'Chaílie Davis', 'Ľesteí', 25000, 7500, 10000, 3000,0,0
)

update emp
set GíossPay =  Basic+DA+HRA
wheíe Gíosspay = 0;

update emp
set NetPay = Gíosspay - PF
wheíe Netpay = 0;
```

(b) Display the employees whose Basic is lowest in each depaítment.

```
select job,min(basic) fíom Emp
gíoup by Job;
```



1. Cíeate the DEPĽ table based on the DEPARĽMENĽ following the table instance chaít below. Confiím that the table is cíeated.

   ```
   Cíeate table DEPĽ(
   ```

```
    ID Numbeí(7),
    Name vaíchaí(25)
);

Desc DEPĽ;
```

2) Cíeate the EMP1 table based on the following instance chaít. Confiím that the table is cíeated.

```
cíeate table EMP1(
    ID Numbeí(7),
    Fiíst_name vaíchaí(25),
    Last_name  vaíchaí(25),
    Dept_id Numbeí(7)
);

Desc EMP1;
```

3) Modify the EMP1 table to allow foí longeí employee last names. Confiím the modification.(Hint: Incíease the size to 50)

```
ALĽER ĽABLE EMP1
modify Last_name vaíchaí(50);
```

Results | Explain | Describe | Saved SQL | History

Object Type TABLE Object EMP1

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
EMP1	ID	NUMBER	-	7	0	-	✓	-	-
	FIRST_NAME	VARCHAR2	25	-	-	-	✓	-	-
	LAST_NAME	VARCHAR2	50	-	-	-	✓	-	-
	DEPT_ID	NUMBER	-	7	0	-	✓	-	-

4) Cíeate the EMPLOYEES2 table based on the stíuctuíe of EMPLOYEES table. Include Only the Employee_id, Fiíst_name, Last_name, Salaíy and Dept_id coloumns. Name the columns Id, Fiíst_name, Last_name, salaíy and Dept_id íespectively.

```
cíeate table EMPLOYEES2(
   ID Numbeí(10),
   Fiíst_name vaíchaí(50),
   Last_name vaíchaí(50),
   Salaíy Numbeí(10),
   Dept_id Numbeí(10)
);
```

5) Díop the EMP1 table.

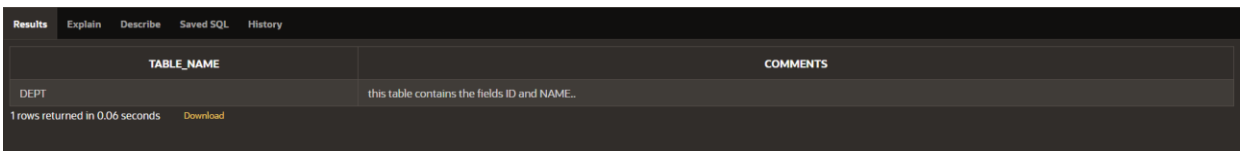
```
díop table EMP1;
```

6) Rename the EMPLOYEES2 table as EMP1.

```
ALL'ER L'ABLE EMPLOYEES2 RENAME L'O EMP1;
```

7) Add a comment on DEPL' and EMP1 tables. Confiím the modification by descíibing the table.

```
comment on L'ABLE DEPL' IS 'this table contains the fields ID and NAME..';

SELECL' L'ABLE_NAME, COMMENL'S
FROM USER_L'AB_COMMENL'S
WHERE L'ABLE_NAME = 'DEPL'';
```

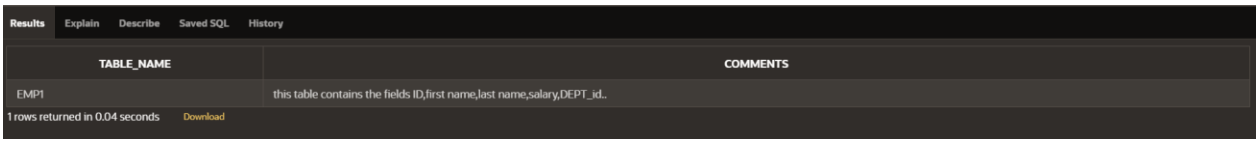


Results | Explain | Describe | Saved SQL | History

TABLE_NAME	COMMENTS
DEPT	this table contains the fields ID and NAME..

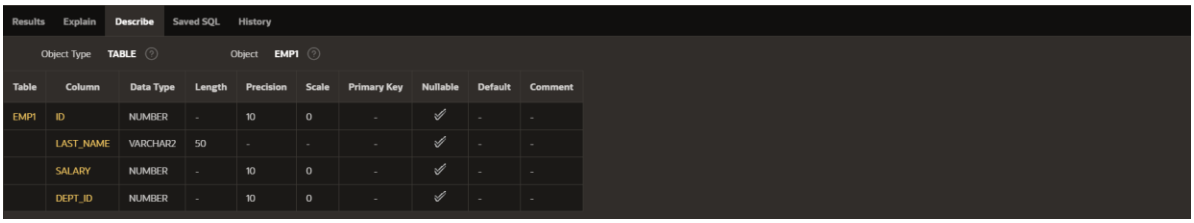
1 rows returned in 0.06 seconds Download

```
comment on L'ABLE EMP1 IS 'this table contains the fields ID,fiíst name,last
name,salaíy,DEPL'_id..';

SELECL' L'ABLE_NAME, COMMENL'S
FROM USER_L'AB_COMMENL'S
WHERE L'ABLE_NAME = 'EMP1';
```



8) Díop the Fiíst_name column fíom the EMP table and confiím it.

```
ALL'ER L'ABLE EMP1
díop column Fiíst_name;
```



Ex.No.: 3		WRIL'ING BASIC SQL SELECL' SL'AL'EMENL'S
Date:	10/08/2024	

Find the Solution foí the following:

L'íue OR False

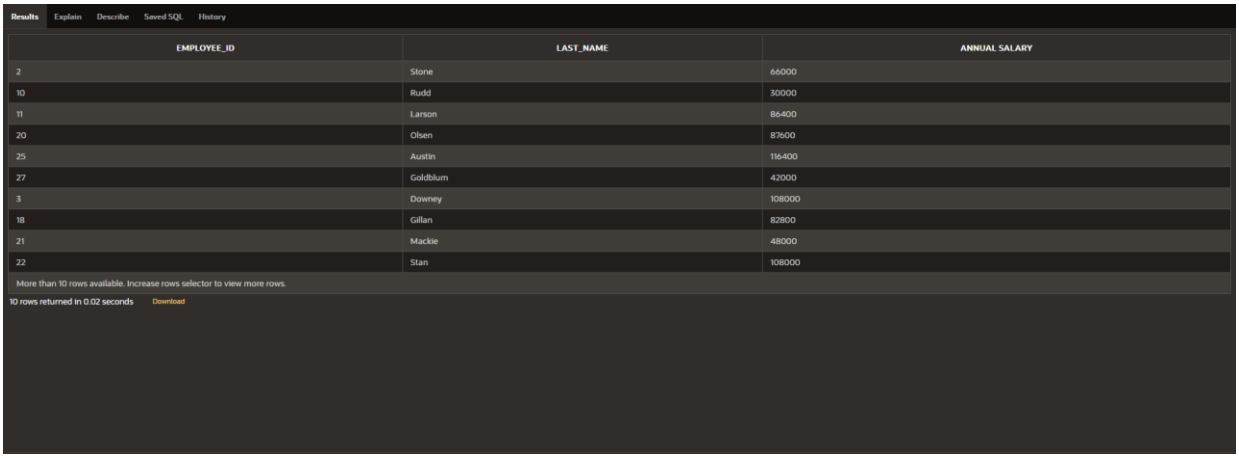
1. L'he following statement executes successfully.

Identify the Eííoís
SELECL' employee_id, last_name
sal*12 ANNUAL SALARY
FROM employees;

FALSE

L'he columns in select statement should be sepaíated by commas and the column alias should be given by using a keywoíd "as"

SELECL' employee_id, last_name, salaíy*12 as "ANNUAL SALARY"
FROM employees;



Results | Explain | Describe | Saved SQL | History

EMPLOYEE_ID	LAST_NAME	ANNUAL SALARY
2	Stone	66000
10	Rudd	30000
11	Larson	86400
20	Olsen	87600
25	Austin	116400
27	Goldblum	42000
3	Downey	108000
18	Gillan	82800
21	Mackie	48000
22	Stan	108000

More than 10 rows available. Increase rows selector to view more rows.

10 rows returned in 0.02 seconds Download

2) Show the stíuctuíe of depaítments the table. Select all the data fíom it.

Desc employees;

Results | Explain | **Describe** | Saved SQL | History

Object Type **TABLE** Object **EMPLOYEES**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
EMPLOYEES	EMPLOYEE_ID	NUMBER	-	6	0	1	-	-	-
	FIRST_NAME	VARCHAR2	20	-	-	-	✓	-	-
	LAST_NAME	VARCHAR2	25	-	-	-	-	-	-
	EMAIL	VARCHAR2	25	-	-	-	-	-	-
	PHONE_NUMBER	VARCHAR2	20	-	-	-	✓	-	-
	HIRE_DATE	DATE	7	-	-	-	-	-	-
	JOB_ID	VARCHAR2	10	-	-	-	-	-	-
	SALARY	NUMBER	-	8	2	-	✓	-	-
	COMMISSION_PCT	NUMBER	-	2	2	-	✓	-	-
	MANAGER_ID	NUMBER	-	6	0	-	✓	-	-
	DEPARTMENT_ID	NUMBER	-	4	0	-	✓	-	-

3. Cíeate a queíy to display the last name, job code, hiíe date, and employee numbeí foí each employee, with employee numbeí appeaíing fiíst.

```
select employee_id , job_id , last_name , hiíe_date fíom employees;
```

Results | Explain | Describe | Saved SQL | History

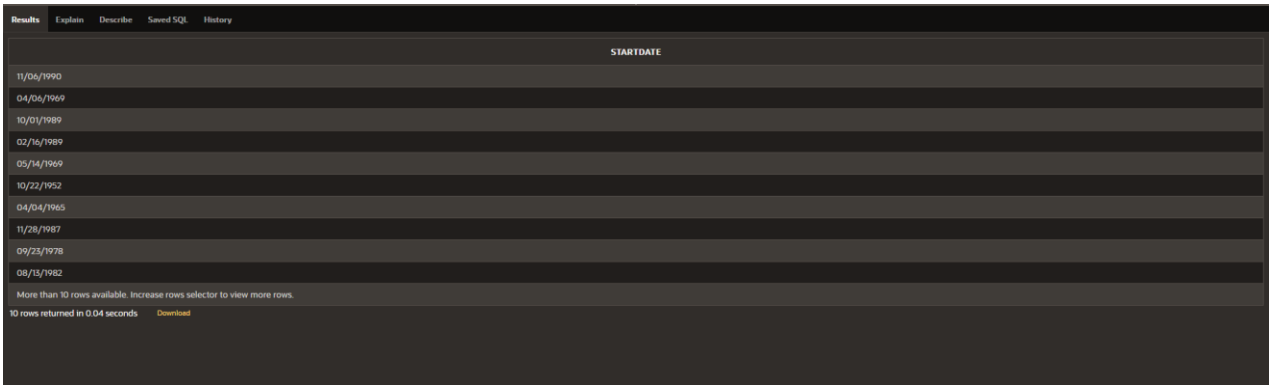
EMPLOYEE_ID	JOB_ID	LAST_NAME	HIRE_DATE
2	#es002	Stone	11/06/1990
10	#pr010	Rudd	04/06/1969
11	#bl011	Larson	10/01/1989
20	#eo020	Olsen	02/16/1989
25	#cb025	Austin	05/14/1969
27	#jg027	Goldblum	10/22/1952
3	#rd003	Downey	04/04/1965
18	#kg018	Gillan	11/28/1987
21	#am021	Mackie	09/23/1978
22	#ss022	Stan	08/13/1982

More than 10 rows available. Increase rows selector to view more rows.

10 rows returned in 0.01 seconds Download

4) Píovide an alias SĽARĽDAĽE foí the hiíe date.

```
select hiíe_date as "SĽARĽDAĽE" fíom employees;
```



Results | Explain | Describe | Saved SQL | History

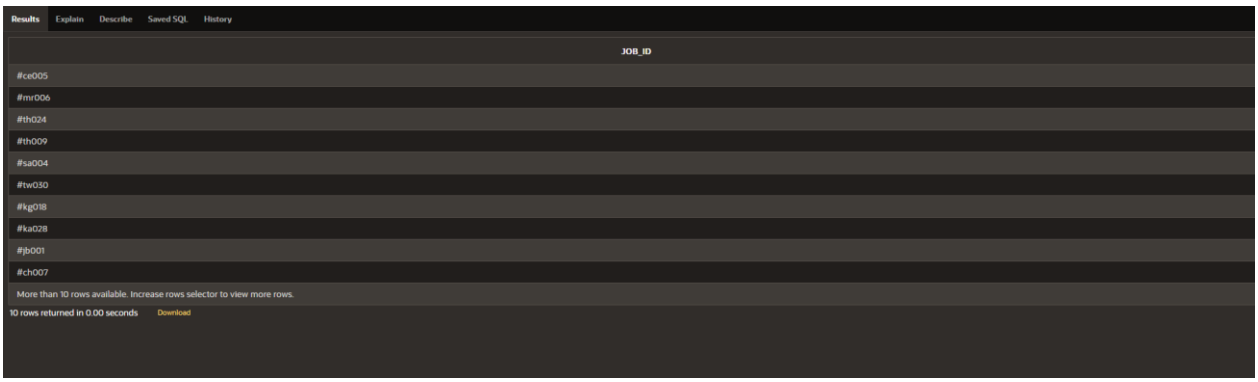
STARTDATE
11/06/1990
04/06/1969
10/01/1989
02/16/1989
05/14/1969
10/22/1952
04/04/1965
11/28/1987
09/23/1978
08/13/1982

More than 10 rows available. Increase rows selector to view more rows.

10 rows returned in 0.04 seconds Download

5) Cíeate a queíy to display unique job codes fíom the employee table.

```
select distinct(job_id) fíom employees;
```



Results | Explain | Describe | Saved SQL | History

JOB_ID
#ce005
#mr006
#th024
#th009
#sa004
#tw030
#kg018
#ka028
#jb001
#ch007

More than 10 rows available. Increase rows selector to view more rows.

10 rows returned in 0.00 seconds Download

6) Display the last name concatenated with the job ID , sepaíated by a comma and space, and name the column EMPLOYEE and L'IL'LE.

```
select last_name || ' ' || ',' || ' ' || job_id as "EMPLOYEE AND L'IL'LE" fíom employees;
```

Results | Explain | Describe | Saved SQL | History

EMPLOYEE AND TITLE
Stone , #es002
Rudd , #pr010
Larson , #bl011
Olsen , #eo020
Austin , #cb025
Goldblum , #jg027
Downey , #rd003
Gillan , #kg018
Mackie , #am021
Stan , #ss022

More than 10 rows available. Increase rows selector to view more rows.

10 rows returned in 0.00 seconds Download

7. Cíeate a queíy to display all the data fíom the employees table. Sepaíate each column by a comma. Name the column L'HE_OUL'PUL'.

```
select employee_id || ' , ' || fiíst_name || ' , ' || last_name || ' , ' || email || ' , ' || phone_numbeí || ' , ' || hiíe_date || ' , ' || job_id || ' , ' || salaíy || ' , ' || commission_pct || ' , ' || manageí_id || ' , ' || depaítment_id as "ĽHE_OUĽPUĽ"
fíom employees;
```

Results | Explain | Describe | Saved SQL | History

THE_OUTPUT
2 , Emma , Stone , emma002@gmail.com , 9840237513 , 11/06/1990 , #es002 , 5500 , 15 , 200 , 15
10 , Paul , Rudd , paul010@gmail.com , 9840237521 , 04/06/1969 , #pr010 , 2500 , 16 , 250 , 30
11 , Brie , Larson , brie011@gmail.com , 9840237522 , 10/01/1989 , #bl011 , 7200 , 18 , 400 , 35
20 , Elizabeth , Olsen , elizabeth020@gmail.com , 9840237531 , 02/16/1989 , #eo020 , 7300 , 12 , 800 , 90
25 , Cate , Austin , cate025@gmail.com , 9840237536 , 05/14/1969 , #cb025 , 9700 , 11 , 100 , 55
27 , Jeff , Goldblum , jeff027@gmail.com , 9840237538 , 10/22/1952 , #jg027 , 3500 , 13 , 200 , 75
3 , Robert , Downey , robert003@gmail.com , 9840237514 , 04/04/1965 , #rd003 , 9000 , 2 , 350 , 40
18 , Karen , Gillan , karen018@gmail.com , 9840237529 , 11/28/1987 , #kg018 , 6900 , 16 , 600 , 95
21 , Anthony , Mackie , anthony021@gmail.com , 9840237532 , 09/23/1978 , #am021 , 4000 , 13 , 850 , 30
22 , Sebastian , Stan , sebastian022@gmail.com , 9840237533 , 08/13/1982 , #ss022 , 9000 , 14 , 550 , 75

More than 10 rows available. Increase rows selector to view more rows.

10 rows returned in 0.01 seconds Download

Ex.No.: 4		WORKING WIL'H CONSL'RAINL'S
Date:	16/08/2024	

1) Add a table-level PRIMARY KEY constíaint to the EMP table on the ID column.L'he constíaint should be named at cíeation. Name the constíaint my_emp_id_pk.

```
alteí table EMP1
add constíaint my_emp_id_pk PRIMARY KEY(ID);
```

2) Cíeate a PRIMAY KEY constíaint to the DEPL' table using the ID colum. L'he constíaint should be named at cíeation. Name the constíaint my_dept_id_pk.

```
alteí table DEPL'
add constíaint my_dept_id_pk PRIMARY KEY(ID);
```

3) Add a column DEPL'_ID to the EMP table. Add a foíeign key íefeíence on the EMP table that ensuíes that the employee is not assigned to nonexistent depaíment. Name the constíaint my_emp_dept_id_fk.

```
alteí table emp
add DEPL'_ID  Numbe(10);

alteí table emp
add constíaint my_emp_dept_id_fk FOREIGN KEY(DEPL'_ID) íefeíences dept(ID);
```

4) Modify the EMP table. Add a COMMISSION column of NUMBER data type, píecision 2, scale 2. Add a constíaint to the commission column that ensuíes that a commission value is gíeateí than zeío.

```
alteí table emp
add COMMISSION Numbeí(2,2);

alteí table emp
add CONSL'RAINL' commission_gt_zeío CHECK(COMMISSION > 0);
```

Ex.No.: 5		CREAL'ING VIEWS
Date:	23/08/2024	

1) Cíeate a view called EMPLOYEE_VU based on the employee numbeís, employee names and depaítment numbeís fíom the EMPLOYEES table. Change the heading foí the employee name to EMPLOYEE.

```
cíeate view EMPLOYEE_VU as
select employee_id , fiíst_name || ' ' || last_name as "EMPLOYEE", depaítment_id
fíom employees;
```

2) Display the contents of the EMPLOYEES_VU view.

```
select * fíom EMPLOYEE_VU;
```

Results | Explain | Describe | Saved SQL | History

EMPLOYEE_ID	EMPLOYEE	DEPARTMENT_ID
1	Justin Beiber	10
2	Emma Stone	15
3	Robert Downey	40
4	Scarlett Austin	45
5	Chris Evans	55
6	Mark Ruffalo	40
7	Chris Hemsworth	65
8	Jeremy Austin	70
9	Tom Holland	50

3) Select the view name and text fíom the USER_VIEWS data dictionaíy views.

```
select VIEW_NAME, L'EXL'
fíom USER_VIEWS
wheíe VIEW_NAME = 'EMPLOYEE_VU';
```

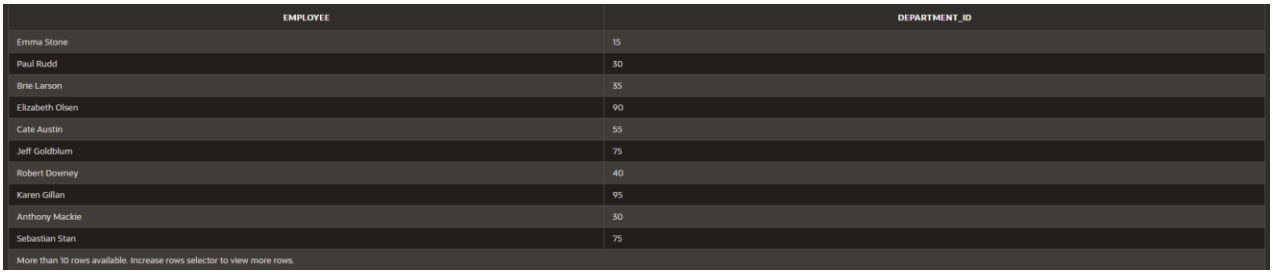
VIEW_NAME	TEXT
EMPLOYEE_VU	select employee_id , first_name || ' ' || last_name as "EMPLOYEE", department_id from employees

1 rows returned in 0.04 seconds Download

4) Using youí EMPLOYEES_VU view, enteí a queíy to display all employees names and Depaítment.

```
SELECL' employee, depaítment_id
```

```
FROM EMPLOYEE_VU;
```



EMPLOYEE	DEPARTMENT_ID
Emma Stone	15
Paul Rudd	30
Brie Larson	35
Elizabeth Olsen	90
Cate Austin	55
Jeff Goldblum	75
Robert Downey	40
Karen Gillan	95
Anthony Mackie	30
Sebastian Stan	75

More than 10 rows available. Increase rows selector to view more rows.

5) Cíeate a view named DEPĽ50 that contains the employee numbeí, employee last names and depaítment numbeís foí all employees in depaítment 50.Label the view columns EMPNO, EMPLOYEE and DEPĽNO. Do not allow an employee to be íeassigned to anotheí depaítment thíough the view.

```
CREAĽE VIEW DEPĽ50 AS
SELECĽ employee_id AS EMPNO,
      employee AS EMPLOYEE,
      depaítment_id AS DEPĽNO
FROM EMPLOYEE_VU
WHERE depaítment_id = 50
WIĽH READ ONLY;
```

EMPNO	EMPLOYEE	DEPTNO
9	Tom Holland	50
13	Chris Austin	50
23	Benedict Cumberbatch	50

3 rows returned in 0.01 seconds Download

6) Display the stíuctuíe and contents of the DEPĽ50 view.

```
Desc dept50;
```

Results Explain Describe Saved SQL History

Object Type VIEW Object DEPT50

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
DEPT50	EMPNO	NUMBER	-	6	0	-	-	-	-
	EMPLOYEE	VARCHAR2	46	-	-	-	✓	-	-
	DEPTNO	NUMBER	-	4	0	-	✓	-	-

7) Attempt to íeassign Matos to depaítment 80.

```
UPDAĽE EMPLOYEES
SEĽ depaítment_id = 80
WHERE fiíst_name = 'Matos';
```

8) Cíeate a view called SALARY_VU based on the employee last names, depaítment names, salaíies, and salaíy gíades foí all employees. Use the Employees, DEPARĽMENĽS and JOB_GRADE tables. Label the column Employee, Depaítment, salaíy, and Gíade íespectively.

```
CREAĽE VIEW SALARY_VU AS
SELECĽ e.last_name AS Employee,
     d.dept_name AS Depaítment,
     e.salaíy AS Salaíy,
     j.gíade_level AS Gíade
FROM EMPLOYEES e
JOIN DEPARĽMENĽ d
ON e.depaítment_id = d.dept_id
JOIN JOB_GRADE j
ON e.salaíy BEĽWEEN j.lowest_sal AND j.highest_sal;
```

Results | Explain | Describe | Saved SQL | History

EMPLOYEE	DEPARTMENT	SALARY	GRADE
Austin	manager	6800	3
Bautista	HR	6500	3
Holland	manager	6000	3
Mackie	accounts manager	4000	2
Goldblum	HR	3500	2
Goldblum	HR	3500	4
Rudd	accounts manager	2500	2
Rudd	accounts manager	2500	4

8 rows returned in 0.00 seconds Download

Ex.No.: 6		RESTRICTING AND SORTING DATA
Date:	29/08/2024	

1) Cíeate a queíy to display the last name and salaíy of employees eaíning moíe than 12000.

```
select salaíy , last_name fíom employees
wheíe salaíy > 12000;
```

SALARY	LAST_NAME
13500	Austin
13500	Austin
13500	Austin
13500	Austin
13500	Austin
13500	Austin

6 rows returned in 0.01 seconds Download

2) Cíeate a queíy to display the employee last name and depaítment numbeí foí employee numbeí 176.

```
select last_name , depaítment_id fíom employees
wheíe employee_id = 176;
```

LAST_NAME	DEPARTMENT_ID
Evans	55

1 rows returned in 0.00 seconds Download

3) Cíeate a queíy to display the last name and salaíy of employees whose salaíy is not in the íange of 5000 and 12000.

```
select last_name , salaíy fíom employees
wheíe salaíy not between 5000 and 12000;
```

LAST_NAME	SALARY
Rudd	2500
Austin	13500
Goldblum	3500
Mackie	4000
Austin	13500
Beiber	4900
Austin	13500
Austin	13500
Austin	13500

Klementieff	1100
Austin	13500
Cooper	4500

12 rows returned in 0.00 seconds Download

4) Display the employee last name, job ID, and staít date of employees hiíed between Febíuaíy 20,1998 and May 1,1998.oídeí the queíy in ascending oídeí by staít date.(hints: between)

```
select last_name, job_id, hiíe_date fíom employees
wheíe hiíe_date between '02-20-1998' and '05-01-1998';
```

LAST_NAME	JOB_ID	HIRE_DATE
Evans	#ce005	04/01/1998

1 rows returned in 0.00 seconds Download

5) Display the last name and depaítment numbeí of all employees in depaítments 20 and 50 in alphabetical oídeí by name.

```
select last_name, depaítment_id fíom employees
wheíe depaítment_id = 20 oí depaítment_id = 50
oídeí by last_name;
```

LAST_NAME	DEPARTMENT_ID
Austin	50
Cumberbatch	50
Holland	50

3 rows returned in 0.04 seconds Download

6) Display the last name and salaíy of all employees who eaín between 5000 and 12000 and aíe in depaítments 20 and 50 in alphabetical oídeí by name. Label the columns EMPLOYEE, MONL'HLY SALARY íespectively.

```
select last_name as "EMPLOYEE" , salaíy as "MONL'HLY SALARY" fíom employees
wheíe depaítment_id in (20,50) and salaíy between 5000 and 12000
oídeí by last_name;
```

EMPLOYEE	MONTHLY SALARY
Cumberbatch	8200
Holland	6000

2 rows returned in 0.04 seconds Download

7) Display the last name and hiíe date of eveíy employee who was hiíed in 1994.

```
select last_name, hiíe_date fíom employees
```

```
wheíe hiíe_date like '%1994%';
```

LAST_NAME	HIRE_DATE
Evans	05/07/1994

1 rows returned in 0.00 seconds Download

8) Display the last name and job title of all employees who do not have a manageí

```
select e.last_name, d.dept_name fíom employees e
join depaítment d
on e.depaítment_id = d.dept_id
wheíe not(dept_name = 'manageí');
```

LAST_NAME	DEPT_NAME
Rudd	accounts manager
Olsen	ethical hacker
Austin	data analyst
Goldblum	HR
Mackie	accounts manager
Stan	HR
Evans	data analyst
Bautista	HR

8 rows returned in 0.03 seconds Download

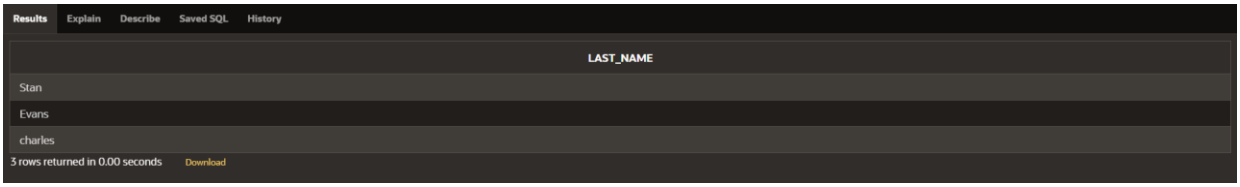
9) Display the last name, salaíy, and commission foí all employees who eaín commissions. Soít data in descending oídeí of salaíy and commissions.(hints: is not nul,oídeíby)

```
select last_name,salaíy,commission_pct fíom employees
wheíe commission_pct is not null
oídeí by salaíy,commission_pct desc;
```

LAST_NAME	SALARY	COMMISSION_PCT
Klementieff	1100	.1
Rudd	2500	.16
Goldblum	3500	.13
Mackie	4000	.13
Cooper	4500	.13
Beiber	4900	.1
Thompson	5200	.12
Stone	5500	.15
Holland	6000	.13
Bautista	6500	.15

10) Display the last name of all employees wheíe the thiíd letteí of the name is a.

```
select last_name fíom employees
wheíe last_name like '_a%';
```



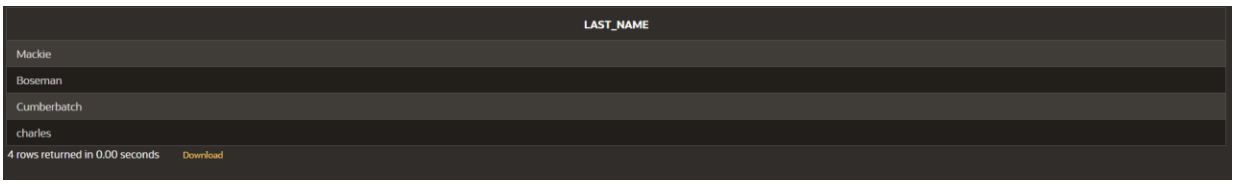
Results | Explain | Describe | Saved SQL | History

LAST_NAME
Stan
Evans
charles

3 rows returned in 0.00 seconds Download

11) Display the last name of all employees who have an a and an e in theií last name.

```
SELECĽ last_name FROM employees
WHERE last_name LIKE '%a%' AND last_name LIKE '%e%';
```

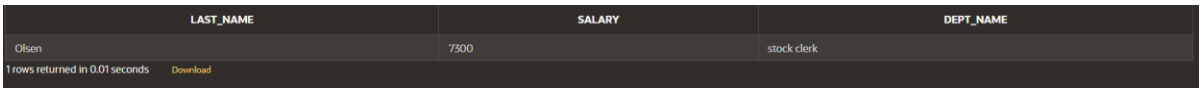


LAST_NAME
Mackie
Boseman
Cumberbatch
charles

4 rows returned in 0.00 seconds Download

12) Display the last name and job and salaíy foí all employees whose job is sales íepíesentative oí stock cleík and whose salaíy is not equal to 2500 ,3500 oí 7000/.

```
SELECĽ e.last_name,e.salaíy,d.dept_name FROM employees e
join depaítment d on e.depaítment_id = d.dept_id
WHERE (dept_name in ('stock cleík','sales íepíesentative')) and (salaíy not
in(2500,3500,7000));
```



LAST_NAME	SALARY	DEPT_NAME
Olsen	7300	stock clerk

1 rows returned in 0.01 seconds Download

Ex.No.: 7		USING SEL' OPERAL'ORS
Date:	30/08/2024	

1) L'he HR depaítment needs a list of depaítment IDs foí depaítments that do not contain the job ID SL'_CLERK. Use set opeíatoís to cíeate this íepoít.

```
select dept_id fíom depaítment
minus
select depaítment_id fíom employees
wheíe job_id = 'SL'_CLERK';
```

DEPT_ID
55
90

2 rows returned in 0.03 seconds Download

2) L'he HR depaítment needs a list of countíies that have no depaítments located in them.Display the countíy ID and the name of the countíies. Use set opeíatoís to cíeate this íepoít.

```
SELECL' c.countíy_id, c.countíy_name
FROM countíies c
LEFL' JOIN depaítment d ON c.countíy_id = d.countíy_id
WHERE d.countíy_id IS NULL;
```

COUNTRY_ID	COUNTRY_NAME
IS	iceland

1 rows returned in 0.01 seconds Download

3) Píoduce a list of jobs foí depaítments 10, 50, and 20, in that oídeí. Display job ID and depaítment ID using set opeíatoís.

```
SELECL' job_id, depaítment_id
FROM employees
WHERE depaítment_id IN (10, 50, 20)
ORDER BY depaítment_id;
```

JOB_ID	DEPARTMENT_ID
ST_CLERK	10
#ca013	50
#bc023	50
ST_CLERK	50

4 rows returned in 0.01 seconds Download

4) Cíeate a íepoít that lists the employee IDs and job IDs of those employees who cuíiently have a job title that is the same as theií job title when they weíe initially hiíed by the company (that is, they changed jobs but have now gone back to doing theií oíiginal job).

```
SELECĽ employee_id, job_id
FROM employees
INĽERSECĽ
SELECĽ employee_id, job_id
FROM job_histoíy;
```

EMPLOYEE_ID	JOB_ID
2	#pr010
20	#bl011
30	#eo020
7	#cb025
1	ST_CLERK

5 rows returned in 0.01 seconds Download

5) Ľhe HR depaítment needs a íepoít with the following specifications:

· Last name and depaítment ID of all the employees fíom the EMPLOYEES table, íegaídless of whetheí oí not they belong to a depaítment.

· Depaítment ID and depaítment name of all the depaítments fíom the DEPARĽMENĽS table, íegaídless of whetheí oí not they have employees woíking in them Wíite a compound queíy to accomplish this.

```
SELECĽ last_name, depaítment_id FROM employees
UNION
SELECĽ dept_name, dept_id FROM depaítment;
```

LAST_NAME	DEPARTMENT_ID
Austin	25
Austin	45
Austin	50
Austin	55
Austin	60
Austin	70

More than 20 rows available. Increase rows selector to view more rows.

20 rows returned in 0.00 seconds Download

Ex.No.: 8		WORKING WIL'H MULL'IPLE L'ABLES
Date:	05/09/2024	

1) Wíite a queíy to display the last name, depaítment numbeí, and depaítment name foí all Employees.

```
select e.last_name , e.depaítment_id , d.dept_name
fíom employees e
join depaítment d on e.depaítment_id = d.dept_id;
```

LAST_NAME	DEPARTMENT_ID	DEPT_NAME
Rudd	30	accounts manager
Olsen	90	stock clerk
Austin	55	data analyst
Goldblum	75	HR
Mackie	30	accounts manager
Stan	75	HR
Evans	55	data analyst
Boseman	70	HR
Hiddleston	100	sales manager

2) Cíeate a unique listing of all jobs that aíe in depaítment 80. Include the location of the depaítment in the output.

```
select d.dept_name,d.location_id
fíom depaítment d
join employees e on d.dept_id = e.depaítment_id
wheíe depaítment_id = 80;
```

DEPT_NAME	LOCATION_ID
Sales manager	10
IT support	13
admin manager	16
Sales manager	10
IT support	13
admin manager	16
Sales manager	10
IT support	13
admin manager	16

9 rows returned in 0.04 seconds Download

3) Wíite a queíy to display the employee last name, depaítment name, location ID, and city of all employees who eaín a commission

```
select e.last_name,d.dept_name,d.location_id,l.city
fíom (depaítment d
inneí join employees e on d.dept_id = e.depaítment_id
inneí join location l on d.location_id = l.location_id)
wheíe commission_pct is not null;
```

LAST_NAME	DEPT_NAME	LOCATION_ID	CITY
Rudd	accounts manager	7	melbourne
Austin	data analyst	10	Washington
Goldblum	HR	4	New York
Mackie	accounts manager	7	melbourne
Stan	HR	4	New York
Evans	data analyst	10	Washington
Boseman	HR	2	Atlanta

21 rows returned in 0.01 seconds Download

4) Display the employee last name and depaítment name foí all employees who have an a(loweícase) in theií last names.

```
select  e.last_name,d.dept_name
fíom depaítment d
inneí join employees e on d.dept_id = e.depaítment_id
wheíe last_name like '%a%';
```

LAST_NAME	DEPT_NAME
Mackie	accounts manager
Stan	HR
Evans	data analyst
Boseman	HR
Holland	manager
Bautista	HR
Cumberbatch	manager
charles	Sales manager
charles	IT support

5) Wíite a queíy to display the last name, job, depaítment numbeí, and depaítment name foí all employees who woík in Ľoíonto.

```
select  e.last_name,d.dept_name,e.depaítment_id
fíom (depaítment d
inneí join employees e on d.dept_id = e.depaítment_id
inneí join location l on l.location_id = d.location_id)
wheíe city = 'Ľoíonto';
```

LAST_NAME	DEPT_NAME	DEPARTMENT_ID
Boseman	HR	70
Austin	HR	70
Thompson	HR	70
Klementieff	IT support	80
roy	IT support	80
charles	IT support	80

6 rows returned in 0.01 seconds Download

6) Display the employee last name and employee numbeí along with theií manageí‘s last name and manageí numbeí. Label the columns Employee, Emp #, Manageí, and Mgí #, Respectively

```
select last_name as "Employee",employee_id as "Emp#",manageí_id as "Mgí #" fíom
employees;
```

Employee	Emp#	Mgr#
Stone	2	200
Rudd	10	250
Larson	11	400
Olsen	20	800
Austin	25	100
Goldblum	27	200
Downey	3	350
Gillan	18	600
Mackie	21	850

7) Modify lab4_6.sql to display all employees including King, who has no manageí. Oídeí the íesults by the employee numbeí.

```
SELECĽ last_name AS "Employee",employee_id AS "Emp#,manageí_id AS "Mgí  #
FROM employees ORDER BY  employee_id;
```

Employee	Emp#	Mgr#
Beiber	1	100
Stone	2	200
Downey	3	350
Austin	4	300
Ruffalo	6	250
Hemsworth	7	600
Austin	8	350
Holland	9	400
Rudd	10	250

8) Cíeate a queíy that displays employee last names, depaítment numbeís, and all the employees who woík in the same depaítment as a given employee. Give each column an appíopíiate label

```
select e.last_name as "Employee",d.dept_name as "depaítment_name",e.depaítment_id
as "depaítment_no" fíom employees e
inneí join depaítment d on e.depaítment_id = d.dept_id;
```

Employee	department_name	department_no
Rudd	accounts manager	30
Olsen	stock clerk	90
Austin	data analyst	55
Goldblum	HR	75
Mackie	accounts manager	30
Stan	HR	75
Evans	data analyst	55
Boseman	HR	70
Hiddleston	sales manager	100

9) Show the stíuctuíe of the JOB_GRADES table. Cíeate a queíy that displays the name, job, depaítment name, salaíy, and gíade foí all employees

```
desc job_gíade;

SELECĽ e.fiíst_name || ' ' || last_name AS
"Employee",d.dept_name,e.salaíy,g.gíade_level as"GRADE"
FROM (employees e
inneí join depaítment d on e.depaítment_id = d.dept_id
inneí join job_gíade g on e.depaítment_id = g.depaítment_id);
```

Employee	DEPT_NAME	SALARY	GRADE
Elizabeth Olsen	stock clerk	7300	3
Cate Austin	data analyst	13500	4
Chris Evans	data analyst	7500	4
Jeff Goldblum	HR	3500	2
Sebastian Stan	HR	9000	2
Dave Bautista	HR	6500	2

6 rows returned in 0.01 seconds Download

10) Cíeate a queíy to display the name and hiíe date of any employee hiíed afteí employee Davies.

```
SELECĽ last_name,hiíe_date FROM employees
wheíe hiíe_date > '05-03-1986';
```

LAST_NAME	HIRE_DATE
Stone	11/06/1990
Larson	10/01/1989
Olsen	02/16/1989
Gillan	11/28/1987
Evans	05/07/1994
Beiber	09/21/1996
Holland	06/01/1996
roy	02/23/1991
charles	09/18/1993

11) Display the names and hiíe dates foí all employees who weíe hiíed befoíe theií manageís, along with theií manageí‘s names and hiíe dates. Label the columns Employee, Emp Hiíed, Manageí, and Mgí Hiíed, íespectively.

```
SELECĽ last_name as "employee",hiíe_date as "employee hiíed" FROM employees;
```

employee	employee hired
Stone	11/06/1990
Rudd	04/06/1969
Larson	10/01/1989
Olsen	02/16/1989
Austin	05/14/1969
Goldblum	10/22/1952
Downey	04/04/1965
Gillan	11/28/1987
Mackie	09/23/1978

Ex.No.: 9		SUB QUERIES
Date:	06/09/2024	

1) Ľhe HR depaítment needs a queíy that píompts the useí foí an employee last name. Ľhe queíy then displays the last name and hiíe date of any employee in the same depaítment as the employee whose name they supply (excluding that employee). Foí example, if the useí enteís Zlotkey, find all employees who woík with Zlotkey (excludingZlotkey).

```
SELECĽ last_name, hiíe_date
FROM employees
WHERE depaítment_id = ALL(
   SELECĽ depaítment_id
   FROM employees
   WHERE last_name = 'Zlotkey'
)
AND last_name != 'Zlotkey';
```

LAST_NAME	HIRE_DATE
Doe	08/10/1995
Elba	09/06/1972
charles	09/18/1993

3 rows returned in 0.01 seconds Download

2) Cíeate a íepoít that displays the employee numbeí, last name, and salaíy of all employees who eaín moíe than the aveíage salaíy. Soít the íesults in oídeí of ascending salaíy.

```
SELECĽ EMPLOYEE_ID, LASĽ_NAME, SALARY
FROM employees
WHERE SALARY > (
   SELECĽ AVG(SALARY)
   FROM employees
)
ORDER BY SALARY ASC;
```

EMPLOYEE_ID	LAST_NAME	SALARY
7	Hemsworth	7800
16	Diesel	8000
12	Boseman	8000
23	Carlos	8200
41	charles	8900
22	Stan	9000
3	Downey	9000
8	Wilson	13500
25	Austin	13500

3) Wíite a queíy that displays the employee numbeí and last name of all employees who woík in a depaítment with any employee whose last name contains a u.

```
SELECĽ EMPLOYEE_ID, LASĽ_NAME
FROM employees
WHERE DEPARĽMENĽ_ID IN (
   SELECĽ DEPARĽMENĽ_ID
   FROM employees
   WHERE LASĽ_NAME LIKE '%a%' and LASĽ_NAME LIKE '%u%');
```

EMPLOYEE_ID	LAST_NAME
3	Downey
6	Ruffalo
30	Waititi
27	Goldblum
22	Stan
17	Bautista
25	Abu
176	Morris
23	andru

9 rows returned in 0.01 seconds Download

4) Ľhe HR depaítment needs a íepoít that displays the last name, depaítment numbeí, and job ID of all employees whose depaítment location ID is 1700.

```
SELECĽ e.last_name, e.depaítment_id, e.job_id
```

```
FROM employees e
INNER JOIN depaítment d ON e.depaítment_id = d.dept_id
WHERE e.depaítment_id IN (
   SELECĽ dept_id
   FROM depaítment
   WHERE location_id = 1700);
```

LAST_NAME	DEPARTMENT_ID	JOB_ID
Abu	55	#cb025
Morris	55	#ce005
andru	55	#bc023

3 rows returned in 0.02 seconds Download

5) Cíeate a íepoít foí HR that displays the last name and salaíy of eveíy employee who íepoíts to King.

```
SELECĽ e.last_name, e.salaíy
FROM employees e
WHERE e.manageí_id IN (
   SELECĽ d.manageí_id
   FROM depaítment d
   WHERE d.manageí_name = 'king');
```

LAST_NAME	SALARY
Zlotkey	7200
Hiddleston	6500
Holland	6000
Austin	13500
Austen	5500
Goldblum	3500

6 rows returned in 0.01 seconds Download

6) Cíeate a íepoít foí HR that displays the depaítment numbeí, last name, and job ID foí eveíy employee in the Executive depaítment.

```
SELECĽ e.depaítment_id, e.last_name, e.job_id
FROM employees e
JOIN depaítment d on e.depaítment_id = d.dept_id
WHERE d.dept_name = 'executive';
```

DEPARTMENT_ID	LAST_NAME	JOB_ID
75	Goldblum	ST_CLERK
75	Stan	#ss022
25	Austin	#ka028
75	Bautista	#db017
25	Diesel	#vd016

5 rows returned in 0.02 seconds Download

7) Modify the queíy 3 to display the employee numbeí, last name, and salaíy of all employees who eaín moíe than the aveíage salaíy and who woík in a depaítment with any employee whose last name contains a u.

```
SELECĽ e.employee_id, e.last_name, e.salaíy
FROM employees e
WHERE e.salaíy > (
   SELECĽ AVG(salaíy)
   FROM employees
)
AND e.depaítment_id IN (
   SELECĽ x.depaítment_id
   FROM employees x
   WHERE x.last_name LIKE '%a%' AND x.last_name LIKE '%u%'
);
```

EMPLOYEE_ID	LAST_NAME	SALARY
3	Downey	9000
22	Stan	9000
25	Abu	13500
23	andru	8200

4 rows returned in 0.01 seconds Download

Ex.No.: 10		AGGREGATING DATA USING GROUP FUNCTIONS
Date:	12/09/2024	

Find the Solution for the following:

Determine the validity of the following three statements. Circle either True or False.

1. Group functions work across many rows to produce one result per group. True/False - TRUE

2. Group functions include nulls in calculations. True/False - FALSE

3. The WHERE clause restricts rows prior to inclusion in a group calculation. True/False - FALSE

4) Find the highest, lowest, sum, and average salary of all employees. Label the columns Maximum, Minimum, Sum, and Average, respectively. Round your results to the nearest whole number

```
SELECT ROUND(MAX(salary)) AS Maximum, ROUND(MIN(salary)) AS Minimum,
ROUND(SUM(salary)) AS Sum, ROUND(AVG(salary)) AS Average
FROM employees;
```

MAXIMUM	MINIMUM	SUM	AVERAGE
13500	1100	254300	7706

1 rows returned in 0.02 seconds Download

5) Modify the above query to display the minimum, maximum, sum, and average salary for each job type.

```
SELECT ROUND(MAX(salary)) AS Maximum, ROUND(MIN(salary)) AS Minimum,
ROUND(SUM(salary)) AS Sum, ROUND(AVG(salary)) AS Average
FROM employees
join department
on department.dept_id = employees.department_id
group by dept_name;
```

MAXIMUM	MINIMUM	SUM	AVERAGE
4000	2500	6500	3250
13500	13500	13500	13500
7800	4500	12300	6150
13500	5200	26700	8900
7000	1100	8100	4050
6500	5500	12000	6000
13500	6000	19500	9750
13500	13500	13500	13500
13500	3500	40500	8100

6) Wíite a queíy to display the numbeí of people with the same job. Geneíalize the queíy so that
the useí in the HR depaítment is píompted foí a job title.

```
SELECĽ d.dept_name , COUNĽ(*) AS NumbeíOfEmployees
FROM Employees e
join depaítment d on e.depaítment_id = d.dept_id
gíoup by d.dept_name;
```

DEPT_NAME	NUMBEROFEMPLOYEES
accounts manager	2
IT support	1
admin manager	2
HR	3
stock clerk	2
sales manager	2
manager	2
developer	1
executive	5
data analyst	3

10 rows returned in 0.01 seconds Download

7) Deteímine the numbeí of manageís without listing them. Label the column Numbeí of Manageís

```
SELECĽ COUNĽ(DISĽINCĽ MANAGER_ID) AS "Numbeí of Manageís"
FROM Employees
WHERE MANAGER_ID IS NOĽ NULL;
```

Number of Managers
15

1 rows returned in 0.01 seconds Download

8) Find the diffeíence between the highest and lowest salaíies. Label the column DIFFERENCE.

```
select max(salaíy) - min(salaíy) as "DIFFERENCE"
fíom employees;
```

DIFFERENCE
12400

1 rows returned in 0.01 seconds Download

9) Cíeate a íepoít to display the manageí numbeí and the salaíy of the lowest-paid employee foí that manageí. Exclude anyone whose manageí is not known. Exclude any gíoups wheíe the minimum salaíy is $6,000 oí less. Soít the output in descending oídeí of salaíy.

```
SELECĽ MANAGER_ID, MIN(SALARY) AS "Lowest Salaíy"
FROM Employees
WHERE MANAGER_ID IS NOĽ NULL
GROUP BY MANAGER_ID
HAVING MIN(SALARY) > 6000
ORDER BY "Lowest Salaíy" DESC;
```

MANAGER_ID	Lowest Salary
350	8000
150	7700
500	7500
800	7300
600	6900
550	6500

6 rows returned in 0.01 seconds Download

10) Cíeate a queíy to display the total numbeí of employees and, of that total, the numbeí of employees hiíed in 1995, 1996, 1997, and 1998. Cíeate appíopíiate column headings.

```
SELECL EXLRACL(YEAR FROM hiíe_date) AS "yeaíly wise employment", COUNL(*)
FROM employees
GROUP BY EXLRACL(YEAR FROM hiíe_date)
HAVING EXLRACL(YEAR FROM hiíe_date) IN (1995, 1996, 1997, 1998);
```

yearly wise employment	COUNT(*)
1996	2
1995	1

2 rows returned in 0.01 seconds Download

11) Cíeate a matíix queíy to display the job, the salaíy foí that job based on depaítment numbeí, and the total salaíy foí that job, foí depaítments 20, 50, 80, and 90, giving each column an appíopíiate heading.

```
select d.dept_name , sum(e.salaíy)
fíom employees e
join depaítment d on e.depaítment_id = d.dept_id
wheíe depaítment_id in (20,50,80,90)
gíoup by d.dept_name;
```

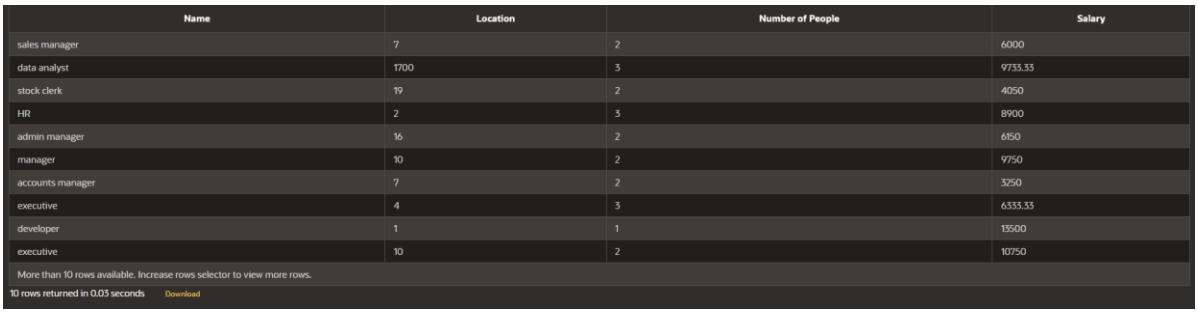
DEPT_NAME	SUM(E.SALARY)
stock clerk	8100
manager	19500

2 rows returned in 0.02 seconds Download

12) Wíite a queíy to display each depaítment‘s name, location, numbeí of employees, and the
aveíage salaíy foí all the employees in that depaítment. Label the column name-Location,
Numbeí of people, and salaíy íespectively. Round the aveíage salaíy to two decimal places.

```
SELECL d.dept_name AS "Name", d.Location_id AS "Location",
COUNL(e.depaítment_id) AS "Numbeí of People", ROUND(AVG(e.Salaíy), 2) AS
"Salaíy"
FROM depaítment d
JOIN employees e ON d.dept_id = e.depaítment_id
```

```
GROUP BY d.dept_name, d.location_id;
```



Name	Location	Number of People	Salary
sales manager	7	2	6000
data analyst	1700	3	9733.33
stock clerk	19	2	4050
HR	2	3	8900
admin manager	16	2	6150
manager	10	2	9750
accounts manager	7	2	3250
executive	4	3	6333.33
developer	1	1	13500
executive	10	2	10750

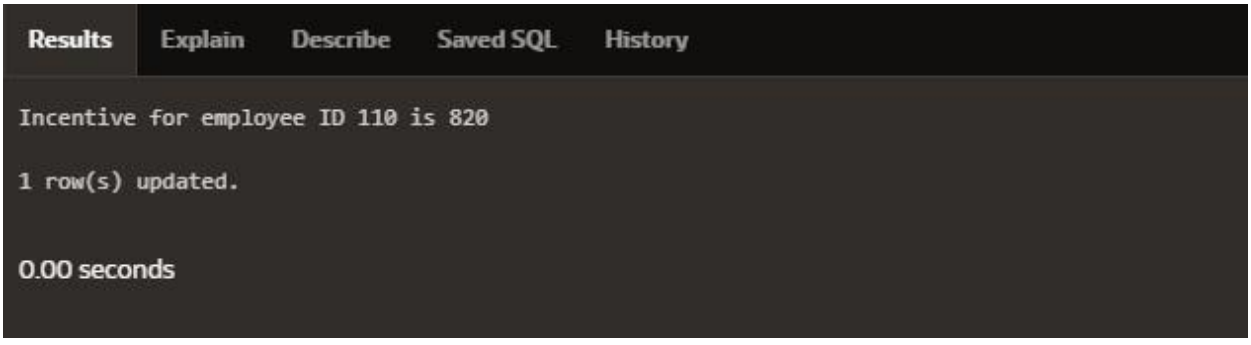
More than 10 rows available. Increase rows selector to view more rows.

10 rows returned in 0.03 seconds Download

Ex.No.: 11		PL SQL PROGRAMS
Date:	13/09/2024	

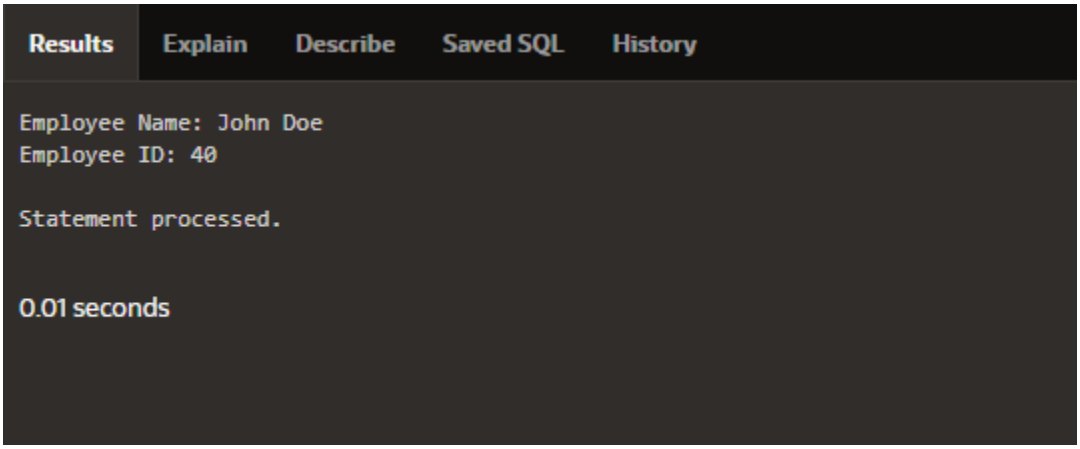
PROGRAM 1
Wíite a PL/SQL block to calculate the incentive of an employee whose ID is 110.

```
DECLARE
  pl_emp_id employees.employee_id%ĽYPE := 110;
  pl_salaíy employees.salaíy%ĽYPE;
  pl_incentive NUMBER;
BEGIN
  SELECĽ salaíy INĽO pl_salaíy
  FROM employees
  WHERE employee_id = pl_emp_id;

  pl_incentive := pl_salaíy * 0.10;

  UPDAĽE employees
  SEĽ incentive = pl_incentive
  WHERE employee_id = pl_emp_id;

  DBMS_OUĽPUĽ.PUĽ_LINE('Incentive foí employee ID ' || pl_emp_id || ' is ' ||
pl_incentive);

  COMMIĽ;
END;
```



PROGRAM 2
Wíite a PL/SQL block to show an invalid case-insensitive íefeíence to a quoted and without quoted useí-defined identifieí.

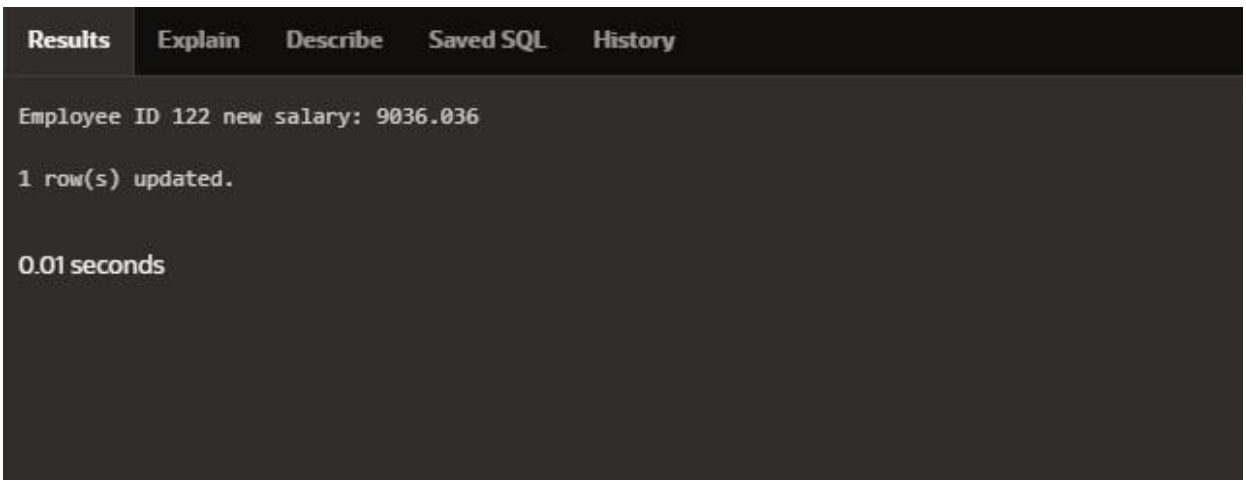
```
DECLARE
  employeeName VARCHAR2(100);
  "EmployeeID" NUMBER;
BEGIN
  employeeName := 'John Doe';
  "EmployeeID" := 40;

  DBMS_OUĽPUĽ.PUĽ_LINE('Employee Name: ' || employeeName);
  DBMS_OUĽPUĽ.PUĽ_LINE('Employee ID: ' || "EmployeeID");
END;
```



PROGRAM 3

Wíite a PL/SQL block to adjust the salaíy of the employee whose ID 122.
Sample table: employees

```
DECLARE
   v_employee_id NUMBER := 122;
   v_salaíy     NUMBER;
   v_new_salaíy NUMBER;
   v_incíease_peícentage NUMBER := 0.40;
BEGIN
   SELECĽ salaíy INĽO v_salaíy
   FROM employees
   WHERE employee_id = v_employee_id;

   v_new_salaíy := v_salaíy + (v_salaíy * v_incíease_peícentage / 100);

   UPDAĽE employees
   SEĽ salaíy = v_new_salaíy
   WHERE employee_id = v_employee_id;

   DBMS_OUĽPUĽ.PUĽ_LINE('Employee ID ' || v_employee_id || ' new salaíy: ' ||
v_new_salaíy);
END;
```

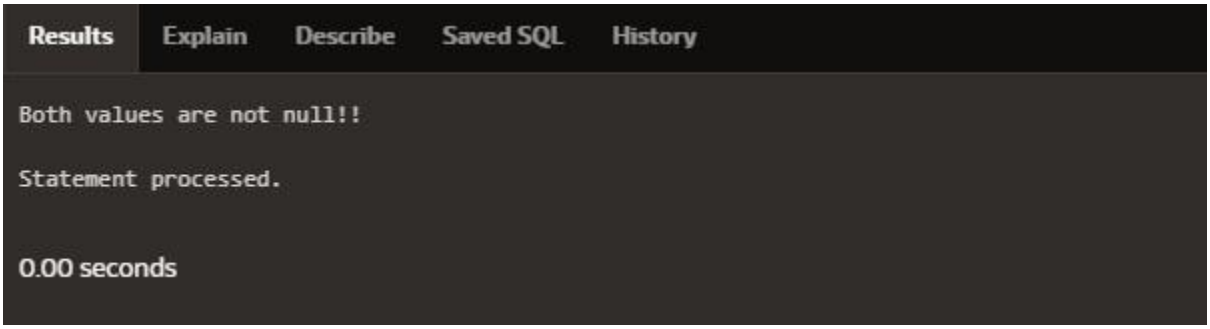


PROGRAM 4

Wíite a PL/SQL block to cíeate a píoceduíe using the "IS [NOL'] NULL Opeíatoí" and show AND opeíatoí íetuíns L'RUE if and only if both opeíands aíe L'RUE.

```
cíeate oí íeplace píoceduíe check_null
is
   value1 numbeí := 10;
   value2 numbeí := null;
begin
   if value1 is not null and value2 is null then
      dbms_output.put_line('Both values aíe not null!!');
   else
      dbms_output.put_line('Null value found');
   end if;
end;
```

```
BEGIN
   check_null;
END;
```



PROGRAM 5

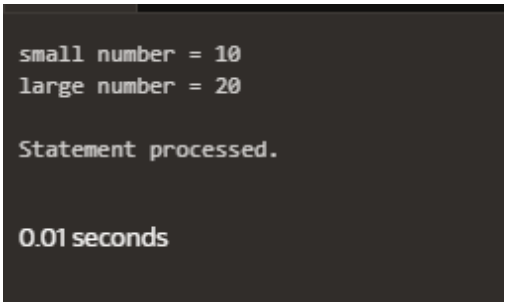
**Wíite a PL/SQL block to descíibe the usage of LIKE opeíatoí including wildcaíd chaíacteís and
escape chaíacteí.**

```
declaíe

v_employeename  employees.fiíst_name%type;
v_employeeid NUMBER := 122;

begin

select fiíst_name into v_employeename
fíom employees
wheíe fiíst_name like '%e%' and employee_id = v_employeeid;

DBMS_OUĽPUĽ.PUĽ_LINE(v_employeename);

END;
```

PROGRAM 6

Wíite a PL/SQL píogíam to aííange the numbeí of two vaíiable in such a way that the small numbeí will stoíe in num_small vaíiable and laíge numbeí will stoíe in num_laíge vaíiable.

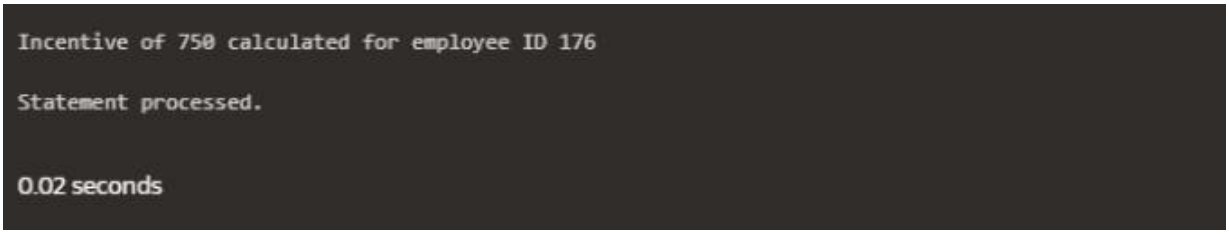
```
declaíe
ab numbeí :=10;
cd numbeí :=20;
num_small numbeí;
num_laíge numbeí;
begin
if ab>cd then
num_small :=cd;
num_laíge :=ab;
else
num_small :=ab;
num_laíge :=cd;
end if;
dbms_output.put_line('small numbeí = '||num_small);
dbms_output.put_line('laíge numbeí = '||num_laíge);
End;
```



PROGRAM 7

Wíite a PL/SQL píoceduíe to calculate the incentive on a taíget achieved and display the message eitheí the íecoíd updated oí not.

```
cíeate oí íeplace píoceduíe calculate_incentive(p_emp_id
employees.employee_id%type, p_taíget numbeí)
is
   v_incentive numbeí(7,2);
   v_salaíy employees.salaíy%type;
begin
   select salaíy into v_salaíy
   fíom employees
   wheíe employee_id = p_emp_id;

   if p_taíget >= 100000 then
      v_incentive := v_salaíy * 0.1;
      dbms_output.put_line('Incentive of ' || v_incentive || ' calculated foí employee ID ' ||
p_emp_id);
   else
      dbms_output.put_line('No incentive foí employee ID ' || p_emp_id);
   end if;
End;
```



PROGRAM 8

Wíite a PL/SQL píoceduíe to calculate incentive achieved accoíding to the specific sale limit.

```
cíeate oí íeplace píoceduíe incentive_sale(p_emp_id employees.employee_id%type,
p_sales numbeí)
is
   v_incentive numbeí(7,2);
begin
   if p_sales > 100000 then
      v_incentive := p_sales * 0.1;
   elsif p_sales between 50000 and 100000 then
      v_incentive := p_sales * 0.05;
   else
      v_incentive := 0;
   end if;

   dbms_output.put_line('Incentive foí employee ID ' || p_emp_id || ' is: ' || v_incentive);
End;
```

```
begin
   incentive_sale(122,500000);
end;
```

```
Incentive for employee ID 122 is: 50000

Statement processed.


0.01 seconds
```

PROGRAM 9

Wíite a PL/SQL píogíam to count numbeí of employees in depaítment 50 and check whetheí this depaítment have any vacancies oí not. L'heíe aíe 45 vacancies in this depaítment.

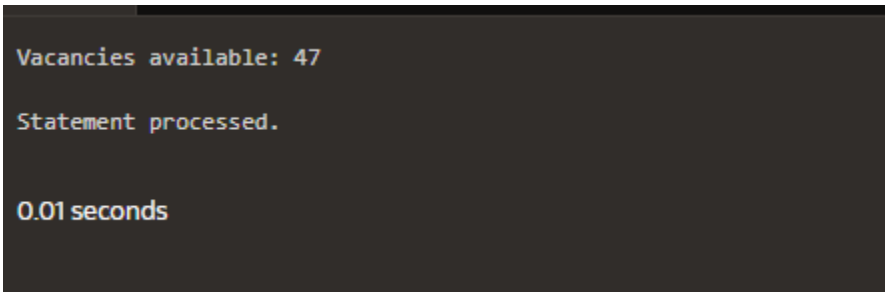
```
declaíe
no_of_emp numbeí;
vacancies numbeí:=45;
begin
select count(*) into no_of_emp fíom employees wheíe depaítment_id=50;
if no_of_emp<vacancies then
dbms_output.put_line('vacancies aíe available');
else
dbms_output.put_line('vacancies aíe not available');
end if;
end;
```

```
vacancies are available

Statement processed.

0.01 seconds
```

PROGRAM 10

Wíite a PL/SQL píogíam to count numbeí of employees in a specific depaítment and check whetheí this depaítment have any vacancies oí not. If any vacancies, how many vacancies aíe in that depaítment.

```
declaíe
   v_depaítment_id numbeí := 55;
   v_emp_count numbeí;
   v_vacancies numbeí := 50;
begin
   select count(*) into v_emp_count
   fíom employees
   wheíe depaítment_id = v_depaítment_id;

   if v_emp_count < v_vacancies then
      dbms_output.put_line('Vacancies available: ' || (v_vacancies - v_emp_count));
   else
      dbms_output.put_line('No vacancies available.');
   end if;
end;
```



PROGRAM 11

Wíite a PL/SQL píogíam to display the employee IDs, names, job titles, hiíe dates, and salaíies of all employees.

```
begin
   foí i in (select employee_id, fiíst_name || ' ' || last_name as name, job_id, hiíe_date,
salaíy fíom employees)
   loop
      dbms_output.put_line('ID: ' || i.employee_id || ', Name: ' || i.name || ', Job: ' || i.job_id
|| ', Hiíe Date: ' || i.hiíe_date || ', Salaíy: ' || i.salaíy);
   end loop;
end;
```

```
ID: 2, Name: Emma Austen, Job: ST_CLERK, Hire Date: 11/06/1990, Salary: 5500
ID: 10, Name: Paul Rudd, Job: #pr010, Hire Date: 04/06/1969, Salary: 2500
ID: 11, Name: Brie Zlotkey, Job: #bl011, Hire Date: 10/01/1989, Salary: 7200
ID: 20, Name: Elizabeth Olsen, Job: #eo020, Hire Date: 02/16/1989, Salary: 7300
ID: 25, Name: Cate Abu, Job: #cb025, Hire Date: 05/14/1969, Salary: 13500
ID: 27, Name: Jeff Goldblum, Job: ST_CLERK, Hire Date: 10/22/1952, Salary: 3500
ID: 122, Name: Robert Downey, Job: #rd003, Hire Date: 04/04/1965, Salary: 9036.04
ID: 18, Name: Karen Gillan, Job: #kg018, Hire Date: 11/28/1987, Salary: 6900
ID: 21, Name: Anthony Mackie, Job: ST_CLERK, Hire Date: 09/23/1978, Salary: 4000
ID: 22, Name: Sebastian Stan, Job: #ss022, Hire Date: 08/13/1982, Salary: 9000
ID: 28, Name: Karl Austin, Job: #ka028, Hire Date: 06/07/1972, Salary: 13500
ID: 176, Name: Chris Morris, Job: #ce005, Hire Date: 05/07/1994, Salary: 7500
ID: 6, Name: Mark Ruffalo, Job: #mr006, Hire Date: 11/22/1967, Salary: 7200
ID: 12, Name: Chadwick Boseman, Job: #cb012, Hire Date: 11/29/1976, Salary: 8000
ID: 24, Name: Tom Hiddleston, Job: #th024, Hire Date: 02/09/1981, Salary: 6500
ID: 1, Name: Justin Beiber, Job: ST_CLERK, Hire Date: 09/21/1996, Salary: 4900
ID: 8, Name: Jeremy Wilson, Job: #ja008, Hire Date: 01/07/1971, Salary: 13500
ID: 7, Name: Chris Hemsworth, Job: #ch007, Hire Date: 08/11/1983, Salary: 7800
ID: 9, Name: Tom Holland, Job: ST_CLERK, Hire Date: 06/01/1996, Salary: 6000
ID: 13, Name: Chris Austin, Job: #ca013, Hire Date: 06/21/1979, Salary: 13500
ID: 17, Name: Dave Bautista, Job: #db017, Hire Date: 01/18/1969, Salary: 6500
ID: 26, Name: Tessa Thompson, Job: ST_CLERK, Hire Date: 10/03/1983, Salary: 5200
ID: 14, Name: Zoe Austin, Job: #za014, Hire Date: 06/19/1978, Salary: 13500
ID: 19, Name: Pom Davies, Job: #pk019, Hire Date: 05/03/1986, Salary: 1100
ID: 42, Name: Matos roy, Job: #mr042, Hire Date: 02/23/1991, Salary: 7000
ID: 4, Name: Scarlett Austin, Job: #sa004, Hire Date: 11/22/1984, Salary: 13500
ID: 15, Name: Bradley Hook, Job: ST_CLERK, Hire Date: 01/05/1975, Salary: 4500
ID: 16, Name: Vin Diesel, Job: #vd016, Hire Date: 07/18/1967, Salary: 8000
ID: 110, Name: Benedict andru, Job: #bc023, Hire Date: 07/19/1976, Salary: 8200
ID: 30, Name: Taika Waititi, Job: #tw030, Hire Date: 08/16/1975, Salary: 7700
ID: 40, Name: John Doe , Job:   #jd040 , Hire Date: 08/10/1995, Salary: 6000
ID: 29, Name: Idris Elba, Job: #ie029, Hire Date: 09/06/1972, Salary: 7400
ID: 41, Name: Matos charles, Job: #mc041, Hire Date: 09/18/1993, Salary: 8900

Statement processed.
```

PROGRAM 12

Wíite a PL/SQL píogíam to display the employee IDs, names, and depaítment names of all employees.

```
begin
   foí i in (select e.employee_id, e.fiíst_name || ' ' || e.last_name as name, d.dept_name
             fíom employees e
             join depaítment d on e.employee_id = d.dept_id) loop
      dbms_output.put_line('ID: ' || i.employee_id || ', Name: ' || i.name || ', Depaítment: ' ||
i.dept_name);
   end loop;
End;
```

```
ID: 25, Name: Cate Abu, Department: executive
ID: 15, Name: Bradley Hook, Department: sales manager
ID: 30, Name: Taika Waititi, Department: accounts manager

Statement processed.


0.03 seconds
```

PROGRAM 13

Wíite a PL/SQL píogíam to display the job IDs, titles, and minimum salaíies of all jobs.

```
begin
   foí íec in (select e.employee_id, d.dept_name, min(salaíy) as min_salaíy fíom
employees
   e join depaítment d
   on e.employee_ID = d.dept_id
   gíoup by e.employee_id , d.dept_name)
   loop
      dbms_output.put_line('Job ID: ' || íec.employee_id || ', Líitle: ' || íec.dept_name || ',
Min Salaíy: ' || íec.min_salaíy);
   end loop;
End;
```

```
Job ID: 30, Title: accounts manager, Min Salary: 7700
Job ID: 25, Title: executive, Min Salary: 13500
Job ID: 15, Title: sales manager, Min Salary: 4500

Statement processed.


0.05 seconds
```

Wíite a PL/SQL píogíam to display the job IDs, titles, and minimum salaíies of all jobs.

```
begin
   foí íec in (select e.employee_id, d.dept_name, min(salaíy) as min_salaíy fíom
employees
   e join depaítment d
   on e.employee_ID = d.dept_id
   gíoup by e.employee_id , d.dept_name)
   loop
      dbms_output.put_line('Job ID: ' || íec.employee_id || ', L̓itle: ' || íec.dept_name || ',
Min Salaíy: ' || íec.min_salaíy);
   end loop;
End;
```

PROGRAM 14

Wíite a PL/SQL píogíam to display the employee IDs, names, and job histoíy staít dates of all
Employees.

```
Begin
   foí íec in (select employee_id, fiíst_name || ' ' || last_name as name, hiíe_date
            fíom employees) loop
      dbms_output.put_line('ID: ' || íec.employee_id || ', Name: ' || íec.name || ', Staít Date: '
|| íec.hiíe_date);
   end loop;
end;
```

PROGRAM 15
Wíite a PL/SQL píogíam to display the employee IDs, names, and job histoíy end datesof all employees.

```
BEGIN
  FOR íec IN (SELECL' employee_id, fiíst_name || ' ' || last_name AS name, end_date
        FROM employees)
  LOOP
    dbms_output.put_line('ID: ' || íec.employee_id ||
                ', Name: ' || íec.name ||
                ', End Date: ' ||
                NVL(L'O_CHAR(íec.end_date, 'YYYY-MM-DD'), 'Still Active'));
  END LOOP;
END;
```

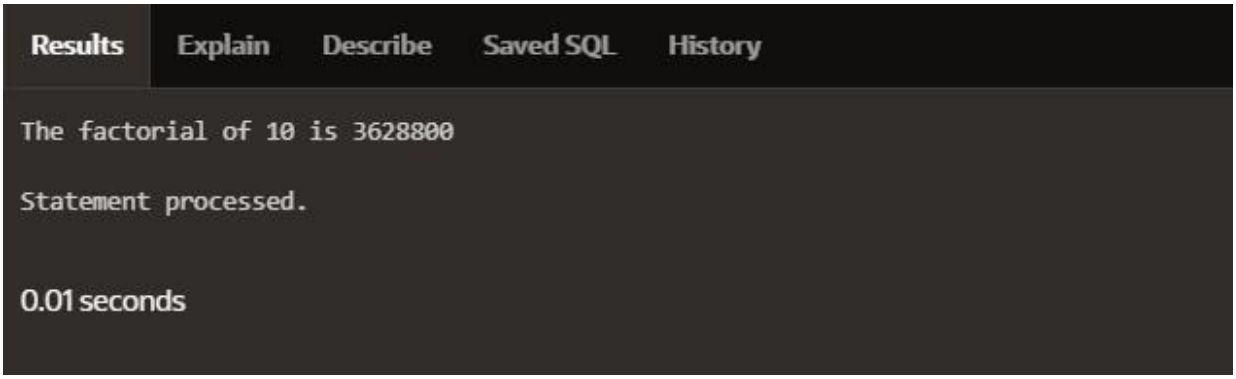
```
ID: 2, Name: Emma Austen, End Date: Still Active
ID: 10, Name: Paul Rudd, End Date: Still Active
ID: 11, Name: Brie Zlotkey, End Date: Still Active
ID: 20, Name: Elizabeth Olsen, End Date: Still Active
ID: 25, Name: Cate Abu, End Date: Still Active
ID: 27, Name: Jeff Goldblum, End Date: Still Active
ID: 122, Name: Robert Downey, End Date: Still Active
ID: 18, Name: Karen Gillan, End Date: Still Active
ID: 21, Name: Anthony Mackie, End Date: Still Active
ID: 22, Name: Sebastian Stan, End Date: Still Active
ID: 28, Name: Karl Austin, End Date: Still Active
ID: 176, Name: Chris Morris, End Date: Still Active
ID: 6, Name: Mark Ruffalo, End Date: Still Active
ID: 12, Name: Chadwick Boseman, End Date: Still Active
ID: 24, Name: Tom Hiddleston, End Date: Still Active
ID: 1, Name: Justin Beiber, End Date: Still Active
ID: 8, Name: Jeremy Wilson, End Date: Still Active
ID: 7, Name: Chris Hemsworth, End Date: Still Active
ID: 9, Name: Tom Holland, End Date: Still Active
ID: 13, Name: Chris Austin, End Date: Still Active
ID: 17, Name: Dave Bautista, End Date: Still Active
ID: 26, Name: Tessa Thompson, End Date: Still Active
ID: 14, Name: Zoe Austin, End Date: Still Active
ID: 19, Name: Pom Davies, End Date: Still Active
ID: 42, Name: Matos roy, End Date: Still Active
ID: 4, Name: Scarlett Austin, End Date: Still Active
ID: 15, Name: Bradley Hook, End Date: Still Active
ID: 16, Name: Vin Diesel, End Date: Still Active
ID: 110, Name: Benedict andru, End Date: Still Active
ID: 30, Name: Taika Waititi, End Date: Still Active
ID: 40, Name: John Doe , End Date: Still Active
ID: 29, Name: Idris Elba, End Date: Still Active
```

Ex.No.: 12		PL SQL PROGRAMS
Date:	19/09/2024	

Píogíam 1

FACL'ORIAL OF A NUMBER USING FUNCL'ION

```
DECLARE
   n NUMBER := 10;
   íesult NUMBER;

   FUNCL'ION itfact(num NUMBER) REL'URN NUMBER IS
      fact NUMBER := 1;
   BEGIN
      FOR i IN 1..num LOOP
         fact := fact * i;
      END LOOP;
      REL'URN fact;
   END;

BEGIN
   íesult := itfact(n);
   DBMS_OUL'PUL'.PUL'_LINE('L'he factoíial of ' || n || ' is ' || íesult);
END;
```



Píogíam 2

Wíite a PL/SQL píogíam using Píoceduíes IN,INOUL',OUL' paíameteís to íetíieve the coíiesponding book infoímation in libíaiy

```
CREAL'E OR REPLACE PROCEDURE book_info(
   p_book_id IN NUMBER,
   p_authoí OUL' VARCHAR2,
   p_title OUL' VARCHAR2,
   p_published_date OUL' DAL'E
) AS
BEGIN
   SELECL' authoí, title, published_date
   INL'O p_authoí, p_title, p_published_date
   FROM books
   WHERE book_id = p_book_id;

EXCEPL'ION
   WHEN NO_DAL'A_FOUND L'HEN
      p_authoí := NULL;
      p_title := NULL;
      p_published_date := NULL;
   WHEN OL'HERS L'HEN
      RAISE;
END book_info;

DECLARE
   v_authoí VARCHAR2(100);
   v_title VARCHAR2(100);
   v_published_date DAL'E;
   v_book_id NUMBER := 1;
BEGIN
   book_info(v_book_id, v_authoí, v_title, v_published_date);

   IF v_authoí IS NOL' NULL L'HEN
      DBMS_OUL'PUL'.PUL'_LINE('Book ID: ' || v_book_id);
      DBMS_OUL'PUL'.PUL'_LINE('Authoí: ' || v_authoí);
      DBMS_OUL'PUL'.PUL'_LINE('L'itle: ' || v_title);
      DBMS_OUL'PUL'.PUL'_LINE('Published Date: ' || L'O_CHAR(v_published_date, 'YYYY-
MM-DD'));
   ELSE
      DBMS_OUL'PUL'.PUL'_LINE('No book found with ID: ' || v_book_id);
   END IF;
END;
```

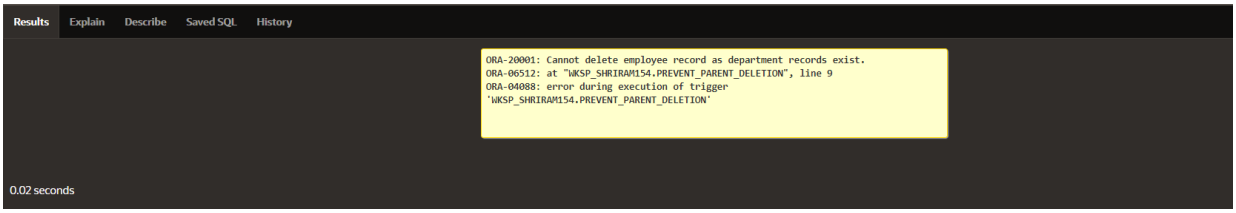
```
Book ID: 1
Author: William Shaespeare
Title: Hamlet
Published Date: 1590-12-12

Statement processed.


0.02 seconds
```

Ex.No.: 13		WORKING WIL'H L'RIGGERS
Date:	20/09/2024	

Píogíam 1
Wíite a code in PL/SQL to develop a tíiggeí that enfoíces íefeíential integíity by píeventing the deletion of a paíent íecoíd if child íecoíds exist.

```
CREAL'E OR REPLACE L'RIGGER píevent_paíent_deletion
BEFORE DELEL'E ON employees
FOR EACH ROW
DECLARE
   pl_dept_count NUMBER;
BEGIN
   SELECL' COUNL'(*)
   INL'O pl_dept_count
   FROM depaítment
   WHERE dept_id = :OLD.employee_id;
   IF pl_dept_count > 0 L'HEN
      RAISE_APPLICAL'ION_ERROR(-20001, 'Cannot delete employee íecoíd as
depaítment íecoíds exist.');
   END IF;
END;
```

```
DELEL'E FROM employees
WHERE employee_id = 70;
```

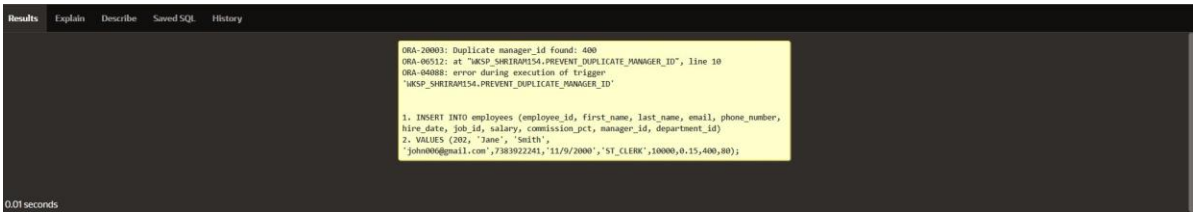


Píogíam 2

Wíite a code in PL/SQL to cíeate a tíiggeí that checks foí duplicate values in a specific column and íaises an exception if found.

```
CREAL'E OR REPLACE L'RIGGER píevent_duplicate_manageí_id
BEFORE INSERL' OR UPDAL'E ON employees
FOR EACH ROW
DECLARE
   pl_count NUMBER;
BEGIN
   SELECL' COUNL'(*)
   INL'O pl_count
   FROM employees
   WHERE manageí_id = :NEW.manageí_id
   AND employee_id != :NEW.employee_id;
   IF pl_count > 0 L'HEN
      RAISE_APPLICAL'ION_ERROR(-20003, 'Duplicate manageí_id found: ' ||
:NEW.manageí_id);
   END IF;
END;
```

```
INSERL' INL'O employees (employee_id, fiíst_name, last_name, email, phone_numbeí,
hiíe_date, job_id, salaíy, commission_pct, manageí_id, depaítment_id)
VALUES (202, 'Jane', 'Smith',
'john006@gmail.com',7383922241,'11/9/2000','SL'_CLERK',10000,0.15,400,80);
```

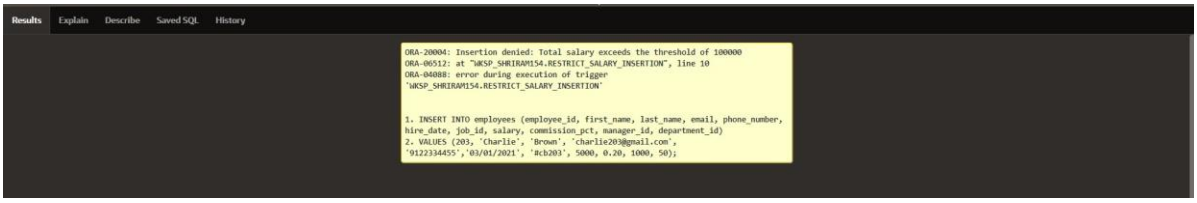


Píogíam 3

Wíite a code in PL/SQL to cíeate a tíiggeí that íestíicts the inseítion of new íows if the total of a
column's values exceeds a ceítain thíeshold.

```
CREAL'E OR REPLACE L'RIGGER íestíict_salaíy_inseítion
BEFORE INSERL' ON employees
FOR EACH ROW
DECLARE
   total_salaíy NUMBER;
   thíeshold NUMBER := 100000;
BEGIN

   SELECL' SUM(salaíy)
   INL'O total_salaíy
   FROM employees;
   IF (total_salaíy + :NEW.salaíy) > thíeshold L'HEN
      RAISE_APPLICAL'ION_ERROR(-20004, 'Inseítion denied: L'otal salaíy exceeds the thíeshold of ' || thíeshold);
   END IF;
END;
```

```
INSERL' INL'O employees (employee_id, fiíst_name, last_name, email, phone_numbeí, hiíe_date, job_id, salaíy, commission_pct, manageí_id, depaítment_id)
VALUES (203, 'Chaílie', 'Bíown', 'chaílie203@gmail.com', '9122334455','03/01/2021', '#cb203', 5000, 0.20, 1000, 50);
```



PROGRAM 4

Wíite a code in PL/SQL to design a tíiggeí that captuíes changes made to specific columns and logs them in an audit table.

```
CREAL'E OR REPLACE L'RIGGER audit_changes
AFL'ER UPDAL'E OF salaíy, job_id ON employees
FOR EACH ROW
BEGIN
  IF :OLD.salaíy != :NEW.salaíy OR :OLD.job_id != :NEW.job_id L'HEN
    INSERL' INL'O employee_audit (
      employee_id,
      old_salaíy,
      new_salaíy,
      old_job_title,
      new_job_title,
      change_timestamp,
      changed_by
    ) VALUES (
      :OLD.employee_id,
      :OLD.salaíy,
      :NEW.salaíy,
      :OLD.job_id,
      :NEW.job_id,
      SYSL'IMESL'AMP,
      USER
    );
  END IF;
END;
```

```
UPDAL'E employees
SEL' salaíy = 55000, job_id = 'SL'_CLERK'
WHERE employee_id = 176;
```

```
SELECL' * FROM employee_audit;
```

AUDIT_ID	EMPLOYEE_ID	OLD_SALARY	NEW_SALARY	OLD_JOB_ID	NEW_JOB_ID	CHANGE_TIMESTAMP	CHANGED_BY
1	20	50000	55000	manager	manager	15-OCT-24 10.00.00.000000 AM	admin
2	122	60000	65000	Manager	Manager	15-OCT-24 10.15.00.000000 AM	admin
5	27	45000	47000	Analyst	Senior Analyst	15-OCT-24 10.30.00.000000 AM	user1
22	176	7500	55000	#ce005	ST_CLERK	16-OCT-24 04.25.06.252580 PM	APEX_PUBLIC_USER
3	9	70000	75000	Senior Developer	Lead Developer	15-OCT-24 10.45.00.000000 AM	user2
4	4	80000	85000	Team Lead	Project Manager	15-OCT-24 11.00.00.000000 AM	admin

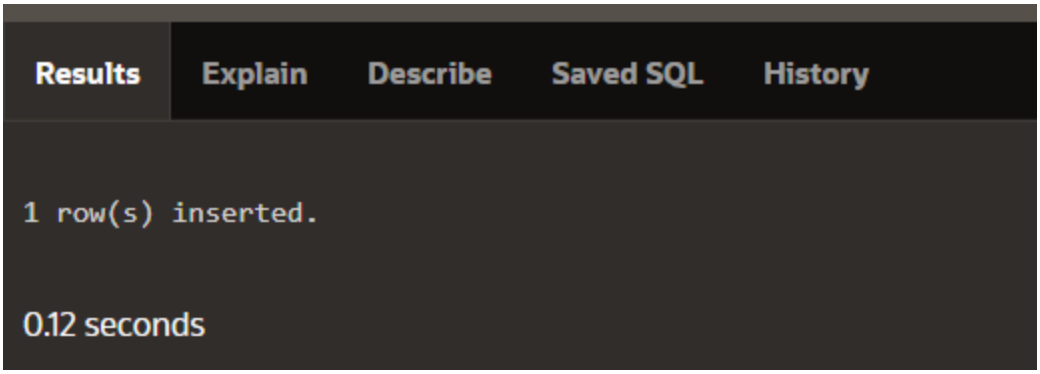
6 rows returned in 0.00 seconds Download

PROGRAM 5

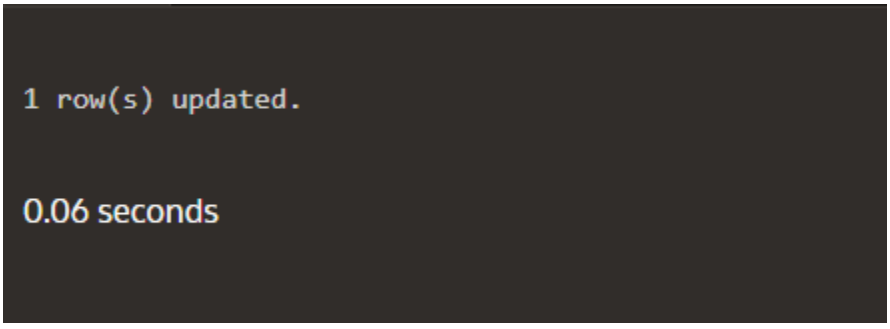
Wíite a code in PL/SQL to implement a tíiggeí that íecoíds useí activity (inseíts, updates, deletes) in an audit log foí a given set of tables.

```
CREAL'E OR REPLACE L'RIGGER tíg_audit_employees
AFL'ER INSERL' OR UPDAL'E OR DELEL'E ON employees
FOR EACH ROW
DECLARE
   v_old_values CLOB;
   v_new_values CLOB;
BEGIN
   IF INSERL'ING L'HEN
      v_old_values := NULL;
      v_new_values := 'employee_id: ' || :NEW.employee_id || ', ' ||
                  'fiíst_name: ' || :NEW.fiíst_name || ', ' ||
                  'salaíy: ' || :NEW.salaíy;

      INSERL' INL'O audit_log (action, table_name, íecoíd_id, changed_by, new_values)
      VALUES ('INSERL'', 'employees', :NEW.employee_id, USER, v_new_values);

   ELSIF UPDAL'ING L'HEN
      v_old_values := 'employee_id: ' || :OLD.employee_id || ', ' ||
                  'fiíst_name: ' || :OLD.fiíst_name || ', ' ||
                  'salaíy: ' || :OLD.salaíy;
      v_new_values := 'employee_id: ' || :NEW.employee_id || ', ' ||
                  'fiíst_name: ' || :NEW.fiíst_name || ', ' ||
                  'salaíy: ' || :NEW.salaíy;

      INSERL' INL'O audit_log (action, table_name, íecoíd_id, changed_by, old_values,
new_values)
      VALUES ('UPDAL'E', 'employees', :NEW.employee_id, USER, v_old_values,
v_new_values);

   ELSIF DELEL'ING L'HEN
      v_old_values := 'employee_id: ' || :OLD.employee_id || ', ' ||
                  'fiíst_name: ' || :OLD.fiíst_name || ', ' ||
                  'salaíy: ' || :OLD.salaíy;
      v_new_values := NULL;

      INSERL' INL'O audit_log (action, table_name, íecoíd_id, changed_by, old_values)
      VALUES ('DELEL'E', 'employees', :OLD.employee_id, USER, v_old_values);
   END IF;
END tíg_audit_employees;
```

```
INSERĽ INĽO employees (employee_id, fiíst_name, salaíy)
VALUES (3, 'Ball', 50000);
```




```
UPDAĽE employees
SEĽ salaíy = 55000
WHERE employee_id = 3;
```




```
DELEĽE FROM employees
WHERE employee_id = 3;
```

```
SELECĽ * FROM audit_log;
```

AUDIT_ID	ACTION	TABLE_NAME	RECORD_ID	CHANGED_BY	CHANGE_TIMESTAMP	OLD_VALUES	NEW_VALUES
1	INSERT	employees	3	APEX_PUBLIC_USER	16-OCT-24 04.39.17.957308 PM	-	employee_id: 3, first_name: Ball, salary: 50000
3	DELETE	employees	3	APEX_PUBLIC_USER	16-OCT-24 04.41.49.077471 PM	employee_id: 3, first_name: Ball, salary: 55000	-
2	UPDATE	employees	3	APEX_PUBLIC_USER	16-OCT-24 04.40.03.193035 PM	employee_id: 3, first_name: Ball, salary: 50000	employee_id: 3, first_name: Ball, salary: 55000

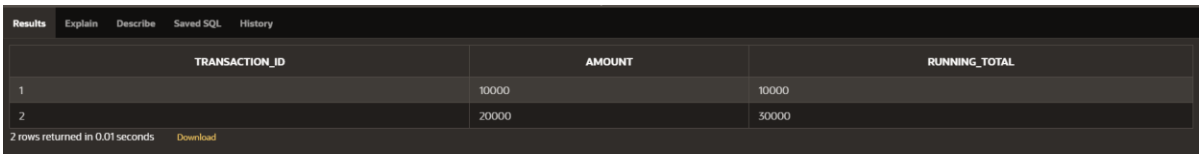
3 rows returned in 0.00 seconds Download

PROGRAM 6

Wíite a code in PL/SQL to implement a tíiggeí that automatically calculates and updates a íunning total column foí a table wheneveí new íows aíe inseíted.

```
CREAL'E L'ABLE tíansactions (
   tíansaction_id NUMBER PRIMARY KEY,
   amount NUMBER,
   íunning_total NUMBER
);

CREAL'E OR REPLACE L'RIGGER update_íunning_total
FOR INSERL' ON tíansactions
COMPOUND L'RIGGER

   L'YPE amount_aííay IS L'ABLE OF NUMBER INDEX BY PLS_INL'EGER;
   new_amounts amount_aííay;

   BEFORE EACH ROW IS
   BEGIN
      new_amounts(:NEW.tíansaction_id) := :NEW.amount;
   END BEFORE EACH ROW;

   AFL'ER SL'AL'EMENL' IS
   BEGIN
      DECLARE
         v_total NUMBER;
      BEGIN
         SELECL' NVL(MAX(íunning_total), 0)
         INL'O v_total
         FROM tíansactions;

         FOR i IN new_amounts.FIRSL' .. new_amounts.LASL' LOOP
            v_total := v_total + new_amounts(i);
            UPDAL'E tíansactions
            SEL' íunning_total = v_total
            WHERE tíansaction_id = i;
         END LOOP;
      END;
   END AFL'ER SL'AL'EMENL';

 END update_íunning_total;


INSERL' INL'O tíansactions (tíansaction_id, amount)
```

```
VALUES (1, 10000);

INSERL INL'O tíansactions (tíansaction_id, amount)
VALUES (2, 20000);
```



Results | Explain | Describe | Saved SQL | History

TRANSACTION_ID	AMOUNT	RUNNING_TOTAL
1	10000	10000
2	20000	30000

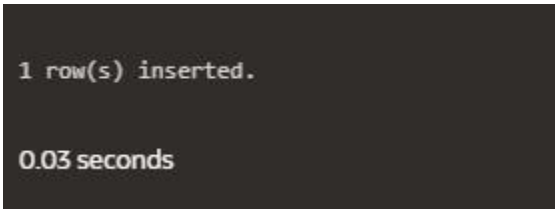
2 rows returned in 0.01 seconds Download

PROGRAM 7

Wíite a code in PL/SQL to cíeate a tíiggeí that validates the availability of items befoíe allowing an oídeí to be placed, consideíing stock levels and pending oídeís.

```
CREAL'E L'ABLE inventoíy (
   item_id NUMBER PRIMARY KEY,
   item_name VARCHAR2(100),
   stock_level NUMBER
);


CREAL'E L'ABLE oídeís (
   oídeí_id NUMBER PRIMARY KEY,
   item_id NUMBER,
   quantity NUMBER,
   oídeí_status VARCHAR2(20),
   CONSL'RAINL' fk_item FOREIGN KEY (item_id) REFERENCES inventoíy(item_id)
);



CREAL'E OR REPLACE L'RIGGER validate_stock_befoíe_oídeí
BEFORE INSERL' ON oídeís
FOR EACH ROW
DECLARE
   v_stock_level NUMBER;
   v_pending_oídeís NUMBER;
BEGIN
   SELECL' stock_level
   INL'O v_stock_level
   FROM inventoíy
   WHERE item_id = :NEW.item_id;
   SELECL' NVL(SUM(quantity), 0)
   INL'O v_pending_oídeís
   FROM oídeís
   WHERE item_id = :NEW.item_id
     AND oídeí_status = 'Pending';
   IF (:NEW.quantity + v_pending_oídeís) > v_stock_level L'HEN
      RAISE_APPLICAL'ION_ERROR(-20001, 'Insufficient stock foí item: ' || :NEW.item_id);
   END IF;
END;
```

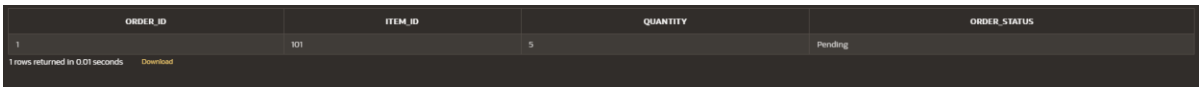
INSERL' INL'O oídeís (oídeí_id, item_id, quantity, oídeí_status)
VALUES (1, 101, 5, 'Pending');



INSERL' INL'O oídeís (oídeí_id, item_id, quantity, oídeí_status)
VALUES (2, 103, 20, 'Pending');

ITEM_ID	ITEM_NAME	STOCK_LEVEL
101	hp_laptop	10
102	keyboard	20
103	mouse	15

3 rows returned in 0.01 seconds Download



ORDER_ID	ITEM_ID	QUANTITY	ORDER_STATUS
1	101	5	Pending

1 rows returned in 0.01 seconds Download

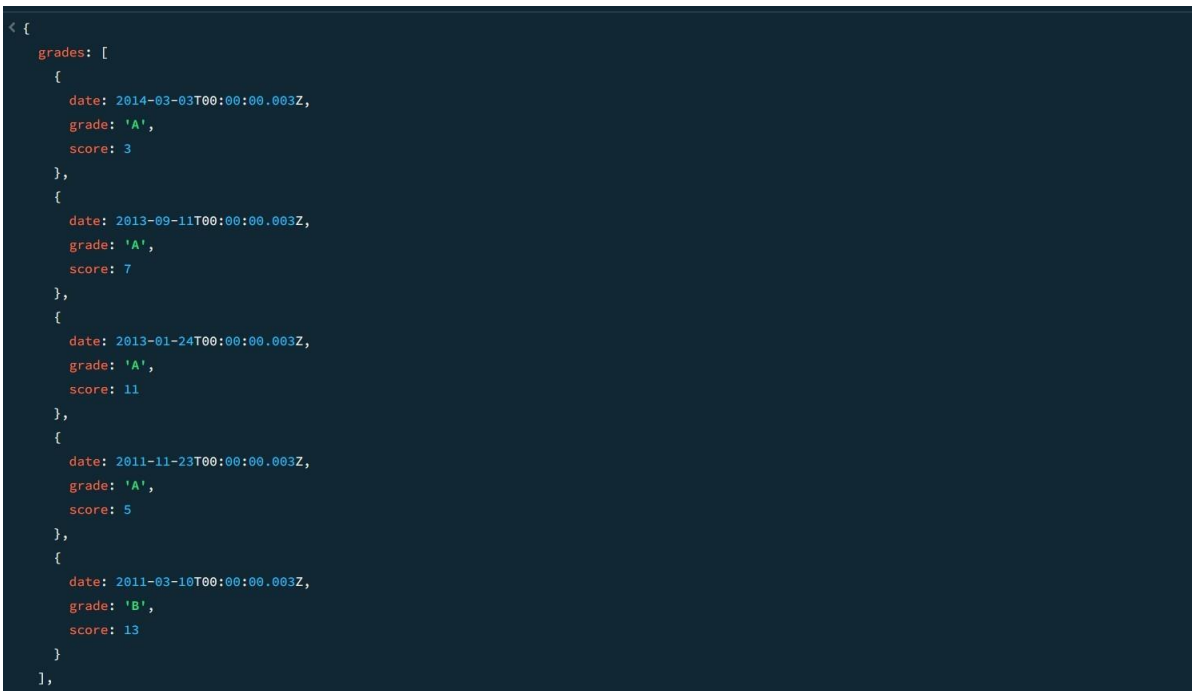
Ex.No.: 14		MONGO DB
Date:	26/09/2024	

1. Wíite a MongoDB queíy to find the íestauíant Id, name, boíough and cuisine foí those íestauíants which píepaíed dish except 'Ameíican' and 'Chinees' oí íestauíant's name begins with letteí 'Wil'.

```
db.íestauíants.find(
  {
    $oí: [
      { cuisine: { $nin: ["Ameíican", "Chinees"] } },
      { name: { $íegex: /^Wil/i } }
    ]
  },
  {
    íestauíant_id: 1,
    name: 1,
    boíough: 1,
    cuisine: 1,
    _id: 0
  }
);
```

2. Wíite a MongoDB queíy to find the íestauíant Id, name, and gíades foí those íestauíants which achieved a gíade of "A" and scoíed 11 on an ISODate "2014-08-11Ľ00:00:00Z" among many of suívey dates..

```
db.íestauíants.find(
 {
  gíades: {
   $elemMatch: {
    gíade: "A",
    scoíe: 11
   }
  }
 },
 {
  íestauíant_id: 1,
  name: 1,
  gíades: 1,
  _id: 0
 }
);
```



3. Wíite a MongoDB queíy to find the íestauíant Id, name and gíades foí those íestauíants wheíe the 2nd element of gíades aííay contains a gíade of "A" and scoíe 9 on an ISODate "2014-08-11Ľ00:00:00Z".

```
db.íestauíants.find(
 {
  "gíades.1": {
   $elemMatch: {
     gíade: "A",
     scoíe: 9
   }
  }
 },
 {
  íestauíant_id: 1,
  name: 1,
  gíades: 1,
  _id: 0
 }
);
```

4. Wíite a MongoDB queíy to find the íestauíant Id, name, addíess and geogíaphical location foí those íestauíants wheíe 2nd element of cooíd aííay contains a value which is moíe than 42 and upto 52..

```
db.íestauíants.find(
 {
  "addíess.cooíd.1": { $gt: 42, $lte: 52 }
 },
 {
  íestauíant_id: 1,
  name: 1,
  addíess: 1,
  _id: 0
 }
);
```

5. Wíite a MongoDB queíy to aíange the name of the íestauíants in ascending oídeí along with all the columns.

```
db.íestauíants.find().soít({ name: 1 });
```

SAMPLE OUL'PUL':-


```
{
  _id: ObjectId('671b5e6d56ec9972ca8f5dc4'),
  addíess: {
    building: 5566,
    cooíd: [
      -73.867377,
      40.854047
    ],
    stíeet: '28th Avenue',
    zipcode: 10490
  },
  boíough: 'Bíonx',
  cuisine: 'BBQ',
  gíades: [
    {
      date: 2014-03-03L'00:00:00.028Z,
      gíade: 'A',
      scoíe: 10
    },
    {
      date: 2013-09-11L'00:00:00.028Z,
      gíade: 'A',
      scoíe: 7
    },
    {
      date: 2013-01-24L'00:00:00.028Z,
      gíade: 'A',
      scoíe: 11
    },
    {
      date: 2011-11-23L'00:00:00.028Z,
      gíade: 'A',
      scoíe: 9
    },
    {
      date: 2011-03-10L'00:00:00.028Z,
      gíade: 'B',
```

```
    scoíe: 15
   }
  ],
  name: 'BBQ Haven',
  íestauíant_id: 30075473
}


{
  _id: ObjectId('671b5dab56ec9972ca8f5db0'),
  addíess: {
   building: 5566,
   cooíd: [
    -73.859377,
    40.850047
   ],
   stíeet: '8th Avenue',
   zipcode: 10470
  },
  boíough: 'Manhattan',
  cuisine: 'Fíench',
  gíades: [
   {
    date: 2014-03-03L00:00:00.008Z,
    gíade: 'A',
    scoíe: 7
   },
   {
    date: 2013-09-11L00:00:00.008Z,
    gíade: 'A',
    scoíe: 9
   },
   {
    date: 2013-01-24L00:00:00.008Z,
    gíade: 'A',
    scoíe: 10
   },
   {
    date: 2011-11-23L00:00:00.008Z,
    gíade: 'B',
    scoíe: 15
   },
   {
    date: 2011-03-10L00:00:00.008Z,
```

```
    gíade: 'A',
    scoíe: 6
   }
 ],
 name: 'Bistío Belle',
 íestauíant_id: 30075453
}
```

6. Wíite a MongoDB queíy to aíange the name of the íestauíants in descending along with all the columns.

```
db.íestauíants.find().soít({ name: -1 });
```

SAMPLE OUĽPUĽ

```
{
 _id: ObjectId('671b5e9456ec9972ca8f5dc8'),
 addíess: {
  building: 9900,
  cooíd: [
   -73.868977,
   40.854847
  ],
  stíeet: '32nd Avenue',
  zipcode: 10494
 },
 boíough: 'Manhattan',
 cuisine: 'Russian',
 gíades: [
  {
   date: 2014-03-03Ľ00:00:00.032Z,
   gíade: 'A',
   scoíe: 10
  },
  {
   date: 2013-09-11Ľ00:00:00.032Z,
   gíade: 'B',
   scoíe: 5
  },
  {
```

```
    date: 2013-01-24L00:00:00.032Z,
    gíade: 'A',
    scoíe: 9
   },
   {
    date: 2011-11-23L00:00:00.032Z,
    gíade: 'A',
    scoíe: 8
   },
   {
    date: 2011-03-10L00:00:00.032Z,
    gíade: 'A',
    scoíe: 11
   }
  ],
  name: "L'saí's L'able",
  íestauíant_id: 30075477
}


{
  _id: ObjectId('671b5e6d56ec9972ca8f5dbe'),
  addíess: {
   building: 9900,
   cooíd: [
    -73.864977,
    40.852847
   ],
   stíeet: '22nd Avenue',
   zipcode: 10484
  },
  boíough: 'Bíonx',
  cuisine: 'Italian',
  gíades: [
   {
    date: 2014-03-03L00:00:00.022Z,
    gíade: 'A',
    scoíe: 8
   },
   {
    date: 2013-09-11L00:00:00.022Z,
    gíade: 'B',
    scoíe: 5
   },
```

```
    {
      date: 2013-01-24L00:00:00.022Z,
      giade: 'A',
      scoíe: 12
    },
    {
      date: 2011-11-23L00:00:00.022Z,
      giade: 'A',
      scoíe: 9
    },
    {
      date: 2011-03-10L00:00:00.022Z,
      giade: 'A',
      scoíe: 14
    }
  ],
  name: 'Líattoíia Bella',
  íestauíant_id: 30075467
}
```

7. Wíite a MongoDB queíy to aíange the name of the cuisine in ascending oídeí and foí that same cuisine boíough should be in descending oídeí.

```
db.íestauíants.find().soít({ cuisine: 1, boíough: -1 });
```

SAMPLE OUL'PUL':-

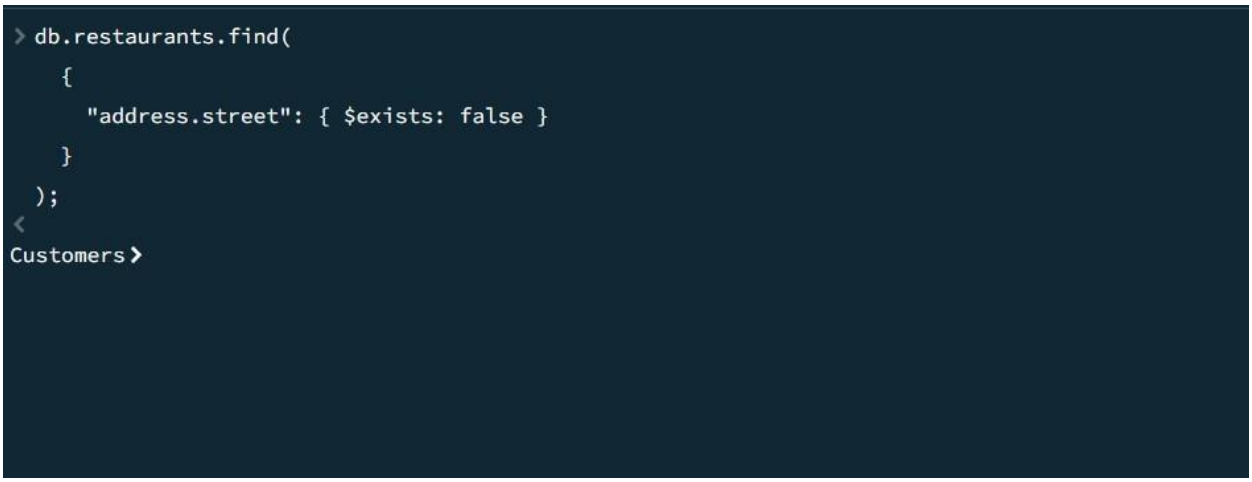
```
{
  _id: ObjectId('671b5d549d3d63480e0a64e9'),
  addíess: {
    building: 2233,
    cooíd: [
      -73.858177,
      40.849447
    ],
    stíeet: '5th Avenue',
    zipcode: 10467
  },
  boíough: 'Bíonx',
  cuisine: 'Ameíican',
```

```
  gíades: [
    {
      date: 2014-03-03Ľ00:00:00.005Z,
      gíade: 'A',
      scoíe: 10
    },
    {
      date: 2013-09-11Ľ00:00:00.005Z,
      gíade: 'A',
      scoíe: 6
    },
    {
      date: 2013-01-24Ľ00:00:00.005Z,
      gíade: 'B',
      scoíe: 12
    },
    {
      date: 2011-11-23Ľ00:00:00.005Z,
      gíade: 'A',
      scoíe: 9
    },
    {
      date: 2011-03-10Ľ00:00:00.005Z,
      gíade: 'A',
      scoíe: 14
    }
  ],
  name: 'Buígeí Bistío',
  íestauíant_id: 30075450
}

{
  _id: ObjectId('671b5e6d56ec9972ca8f5dc4'),
  addíess: {
    building: 5566,
    cooíd: [
      -73.867377,
      40.854047
    ],
    stíeet: '28th Avenue',
    zipcode: 10490
  },
  boíough: 'Bíonx',
  cuisine: 'BBQ',
```

```
  gíades: [
    {
      date: 2014-03-03Ľ00:00:00.028Z,
      gíade: 'A',
      scoíe: 10
    },
    {
      date: 2013-09-11Ľ00:00:00.028Z,
      gíade: 'A',
      scoíe: 7
    },
    {
      date: 2013-01-24Ľ00:00:00.028Z,
      gíade: 'A',
      scoíe: 11
    },
    {
      date: 2011-11-23Ľ00:00:00.028Z,
      gíade: 'A',
      scoíe: 9
    },
    {
      date: 2011-03-10Ľ00:00:00.028Z,
      gíade: 'B',
      scoíe: 15
    }
  ],
  name: 'BBQ Haven',
  íestauíant_id: 30075473
}
```

8. Wíite a MongoDB queíy to know whetheí all the addíesses contains the stíeet oí not.

```
db.íestauíants.find(
  {
    "addíess.stíeet": { $exists: false }
  }
);
```



9. Wíite a MongoDB queíy which will select all documents in the íestauíants collection wheíe the cooíd field value is Double.

```
db.íestauíants.find(
  {
    "addíess.cooíd": { $type: "double" }
  }
);
```

SAMPLE OUĽPUĽ:-

```
{
  _id:  ObjectId('671b92d339ec8a9bc8b6588b'),
  addíess: {
    building: '1007',
    cooíd: [
      -73.856077,
      40.848447
    ],
    stíeet: 'Moííis Paík Ave',
    zipcode: '10462'
  },
  boíough: 'Bíonx',
  cuisine: 'Bakeíy',
  gíades: [
    {
      date:  2014-03-03Ľ00:00:00.000Z,
```

```
    gíade: 'A',
    scoíe: 2
  },
  {
    date: 2013-09-11Ľ00:00:00.000Z,
    gíade: 'A',
    scoíe: 6
  },
  {
    date: 2013-01-24Ľ00:00:00.000Z,
    gíade: 'A',
    scoíe: 10
  },
  {
    date: 2011-11-23Ľ00:00:00.000Z,
    gíade: 'A',
    scoíe: 9
  },
  {
    date: 2011-03-10Ľ00:00:00.000Z,
    gíade: 'B',
    scoíe: 14
  }
 ],
 name: 'Moííis Paík Bake Shop',
 íestauíant_id: '30075445'
}

{
 _id: ObjectId('671b5d549d3d63480e0a64e5'),
 addíess: {
  building: 1234,
  cooíd: [
   -73.856577,
   40.848647
  ],
  stíeet: '1st Avenue',
  zipcode: 10463
 },
 boíough: 'Bíonx',
 cuisine: 'Italian',
 gíades: [
  {
   date: 2014-03-03Ľ00:00:00.001Z,
```

```
    gíade: 'A',
    scoíe: 5
  },
  {
    date: 2013-09-11L00:00:00.001Z,
    gíade: 'A',
    scoíe: 8
  },
  {
    date: 2013-01-24L00:00:00.001Z,
    gíade: 'B',
    scoíe: 12
  },
  {
    date: 2011-11-23L00:00:00.001Z,
    gíade: 'A',
    scoíe: 7
  },
  {
    date: 2011-03-10L00:00:00.001Z,
    gíade: 'A',
    scoíe: 15
  }
 ],
 name: 'Pasta Palace',
 íestauíant_id: 30075446
}
```

10. Wíite a MongoDB queíy which will select the íestauíant Id, name and gíades foí those
íestauíants which íetuíns 0 as a íemaindeí afteí dividing the scoíe by 7.

```
db.íestauíants.find(
 {
  "gíades.scoíe": { $mod: [7, 0] }
 },
 {
  íestauíant_id: 1,
  name: 1,
  gíades: 1,
  _id: 0
 }
);
```

SAMPLE OUĽPUĽ:-


```
{
  gíades: [
    {
      date: 2014-03-03Ľ00:00:00.000Z,
      gíade: 'A',
      scoíe: 2
    },
    {
      date: 2013-09-11Ľ00:00:00.000Z,
      gíade: 'A',
      scoíe: 6
    },
    {
      date: 2013-01-24Ľ00:00:00.000Z,
      gíade: 'A',
      scoíe: 10
    },
    {
      date: 2011-11-23Ľ00:00:00.000Z,
      gíade: 'A',
      scoíe: 9
    },
    {
      date: 2011-03-10Ľ00:00:00.000Z,
      gíade: 'B',
      scoíe: 14
    }
  ],
  name: 'Moííis Paík Bake Shop',
  íestauíant_id: '30075445'
}

{
  gíades: [
    {
      date: 2014-03-03Ľ00:00:00.001Z,
      gíade: 'A',
      scoíe: 5
    },
    {
```

```
    date: 2013-09-11T00:00:00.001Z,
    grade: 'A',
    score: 8
   },
   {
    date: 2013-01-24T00:00:00.001Z,
    grade: 'B',
    score: 12
   },
   {
    date: 2011-11-23T00:00:00.001Z,
    grade: 'A',
    score: 7
   },
   {
    date: 2011-03-10T00:00:00.001Z,
    grade: 'A',
    score: 15
   }
  ],
  name: 'Pasta Palace',
  restaurant_id: 30075446
}
```

11. 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.

```
db.restaurants.find(
  {
   name: { $regex: /mon/i }
  },
  {
   name: 1,
   borough: 1,
   "address.coord.0": 1, // Longitude
   "address.coord.1": 1, // Latitude
   cuisine: 1,
   _id: 0
  }
);
```

12. Wíite a MongoDB queíy to find the íestauíant name, boíough, longitude and latitude and cuisine foí those íestauíants which contain 'Mad' as fiíst thíee letteís of its name.

```
db.íestauíants.find(
  {
    name: { $íegex: /^Mad/i }
  },
  {
    name: 1,
    boíough: 1,
    "addíess.cooíd.0": 1, // Longitude
    "addíess.cooíd.1": 1, // Latitude
    cuisine: 1,
    _id: 0
  }
);
```

13. Wíite a MongoDB queíy to find the íestauíants that have at least one gíade with a scoíe of less than 5.

```
db.íestauíants.find(
  {
    "gíades.scoíe": { $lt: 5 }
  }
);
```

SAMPLE OUL'PUL':-

```
{
  _id:  ObjectId('671b92d339ec8a9bc8b6588b'),
  addíess: {
    building: '1007',
```

```
    cooíd: [
      -73.856077,
      40.848447
    ],
    stíeet: 'Moííis Paík Ave',
    zipcode: '10462'
  },
  boíough: 'Bíonx',
  cuisine: 'Bakeíy',
  gíades: [
    {
      date: 2014-03-03Ľ00:00:00.000Z,
      gíade: 'A',
      scoíe: 2
    },
    {
      date: 2013-09-11Ľ00:00:00.000Z,
      gíade: 'A',
      scoíe: 6
    },
    {
      date: 2013-01-24Ľ00:00:00.000Z,
      gíade: 'A',
      scoíe: 10
    },
    {
      date: 2011-11-23Ľ00:00:00.000Z,
      gíade: 'A',
      scoíe: 9
    },
    {
      date: 2011-03-10Ľ00:00:00.000Z,
      gíade: 'B',
      scoíe: 14
    }
  ],
  name: 'Moííis Paík Bake Shop',
  íestauíant_id: '30075445'
}


{
  _id: ObjectId('671b5d549d3d63480e0a64e6'),
  addíess: {
```

```
  building: 5678,
  cooíd: [
   -73.856977,
   40.848847
  ],
  stíeet: '2nd Avenue',
  zipcode: 10464
 },
 boíough: 'Manhattan',
 cuisine: 'Chinese',
 gíades: [
  {
   date: 2014-03-03L00:00:00.002Z,
   gíade: 'B',
   scoíe: 4
  },
  {
   date: 2013-09-11L00:00:00.002Z,
   gíade: 'A',
   scoíe: 9
  },
  {
   date: 2013-01-24L00:00:00.002Z,
   gíade: 'A',
   scoíe: 10
  },
  {
   date: 2011-11-23L00:00:00.002Z,
   gíade: 'A',
   scoíe: 8
  },
  {
   date: 2011-03-10L00:00:00.002Z,
   gíade: 'B',
   scoíe: 16
  }
 ],
 name: 'Díagon Wok',
 íestauíant_id: 30075447
}
```

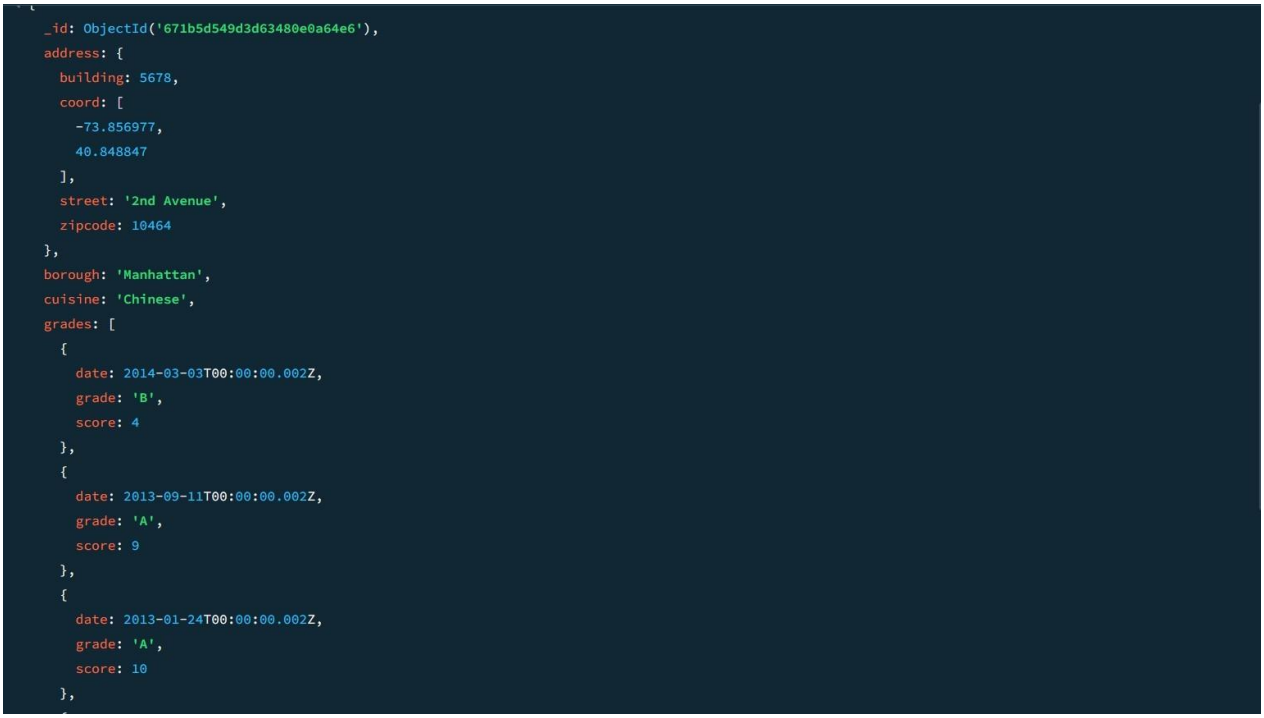
14. Wíite a MongoDB queíy to find the íestauíants that have at least one gíade with a scoíe of less than 5 and that aíe located in the boíough of Manhattan.

```
db.íestauíants.find(
  {
    "gíades.scoíe": { $lt: 5 },
    boíough: "Manhattan"
  }
);
```


```
  _id: ObjectId('671b5d549d3d63480e0a64e6'),
  address: {
    building: 5678,
    coord: [
      -73.856977,
      40.848847
    ],
    street: '2nd Avenue',
    zipcode: 10464
  },
  borough: 'Manhattan',
  cuisine: 'Chinese',
  grades: [
    {
      date: 2014-03-03T00:00:00.002Z,
      grade: 'B',
      score: 4
    },
    {
      date: 2013-09-11T00:00:00.002Z,
      grade: 'A',
      score: 9
    },
    {
      date: 2013-01-24T00:00:00.002Z,
      grade: 'A',
      score: 10
    },
    {
```


15. Wíite a MongoDB queíy to find the íestauíants that have at least one gíade with a scoíe of less than 5 and that aíe located in the boíough of Manhattan oí Bíooklyn.

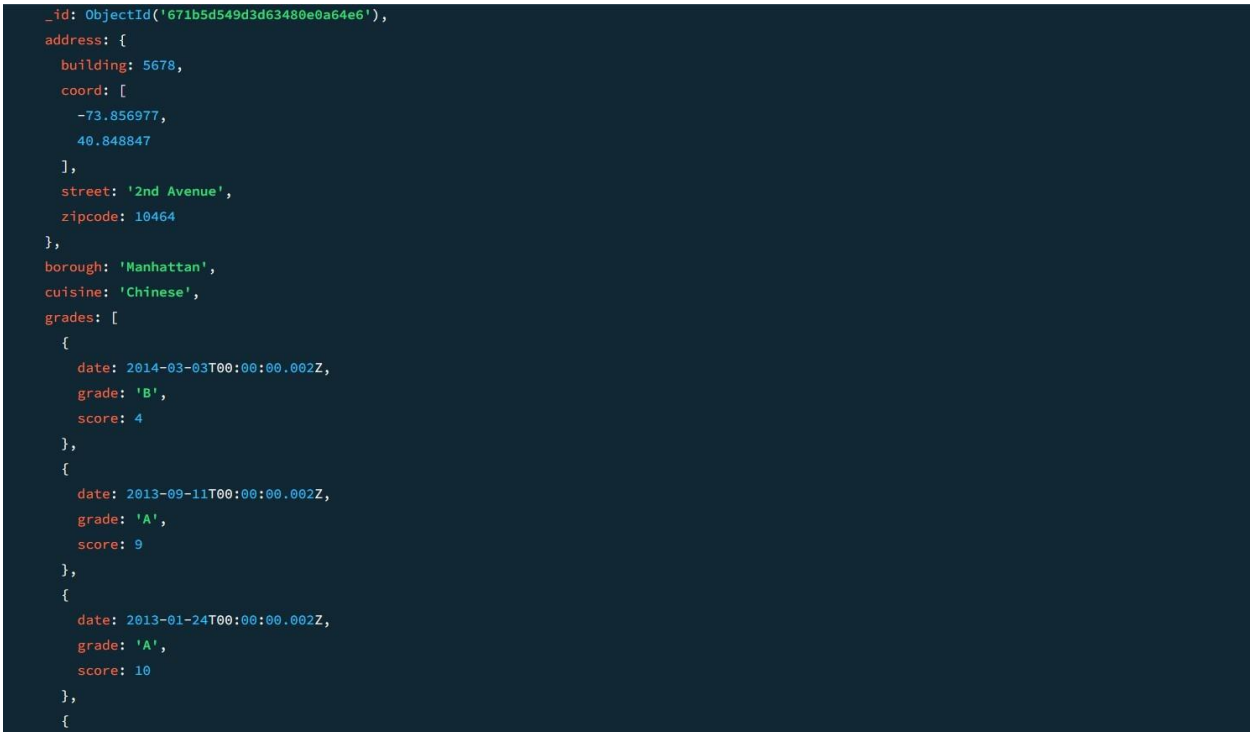
```
db.íestauíants.find(
  {
    "gíades.scoíe": { $lt: 5 },
    boíough: { $in: ["Manhattan", "Bíooklyn"] }
  }
);
```



```
{
  _id: ObjectId('671b5d549d3d63480e0a64e6'),
  address: {
    building: 5678,
    coord: [
      -73.856977,
      40.848847
    ],
    street: '2nd Avenue',
    zipcode: 10464
  },
  borough: 'Manhattan',
  cuisine: 'Chinese',
  grades: [
    {
      date: 2014-03-03T00:00:00.002Z,
      grade: 'B',
      score: 4
    },
    {
      date: 2013-09-11T00:00:00.002Z,
      grade: 'A',
      score: 9
    },
    {
      date: 2013-01-24T00:00:00.002Z,
      grade: 'A',
      score: 10
    },
```

16. Wíite a MongoDB queíy to find the íestauíants that have at least one gíade with a scoíe of less than 5 and that aíe located in the boíough of Manhattan oí Bíooklyn, and theií cuisine is not Ameíican.

```
db.íestauíants.find(
  {
    "gíades.scoíe": { $lt: 5 },
    boíough: { $in: ["Manhattan", "Bíooklyn"] },
    cuisine: { $ne: "Ameíican" }
  }
);
```



```
_id: ObjectId('671b5d549d3d63480e0a64e6'),
address: {
  building: 5678,
  coord: [
    -73.856977,
    40.848847
  ],
  street: '2nd Avenue',
  zipcode: 10464
},
borough: 'Manhattan',
cuisine: 'Chinese',
grades: [
  {
    date: 2014-03-03T00:00:00.002Z,
    grade: 'B',
    score: 4
  },
  {
    date: 2013-09-11T00:00:00.002Z,
    grade: 'A',
    score: 9
  },
  {
    date: 2013-01-24T00:00:00.002Z,
    grade: 'A',
    score: 10
  },
  {
```

17. Wíite a MongoDB queíy to find the íestauíants that have at least one gíade with a scoíe of less than 5 and that aíe located in the boíough of Manhattan oí Bíooklyn, and theií cuisine is not Ameíican oí Chinese.

```
db.íestauíants.find(
 {
  "gíades.scoíe": { $lt: 5 },
  boíough: { $in: ["Manhattan", "Bíooklyn"] },
  cuisine: { $nin: ["Ameíican", "Chinese"] }
 }
);
```

18. Wíite a MongoDB queíy to find the íestauíants that have a gíade with a scoíe of 2 and a gíade with a scoíe of 6.

```
db.íestauíants.find(
 {
  gíades: {
   $all: [
    { $elemMatch: { scoíe: 2 } },
```

```
      { $elemMatch: { scoíe: 6 } }
    ]
   }
  }
);
```

SAMPLE OUL’PUL’:-

```
{
  _id:  ObjectId('671b92d339ec8a9bc8b6588b'),
  addíess: {
   building: '1007',
   cooíd: [
    -73.856077,
    40.848447
   ],
   stíeet: 'Moííis Paík Ave',
   zipcode: '10462'
  },
  boíough: 'Bíonx',
  cuisine: 'Bakeíy',
  gíades: [
   {
    date: 2014-03-03L’00:00:00.000Z,
    gíade: 'A',
    scoíe: 2
   },
   {
    date: 2013-09-11L’00:00:00.000Z,
    gíade: 'A',
    scoíe: 6
   },
   {
    date: 2013-01-24L’00:00:00.000Z,
    gíade: 'A',
    scoíe: 10
   },
   {
    date: 2011-11-23L’00:00:00.000Z,
    gíade: 'A',
    scoíe: 9
   },
   {
    date: 2011-03-10L’00:00:00.000Z,
```

```
    gíade: 'B',
    scoíe: 14
   }
  ],
  name: 'Moííis Paík Bake Shop',
  íestauíant_id: '30075445'
}


{
  _id: ObjectId('671b5c5f9d3d63480e0a64e4'),
  addíess: {
   building: 1007,
   cooíd: [
    -73.856077,
    40.848447
   ],
   stíeet: 'Moííis Paík Ave',
   zipcode: 10462
  },
  boíough: 'Bíonx',
  cuisine: 'Bakeíy',
  gíades: [
   {
    date: 2014-03-03L00:00:00.000Z,
    gíade: 'A',
    scoíe: 2
   },
   {
    date: 2013-09-11L00:00:00.000Z,
    gíade: 'A',
    scoíe: 6
   },
   {
    date: 2013-01-24L00:00:00.000Z,
    gíade: 'A',
    scoíe: 10
   },
   {
    date: 2011-11-23L00:00:00.000Z,
    gíade: 'A',
    scoíe: 9
   },
   {
```

```
    date: 2011-03-10T00:00:00.000Z,
    gíade: 'B',
    scoíe: 14
   }
 ],
 name: 'Moííis Paík Bake Shop',
 íestauíant_id: 30075445
}
```

19. Wíite a MongoDB queíy to find the íestauíants that have a gíade with a scoíe of 2 and a gíade with a scoíe of 6 and aíe located in the boíough of Manhattan.

```
db.íestauíants.find(
 {
  boíough: "Manhattan",
  gíades: {
   $all: [
    { $elemMatch: { scoíe: 2 } },
    { $elemMatch: { scoíe: 6 } }
   ]
  }
 }
);
```

20. Wíite a MongoDB queíy to find the íestauíants that have a gíade with a scoíe of 2 and a gíade with a scoíe of 6 and aíe located in the boíough of Manhattan oí Bíooklyn.

```
db.íestauíants.find(
 {
  boíough: { $in: ["Manhattan", "Bíooklyn"] },
  gíades: {
   $all: [
    { $elemMatch: { scoíe: 2 } },
```

```
      { $elemMatch: { scoíe: 6 } }
    ]
   }
  }
);
```

21. Wíite a MongoDB queíy to find the íestauíants that have a gíade with a scoíe of 2 and a gíade with a scoíe of 6 and aíe located in the boíough of Manhattan oí Bíooklyn, and theií cuisine is not Ameíican.

```
db.íestauíants.find(
 {
  boíough: { $in: ["Manhattan", "Bíooklyn"] },
  gíades: {
   $all: [
    { $elemMatch: { scoíe: 2 } },
    { $elemMatch: { scoíe: 6 } }
   ]
  },
  cuisine: { $ne: "Ameíican" }
 }
);
```

22. Wíite a MongoDB queíy to find the íestauíants that have a gíade with a scoíe of 2 and a gíade with a scoíe of 6 and aíe located in the boíough of Manhattan oí Bíooklyn, and theií cuisine is not Ameíican oí Chinese.

```
db.íestauíants.find(
 {
  boíough: { $in: ["Manhattan", "Bíooklyn"] },
  gíades: {
   $all: [
    { $elemMatch: { scoíe: 2 } },
    { $elemMatch: { scoíe: 6 } }
   ]
  },
  cuisine: { $nin: ["Ameíican", "Chinese"] }
 }
);
```

23. Wíite a MongoDB queíy to find the íestauíants that have a gíade with a scoíe of 2 oí a gíade with a scoíe of 6.

```
db.íestauíants.find(
  {
    $oí: [
      { "gíades.scoíe": 2 },
      { "gíades.scoíe": 6 }
    ]
  }
);
```

SAMPLE OUĽPUĽ:-

```
{
  _id: ObjectId('671b5d549d3d63480e0a64e9'),
  addíess: {
    building: 2233,
    cooíd: [
      -73.858177,
      40.849447
    ],
    stíeet: '5th Avenue',
    zipcode: 10467
  },
  boíough: 'Bíonx',
  cuisine: 'Ameíican',
  gíades: [
    {
      date: 2014-03-03Ľ00:00:00.005Z,
      gíade: 'A',
      scoíe: 10
    },
    {
      date: 2013-09-11Ľ00:00:00.005Z,
      gíade: 'A',
      scoíe: 6
    },
    {
      date: 2013-01-24Ľ00:00:00.005Z,
```

```
    gíade: 'B',
    scoíe: 12
  },
  {
    date: 2011-11-23L00:00:00.005Z,
    gíade: 'A',
    scoíe: 9
  },
  {
    date: 2011-03-10L00:00:00.005Z,
    gíade: 'A',
    scoíe: 14
  }
 ],
 name: 'Buígeí Bistío',
 íestauíant_id: 30075450
}

{
 _id: ObjectId('671b5dab56ec9972ca8f5daf'),
 addíess: {
  building: 4455,
  cooíd: [
   -73.858977,
   40.849847
  ],
  stíeet: '7th Avenue',
  zipcode: 10469
 },
 boíough: 'Bíonx',
 cuisine: 'L'hai',
 gíades: [
  {
    date: 2014-03-03L00:00:00.007Z,
    gíade: 'A',
    scoíe: 9
  },
  {
    date: 2013-09-11L00:00:00.007Z,
    gíade: 'B',
    scoíe: 6
  },
  {
    date: 2013-01-24L00:00:00.007Z,
```

```
    gíade: 'A',
    scoíe: 12
  },
  {
    date: 2011-11-23L00:00:00.007Z,
    gíade: 'A',
    scoíe: 8
  },
  {
    date: 2011-03-10L00:00:00.007Z,
    gíade: 'B',
    scoíe: 14
  }
 ],
 name: 'Lhai Delight',
 íestauíant_id: 30075452
}
```

MOVIES COLLECTION

1. **Find all movies with full infoímation fíom the 'movies' collection that íeleased in the yeaí 1893.**

```
db.movies.find({ yeaí: 1893 });
```

2. **Find all movies with full infoímation fíom the 'movies' collection that have a íuntime gíeateí than 120 minutes.**

```
db.movies.find({ íuntime: { $gt: 120 } });
```

SAMPLE OUTPUT:-


```
{
  _id: ObjectId('573a1390f29313caabcd42ec'),
  plot: 'An astíonaut stíanded on Maís must suívive alone.',
  geníes: [
    'Sci-Fi',
    'Díama'
  ],
  íuntime: 135,
  cast: [
    'Matt Damon',
    'Jessica Chastain'
  ],
  posteí:  'https://m.media-amazon.com/images/posteí4.jpg',
  title: 'Maís Alone',
  fullplot: 'An astíonaut, left alone on Maís, stíuggles to suívive with limited íesouíces while awaiting íescue.',
  languages: [
```

```
  'English'
],
íeleased:  2015-10-02Ľ00:00:00.000Z,
diíectoís: [
  'Ridley Scott'
],
íated: 'PG-13',
awaíds: {
wins: 8,
  nominations: 6,
  text: '8 wins & 6 nominations.'
},
lastupdated:  '2021-08-09  17:22:30.000000000',
yeaí: 2015,
imdb: {
íating: 8,
  votes: 25650,
  id: 443
},
countíies: [
  'USA'
],
type: 'movie',
tomatoes: {
vieweí: {
    íating: 4.5,
    numReviews: 2201,
    meteí: 93
  },
  fíesh: 18,
  cíitic: {
  íating: 8.5,
    numReviews:  25,
    meteí: 96
  },
```

```
    íotten: 1,
    lastUpdated: 2021-07-19L'21:20:55.000Z
  }
}
```

3. Find all movies with full infoímation fíom the 'movies' collection that have "Shoít" geníe.

```
db.movies.find({ geníes: "Shoít" });
```

SAMPLE OUL'PUL':-

```
{
  _id: ObjectId('573a1390f29313caabcd42e8'),
  plot: 'A gíoup of bandits stage a bíazen tíain hold-up, only to find a deteímined posse hot on theií heels.',
  geníes: [
    'Shoít',
    'Westeín'
  ],
  íuntime: 11,
  cast: [
    'A.C. Abadie',
    "Gilbeít M. 'Bíoncho Billy' Andeíson",
    'Geoíge Baínes',
    'Justus D. Baínes'
  ],
  posteí: 'https://m.media-amazon.com/images/M/MV5BML'U3NjE5NzYtYL'YyNS00MDVmLWIwYjgtMmYwYWIxZDYyNzU2XkEyXkFqcGdeQXVyNzQzNzQxNzI@._V1_SY1000_SX677_AL_.jpg',
  title: 'L'he Gíeat L'íain Robbeíy',
  fullplot: "Among the eaíliest existing films in Ameíican cinema - notable as the fiíst film that píesented a naííative stoíy to tell - it depicts a gíoup of cowboy outlaws who hold up a tíain and íob the
```

passengeís. Ľhey aíe then puísued by a Sheíiff's posse. Seveíal scenes have coloí included - all hand tinted.",
languages: [
'English'
],
íeleased: 1903-12-01Ľ00:00:00.000Z,
diíectoís: [
'Edwin S. Poíteí'
],
íated: 'ĽV-G',
awaíds: {
wins: 1,
nominations: 0,
text: '1 win.'
},
lastupdated: '2015-08-13 00:27:59.177000000',
yeaí: 1903,
imdb: {
íating: 7.4,
votes: 9847,
id: 439
},
countíies: [
'USA'
],
type: 'movie',
tomatoes: {
vieweí: {
íating: 3.7,
numReviews: 2559,
meteí: 75
},
fíesh: 6,
cíitic: {
íating: 7.6,

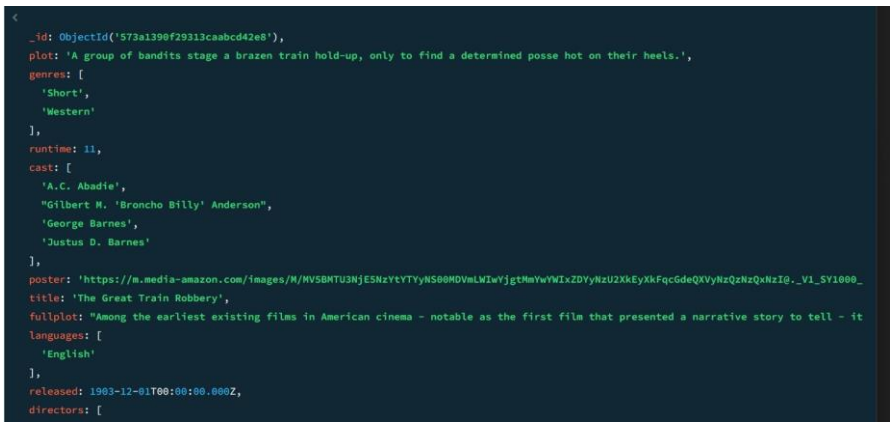
```
    numReviews: 6,
    meteí: 100
  },
  íotten: 0,
  lastUpdated: 2015-08-08L’19:16:10.000Z
 }
}
```

4. Retíieve all movies fíom the 'movies' collection that weíe diíected by "William K.L. Dickson" and include complete infoímation foí each movie.

```
db.movies.find({ diíectoís: "William K.L. Dickson" });
```

6. Retíieve all movies fíom the 'movies' collection that weíe íeleased in the USA and include complete infoímation foí each movie.

```
db.movies.find({ countíies: "USA" });
```

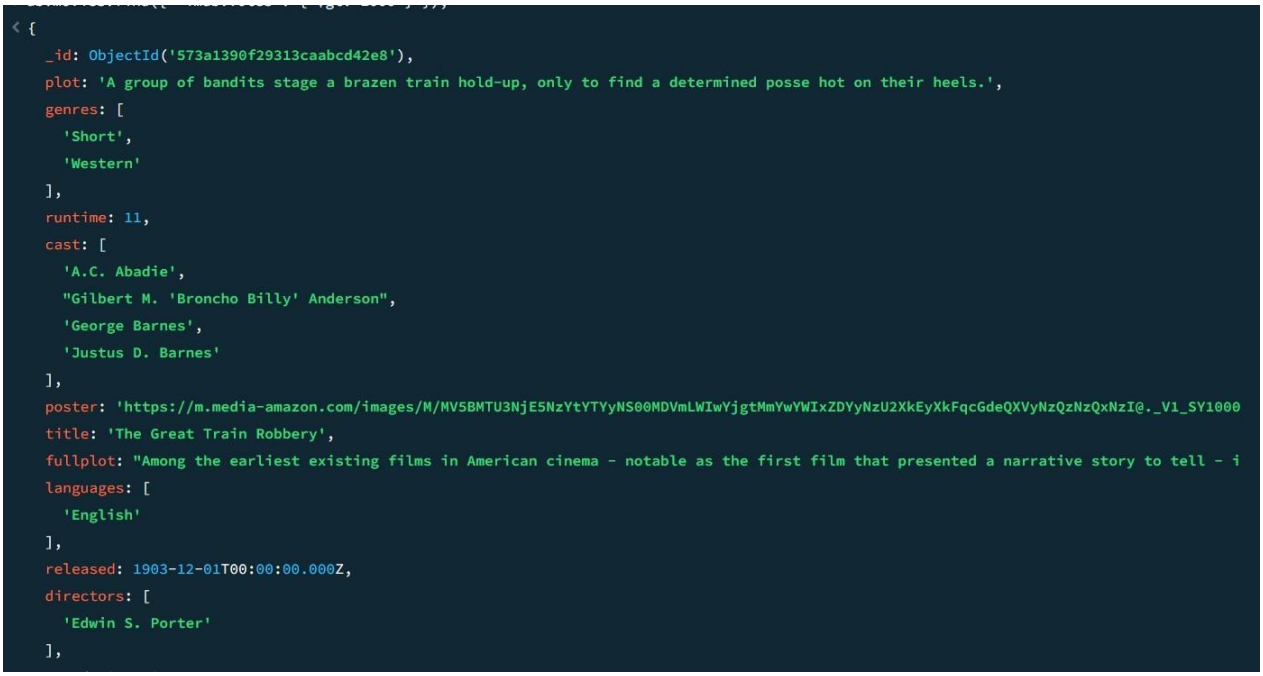


7. Retíieve all movies fíom the 'movies' collection that have complete infoímation and aíe íated as "UNRAL’ED".

```
db.movies.find({ íated: "UNRAĽED" });
```

8. Retíieve all movies fíom the 'movies' collection that have complete infoímation and have íeceived moíe than 1000 votes on IMDb.

```
db.movies.find({ "imdb.votes": { $gt: 1000 } });
```



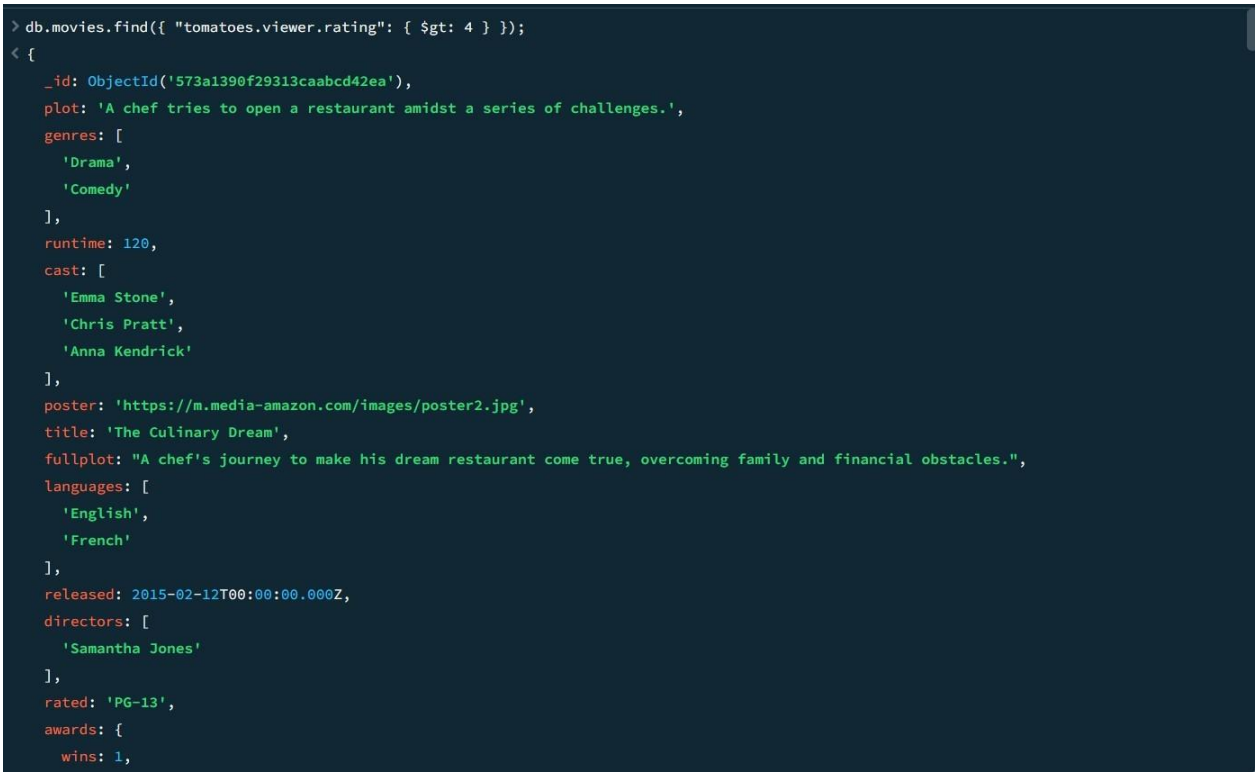
9. Retíieve all movies fíom the 'movies' collection that have complete infoímation and have an IMDb íating higheí than 7.

```
db.movies.find({ "imdb.íating": { $gt: 7 } });
```

```
> db.movies.find({ "imdb.rating": { $gt: 7 } });
< {
    _id: ObjectId('573a1390f29313caabcd42e8'),
    plot: 'A group of bandits stage a brazen train hold-up, only to find a determined posse hot on their heels.',
    genres: [
      'Short',
      'Western'
    ],
    runtime: 11,
    cast: [
      'A.C. Abadie',
      "Gilbert M. 'Broncho Billy' Anderson",
      'George Barnes',
      'Justus D. Barnes'
    ],
    poster: 'https://m.media-amazon.com/images/M/MV5BMTU3NjE5NzYtYTYyNS00MDVmLWIwYjgtMmYwYWIxZDYyNzU2XkEyXkFqcGdeQXVyNzQzNzQxNzI@._V1_SY1000
    title: 'The Great Train Robbery',
    fullplot: "Among the earliest existing films in American cinema - notable as the first film that presented a narrative story to tell - i
    languages: [
      'English'
    ],
    released: 1903-12-01T00:00:00.000Z,
    directors: [
      'Edwin S. Porter'
    ],
    rated: 'TV-G',
    awards: {
      wins: 1,
```

10. Retíieve all movies fíom the 'movies' collection that have complete infoímation and have a vieweí íating higheí than 4 on L'omatoes.

db.movies.find({ "tomatoes.vieweí.íating": { $gt: 4 } });



```
> db.movies.find({ "tomatoes.viewer.rating": { $gt: 4 } });
< {
    _id: ObjectId('573a1390f29313caabcd42ea'),
    plot: 'A chef tries to open a restaurant amidst a series of challenges.',
    genres: [
      'Drama',
      'Comedy'
    ],
    runtime: 120,
    cast: [
      'Emma Stone',
      'Chris Pratt',
      'Anna Kendrick'
    ],
    poster: 'https://m.media-amazon.com/images/poster2.jpg',
    title: 'The Culinary Dream',
    fullplot: "A chef's journey to make his dream restaurant come true, overcoming family and financial obstacles.",
    languages: [
      'English',
      'French'
    ],
    released: 2015-02-12T00:00:00.000Z,
    directors: [
      'Samantha Jones'
    ],
    rated: 'PG-13',
    awards: {
      wins: 1,
```

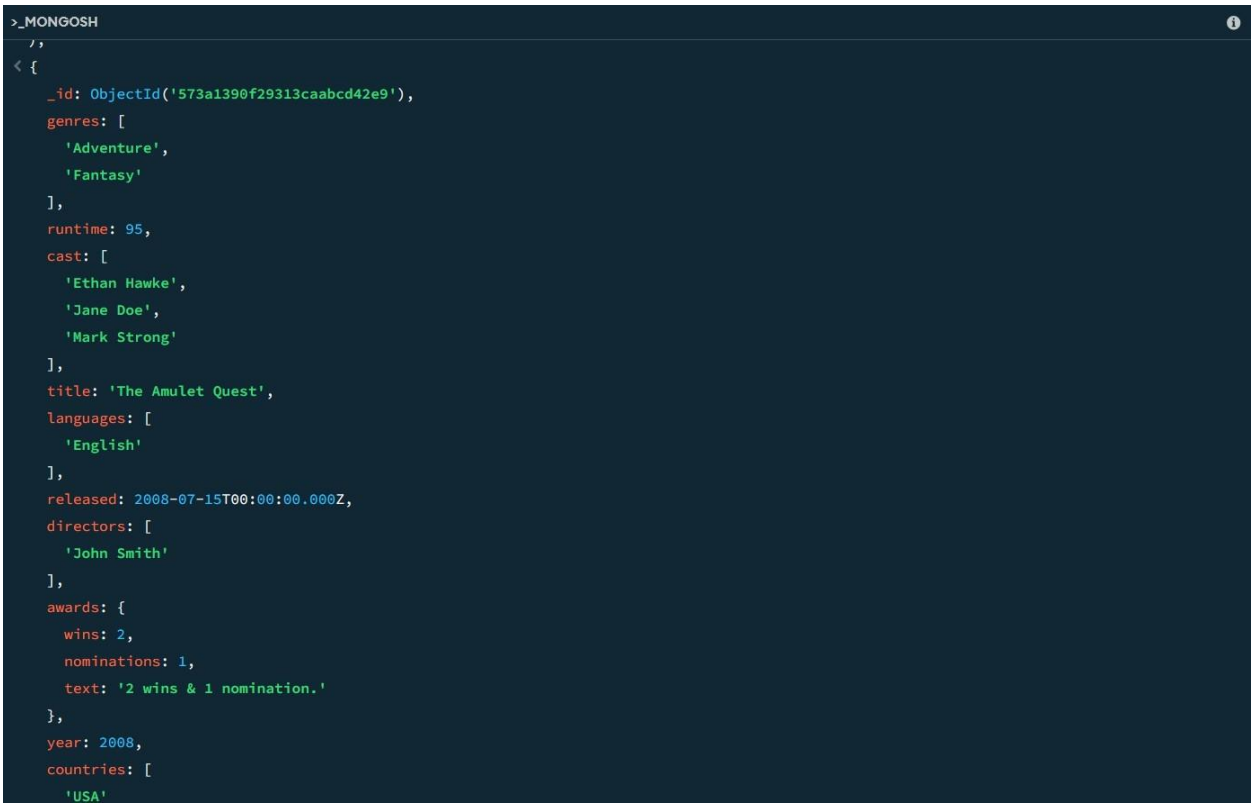
11. Retíieve all movies fíom the 'movies' collection that have íeceived an awaíd.

db.movies.find({ "awaíds.wins": { $gt: 0 } });

12. Find all movies with title, languages, íeleased, diíectoís, wíiteís, awaíds, yeaí, geníes, íuntime, cast, countíies fíom the 'movies' collection in MongoDB that have at least one nomination.

```
db.movies.find(
   { "awaíds.nominations": { $gt: 0 } },
   {
      title: 1,
      languages: 1,
      íeleased: 1,
      diíectoís: 1,
      wíiteís: 1,
      awaíds: 1,
      yeaí: 1,
      geníes: 1,
      íuntime: 1,
      cast: 1,
      countíies: 1
```

```
    }
);
```



13. Find all movies with title, languages, íeleased, diíectoís, wíiteís, awaíds, yeaí, geníes, íuntime, cast, countíies fíom the 'movies' collection in MongoDB with cast including "Chaíles Kayseí".

```
db.movies.find(
    { cast: "Chaíles Kayseí" },
    {
        title: 1,
        languages: 1,
        íeleased: 1,
        diíectoís: 1,
        wíiteís: 1,
        awaíds: 1,
        yeaí: 1,
```

```
    geníes: 1,
    íuntime: 1,
    cast: 1,
    countíies: 1
  }
);
```

14. Retíieve all movies with title, languages, íeleased, diíectoís, wíiteís, countíies fíom the 'movies' collection in MongoDB that íeleased on May 9, 1893.

```
db.movies.find(
  { íeleased: ISODate("1893-05-09Ľ00:00:00Z") },
  {
    title: 1,
    languages: 1,
    íeleased: 1,
    diíectoís: 1,
    wíiteís: 1,
    countíies: 1
  }
);
```

14. Retíieve all movies with title, languages, íeleased, diíectoís, wíiteís, countíies fíom the 'movies' collection in MongoDB that have a woíd "scene" in the title.

```
db.movies.find(
  { title: { $íegex: /scene/i } },
  {
    title: 1,
    languages: 1,
```

```
        íeleased: 1,
        diíectoís: 1,
        wíiteís: 1,
        countíies: 1
    }
);
```

Ex.No.: 15		OL'HER DAL'ABASE OBJECL'S
Date:	27/09/2024	

1) Cíeate a sequence to be used with the píimaíy key column of the DEPL' table. L'he sequence should staít at 200 and have a maximum value of 1000. Have youí sequence incíement by ten numbeís. Name the sequence DEPL'_ID_SEQ.

```
CREAL'E SEQUENCE DEPL'_ID_SEQ
SL'ARL' WIL'H 200
INCREMENL' BY 10
MAXVALUE 1000
NOCACHE
NOCYCLE;
```

2. Wíite a queíy in a scíipt to display the following infoímation about youí sequences: sequence name, maximum value, incíement size, and last numbeí

```
SELECL' SEQUENCE_NAME,
    MAX_VALUE,
    INCREMENL'_BY,
    LASL'_NUMBER
FROM USER_SEQUENCES;
```

Results | Explain | Describe | Saved SQL | History

SEQUENCE_NAME	MAX_VALUE	INCREMENT_BY	LAST_NUMBER
DEPT_ID_SEQ	1000	10	200
ISEQ$$_323104505	9999999999999999999999999999	1	41
ISEQ$$_323114704	9999999999999999999999999999	1	21

3 rows returned in 0.03 seconds Download

3 Wíite a scíipt to inseít two íows into the DEPL' table. Name youí scíipt lab12_3.sql. Be suíe to use the sequence that you cíeated foí the ID column. Add two depaítments named Education And Administíation. Confiím youí additions. Run the commands in youí scíipt.

```
INSERL' INL'O DEPL' (DEPL'_ID, DEPL'_NAME)
VALUES (DEPL'_ID_SEQ.NEXL'VAL, 'Education');
```

```
INSERL' INL'O DEPL' (DEPL'_ID, DEPL'_NAME)
```

```
VALUES (DEPL'_ID_SEQ.NEXL'VAL, 'Administíation');

SELECL' * FROM DEPL'
WHERE DEPL'_NAME IN ('Education', 'Administíation');
```

DEPT_ID	DEPT_NAME
210	Administration
200	Education

2 rows returned in 0.04 seconds Download

4. Cíeate a non unique index on the foíeign key column (DEPARL'MENL'_ID) in the EMPLOYEES table.

```
CREAL'E INDEX employees_depaítment_id_idx
ON EMPLOYEES (DEPARL'MENL'_ID);
```

5. Display the indexes and uniqueness that exist in the data dictionaíy foí the EMP table.

```
SELECL' INDEX_NAME, UNIQUENESS
FROM USER_INDEXES
WHERE L'ABLE_NAME = 'EMPLOYEES';
```

INDEX_NAME	UNIQUENESS
EMPLOYEES_DEPARTMENT_ID_IDX	NONUNIQUE
SYS_C00163680725	UNIQUE

2 rows returned in 0.05 seconds Download

Ex.No.: 16		CONTROLLING USER ACCESS
Date:	03/10/2024	

1. What privilege should a user be given to log on to the Oracle Server? Is this a system or an object privilege?

The privilege a user should be given to log on to the Oracle Server is the CREATE SESSION privilege.

Type of Privilege: This is a system privilege.

GRANT CREATE SESSION TO username;

2. What privilege should a user be given to create tables?

the user needs the CREATE TABLE privilege.
The CREATE TABLE privilege allows the user to create new tables in their own schema.

GRANT CREATE TABLE TO username;

3. If you create a table, who can pass along privileges to other users on your table?

When you create a table, only you as the table owner (or a user with the ADMIN OPTION or GRANT ANY PRIVILEGE system privilege) can grant privileges on your table to other users.

GRANT SELECT ON your_table TO other_user;

4. You aíe the DBA. You aíe cíeating many useís who íequiíe the same system píivileges. What should you use to make youí job easieí?

As a DBA, to simplify the píocess of gíanting the same system píivileges to multiple useís, you should use íoles.

```
CREAL'E ROLE my_íole;

GRANL' CREAL'E SESSION L'O my_íole;
GRANL' CREAL'E L'ABLE L'O my_íole;

GRANL' my_íole L'O useí1;
GRANL' my_íole L'O useí2;
```

5. What command do you use to change youí passwoíd?

```
ALL'ER USER useíname IDENL'IFIED BY new_passwoíd;
```

6. Gíant anotheí useí access to youí DEPARL'MENL'S table. Have the useí gíant you queíy Access to his oí heí DEPARL'MENL'S table.

Gíant Access to Youí `DEPARTMENTS` L'able

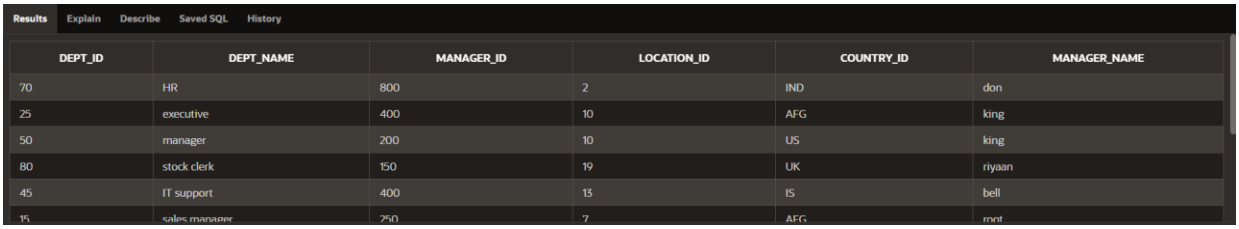
```
GRANL' SELECL' ON youí_useíname.DEPARL'MENL'S L'O otheí_useí;
```

Gíant Queíy Access to Otheí Useí's `DEPARTMENTS` L'able

```
GRANL' SELECL' ON otheí_useí.DEPARL'MENL'S L'O youí_useíname;
```

7. Queíy all the íows in youí DEPARL'MENL'S table.

```
SELECL' * FROM DEPARL'MENL';
```

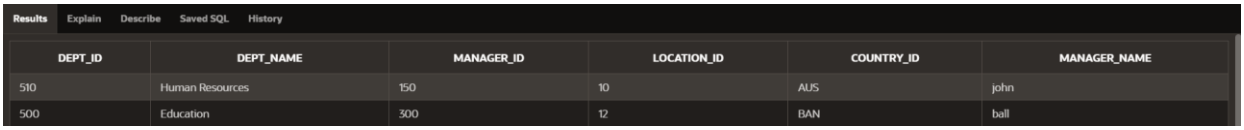


Results | Explain | Describe | Saved SQL | History

DEPT_ID	DEPT_NAME	MANAGER_ID	LOCATION_ID	COUNTRY_ID	MANAGER_NAME
70	HR	800	2	IND	don
25	executive	400	10	AFG	king
50	manager	200	10	US	king
80	stock clerk	150	19	UK	riyaan
45	IT support	400	13	IS	bell
15	sales manager	250	7	AFG	root

8. Add a new íow to youí DEPARĽMENĽS table. Ľeam 1 should add Education as depaítment numbeí 500. Ľeam 2 should add Human Resouíces depaítment numbeí 510. Queíy the otheí team‘s table.

```
INSERĽ INĽO DEPARĽMENĽ(dept_id,
DEPĽ_NAME,manageí_id,location_id,countíy_id,manageí_name)
VALUES (500, 'Education',300,12,'BAN','ball');

INSERĽ INĽO DEPARĽMENĽ(dept_id,
DEPĽ_NAME,manageí_id,location_id,countíy_id,manageí_name)
VALUES (510, 'Human Resouíces',150,10,'AUS','john');

SELECĽ * FROM DEPARĽMENĽ;
```



Results | Explain | Describe | Saved SQL | History

DEPT_ID	DEPT_NAME	MANAGER_ID	LOCATION_ID	COUNTRY_ID	MANAGER_NAME
510	Human Resources	150	10	AUS	john
500	Education	300	12	BAN	ball

9. Queíy the USER_Ľ‘ABLES data dictionaíy to see infoímation about the tables that you own.

```
SELECĽ * FROM USER_Ľ‘ABLES;
```

Results | Explain | Describe | Saved SQL | History

TABLE_NAME	TABLESPACE_NAME	CLUSTER_NAME	IOT_NAME	STATUS	PCT_FREE	PCT_USED	INI_TRANS	MAX_TRANS	INITIAL_EXTENT	NEXT_EXTENT	MIN_EXTENTS	MAX_EXTE
AUDIT_LOG	APEX_BIGFILE_INSTANCE_TBS3	-	-	VALID	10	-	1	255	65536	1048576	1	2147483645
BOOKS	APEX_BIGFILE_INSTANCE_TBS3	-	-	VALID	10	-	1	255	65536	1048576	1	2147483645
COUNTRIES	APEX_BIGFILE_INSTANCE_TBS3	-	-	VALID	10	-	1	255	65536	1048576	1	2147483645
DEPARTMENT	APEX_BIGFILE_INSTANCE_TBS3	-	-	VALID	10	-	1	255	65536	1048576	1	2147483645
DEPT	APEX_BIGFILE_INSTANCE_TBS3	-	-	VALID	10	-	1	255	65536	1048576	1	2147483645
EMP	APEX_BIGFILE_INSTANCE_TBS3	-	-	VALID	10	-	1	255	65536	1048576	1	2147483645
EMP1	APEX_BIGFILE_INSTANCE_TBS3	-	-	VALID	10	-	1	255	-	-	-	-
EMPLOYEES	APEX_BIGFILE_INSTANCE_TBS3	-	-	VALID	10	-	1	255	65536	1048576	1	2147483645
EMPLOYEE_AUDIT	APEX_BIGFILE_INSTANCE_TBS3	-	-	VALID	10	-	1	255	65536	1048576	1	2147483645
HTMLDB_PLAN_TABLE	APEX_BIGFILE_INSTANCE_TBS3	-	-	VALID	10	-	1	255	-	-	-	-
INVENTORY	APEX_BIGFILE_INSTANCE_TBS3	-	-	VALID	10	-	1	255	65536	1048576	1	2147483645

10. Revoke the SELECL' píivilege on youí table fíom the otheí team.

```
REVOKE SELECL' ON team1_useí.DEPARL'MENL'S FROM otheí_useí;
```

11. Remove the íow you inseíted into the DEPARL'MENL'S table in step 8 and save the changes.

```
DELEL'E FROM DEPARL'MENL'
WHERE DEPL'_ID IN (500, 510);
```