DBMS – PRACTISE 11\_1

Problem: −Create a list of all tables whose first two characters in the name of the table is JO −The tables must be owned by the current Oracle User.

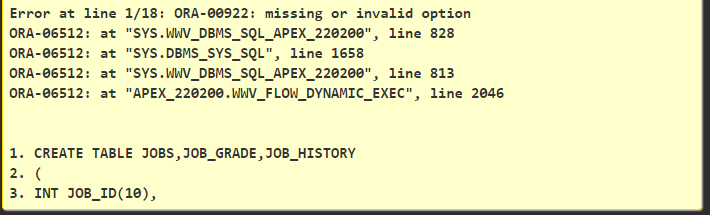
CREATE TABLE JOBS,JOB\_GRADE,JOB\_HISTORY

(

INT JOB\_ID(10),

NAME VARCHAR(100),

);



CREATE TABLE Emp

(

first\_name VARCHAR(100),

last\_name VARCHAR(100));

DESC Emp;

INSERT INTO Emp(first\_name,last\_name) VALUES ('vinay ','kumar');

INSERT INTO Emp(first\_name,last\_name) VALUES ('david','raju');

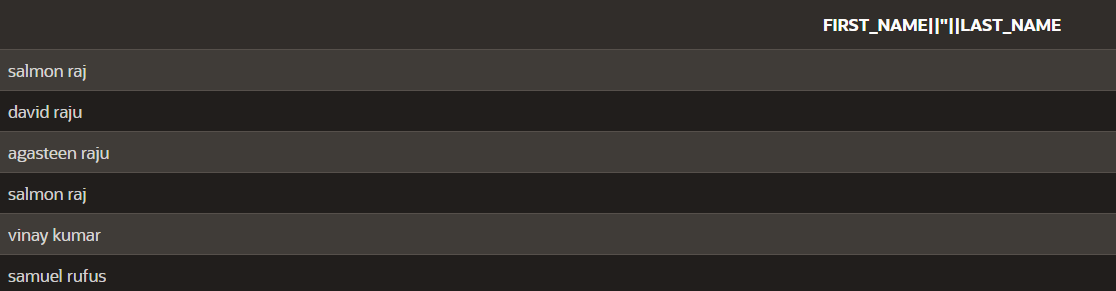
INSERT INTO Emp(first\_name,last\_name) VALUES ('samuel','rufus');

INSERT INTO Emp(first\_name,last\_name) VALUES ('agasteen','raju');

INSERT INTO Emp(first\_name,last\_name) VALUES ('salmon','raj');

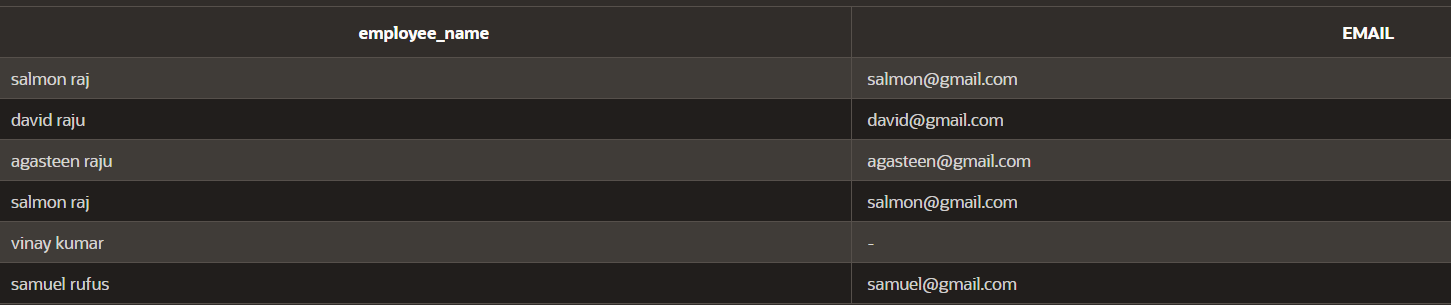
SELECT first\_name||' '||last\_name

FROM Emp;



SELECT first\_name ||' '||last\_name AS "employee\_name",Email

FROM Emp;



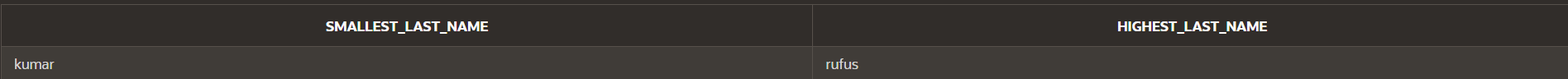
SELECT

MIN(last\_name) AS smallest\_last\_name,

MAX(last\_name) AS highest\_last\_name

FROM

employees;



UPDATE Emp

SET salary=700

WHERE first\_name='vinay';

UPDATE Emp

SET salary=100

WHERE first\_name='david';

UPDATE Emp

SET salary=1000

WHERE first\_name='samuel';

UPDATE Emp

SET salary=2000

WHERE first\_name='agasteen';

UPDATE Emp

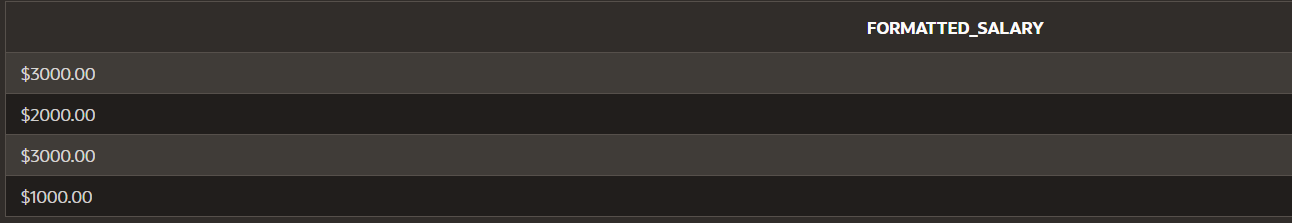
SET salary=3000

WHERE first\_name='salmon';

SELECT TO\_CHAR(salary, '$9999.99') AS formatted\_salary

FROM Emp

WHERE salary BETWEEN 700 AND 3000;

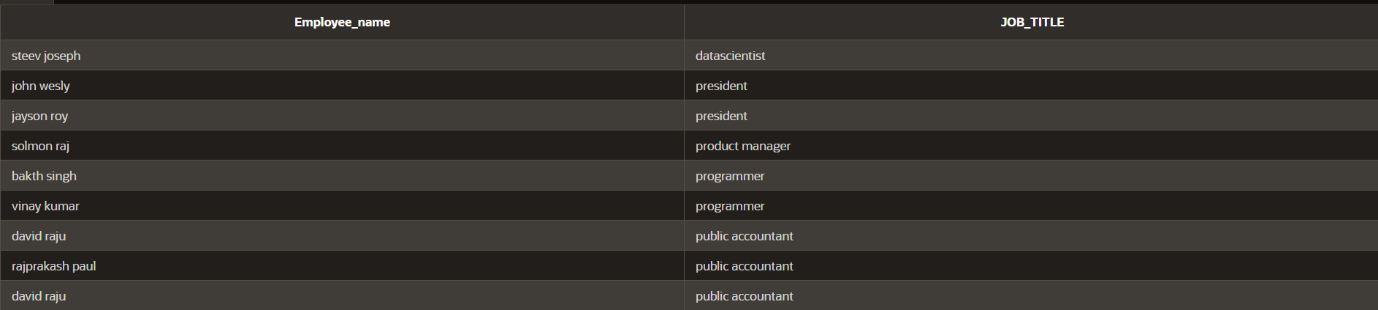


Problem: − Create a list of every employee and his related job title sorted by job\_title

SELECT first\_name||' '||last\_name AS "Employee\_name",job\_title

FROM Employ

ORDER BY job\_title;



SELECT job\_title,MIN(salary) || ' - ' || MAX(salary) AS salary\_range,salary AS employee\_salary

FROM Employ

GROUP BY job\_title, salary

ORDER BY job\_title, salary;



**CSA 0563 – DBMS**

**Cretaing a table by using CREATE**

CREATE TABLE STUDENT2

(

name VARCHAR(10),

std\_id NUMBER(10),

age NUMBER(2),

course VARCHAR(10),

date\_of\_registration DATE

);

Inserting values into table by Using INSERT keyword

INSERT INTO STUDENT2 (name,std\_id,age,course,date\_of\_registration)

VALUES ('ELIYAZAR',192311162,19,'DBMS',TO\_DATE('10-07-2024','DD-MM-YYYY'));

INSERT INTO STUDENT2 (name,std\_id,age,course,date\_of\_registration)

VALUES ('CHANIKYA',192311164,18,'DBMS',TO\_DATE('10-06-2024','DD-MM-YYYY'));

INSERT INTO STUDENT2 (name,std\_id,age,course,date\_of\_registration)

VALUES('CHANIKYA',192311164,18,'DBMS',TO\_DATE('10-06-2024','DD-MM-YYYY'));

INSERT INTO STUDENT2 (name,std\_id,age,course,date\_of\_registration)

VALUES('PRAKASH',192311165,19,'JAVA',TO\_DATE('10-06-2024','DD-MM-YYYY'));

INSERT INTO STUDENT2 (name,std\_id,age,course,date\_of\_registration)

VALUES('POLI',192311166,19,'JAVA',TO\_DATE('11-07-2024','DD-MM-YYYY'));

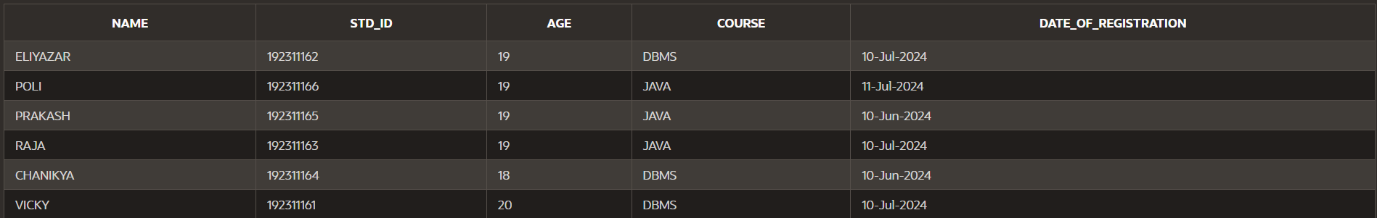
INSERT INTO STUDENT2 (name,std\_id,age,course,date\_of\_registration)

VALUES('VICKY',192311161,20,'DBMS',TO\_DATE('10-07-2024','DD-MM-YYYY'));

**Displaying the tablevalues by using** **SELECT**

SELECT \*

FROM STUDENT2;

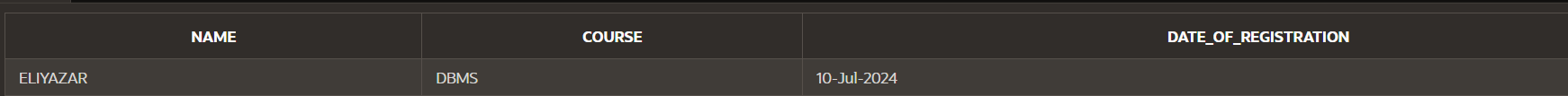


Appluying condition **WHERE**

SELECT name ,course,date\_of\_registration

FROM STUDENT2

WHERE std\_id = 192311162;



Adding another column to a table by using ALTER and ADD

ALTER TABLE STUDENT2

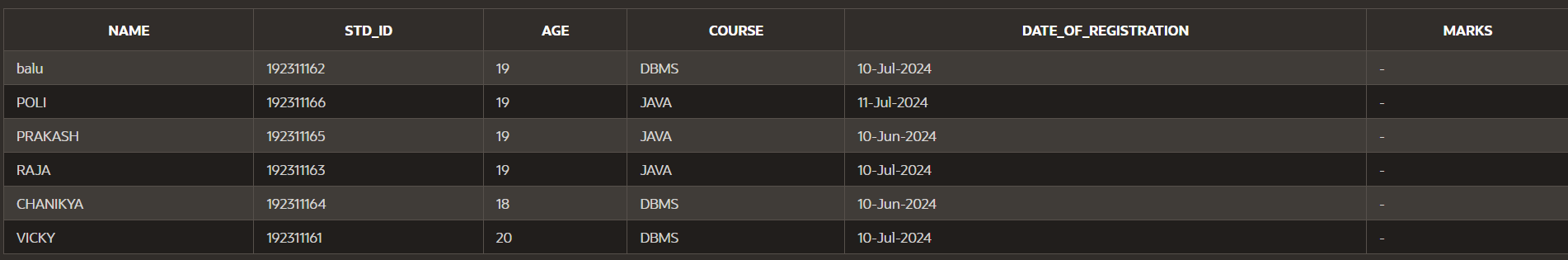
ADD marks FLOAT;



Deleting column name by using ALTER and DROP

ALTER TABLE STUDENT2

DROP (sign);



Updating the specified row in a table by using UPDATE,SET

UPDATE STUDENT2

SET name=’balu’

WHERE std\_id=192311162;



Deleting specifed row in a table by using DELETE keyword

DELETE FROM STUDENT2

WHERE name=’PRAKASH’

SELECT \* FROM STUDENT2



Deleting all the rows in a table by using TRUNCATE keyword

TRUNCATE TABLE STUDENT2;

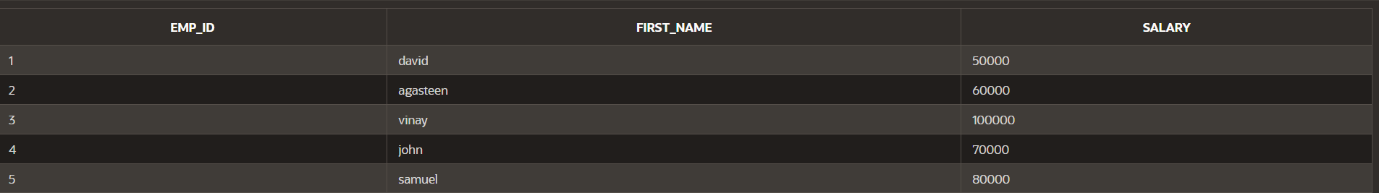
SELECT \* FROM STUDENT2;



Selecting only some columns in the table by using SELECT

SELECT emp\_id,first\_name,salary

FROM Employee;

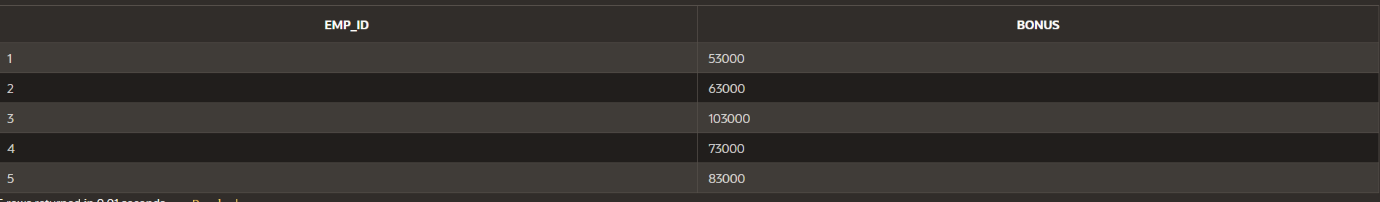


ARTHMETIC OPERATORS :

(+-\*/)

SELECT emp\_id, salary+3000

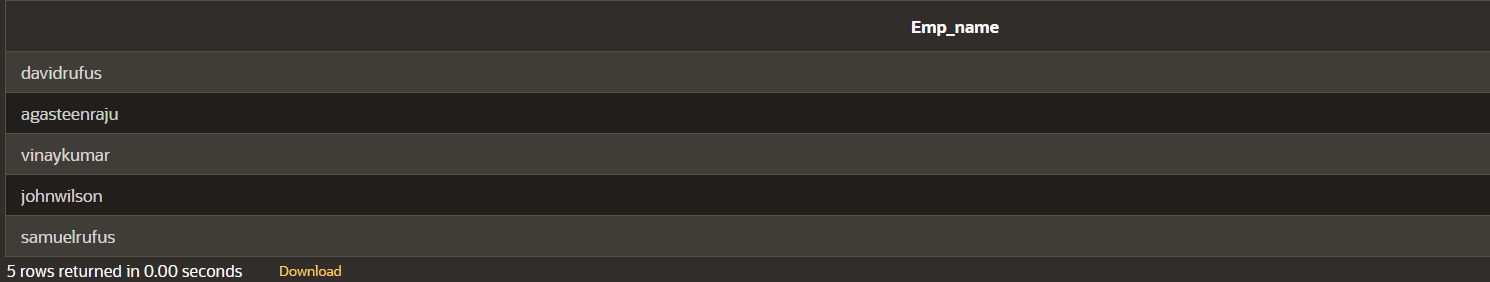
FROM Employee;



CONCATENATE two column name as one name by using AS :

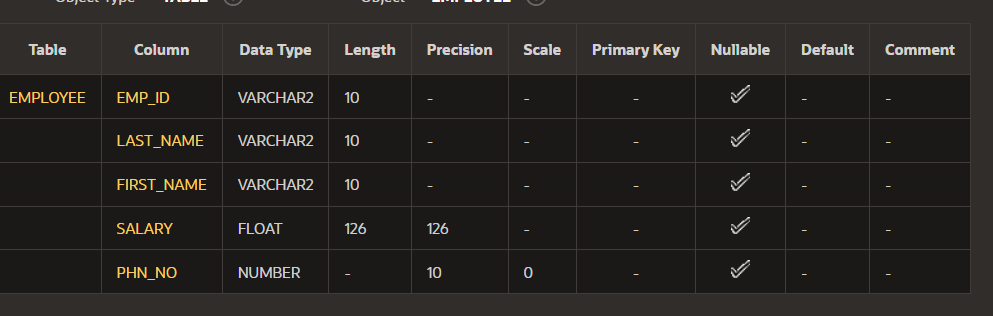
SELECT first\_name | | last\_name AS Emp\_name

FROM Employee;



DESCRIBE Keyword

desc keyword

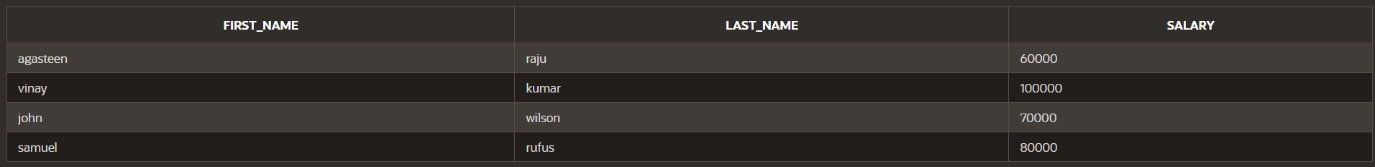


Using WHERE clause

SELECT first\_name,last\_name,salary

FROM Employee

WHERE salary > 50000.0;

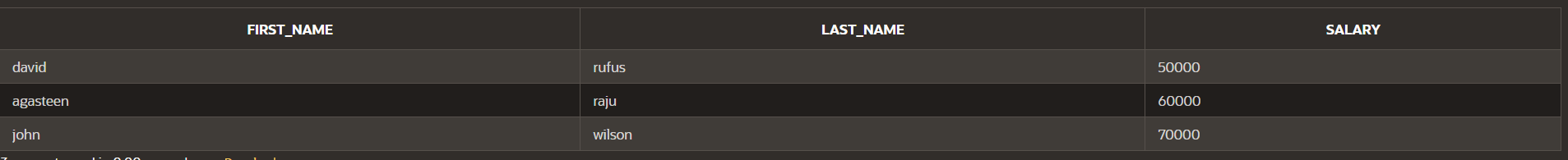


Using BETWEEN and AND operator

SELECT first\_name,last\_name ,salary

FROM Employee

WHERE salary BETWEEN 35000.0 AND 75000.0;

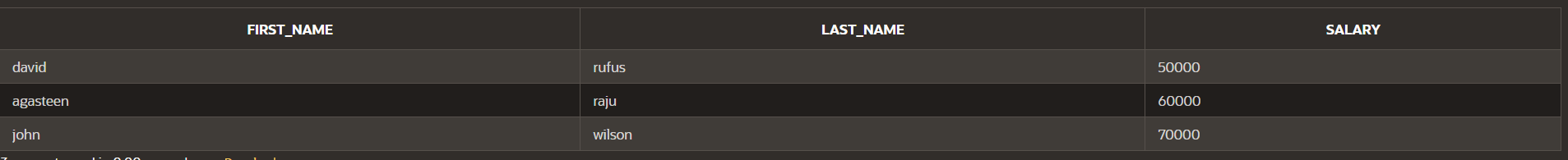


By using IN operator

SELECT first\_name,last\_name,salary

FROM Employee

WHERE emp\_id IN (2,5);

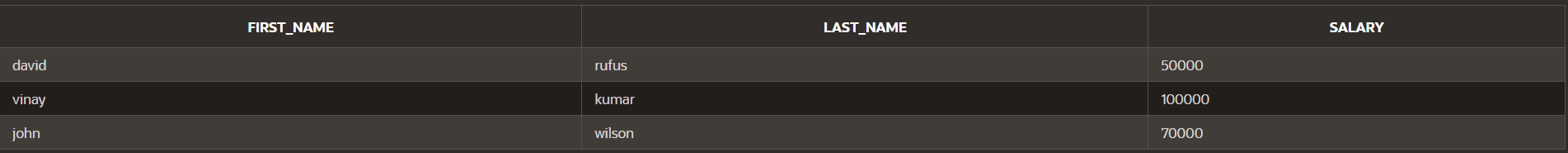


By using NOT IN operator

SELECT first\_name,last\_name,salary

FROM Employee

WHERE emp\_id NOT IN (2,5);

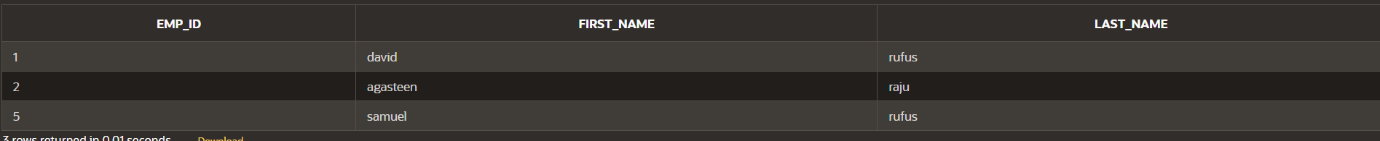


By using LIKE operator

SELECT first\_name,last\_name,salary

FROM Employee

WHERE last\_name LIKE ‘r%’;

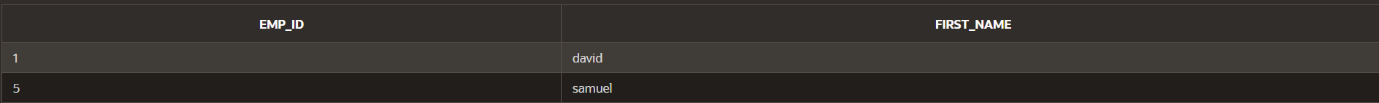


By using OR operator

SELECT emp\_id,first\_name

FROM Employee

WHERE emp\_id=’1’ OR emp\_id = ‘5’;

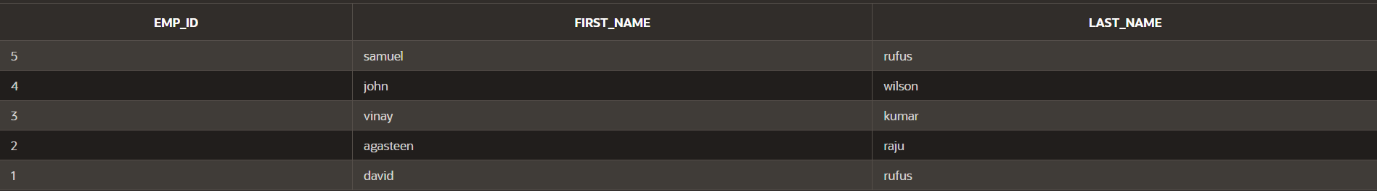


Arranging or ordering the tablein ASCENDING and DESCENDING ORDER by using ORDER BY

SELECT emp\_id,first\_name,last\_name

FROM Employee

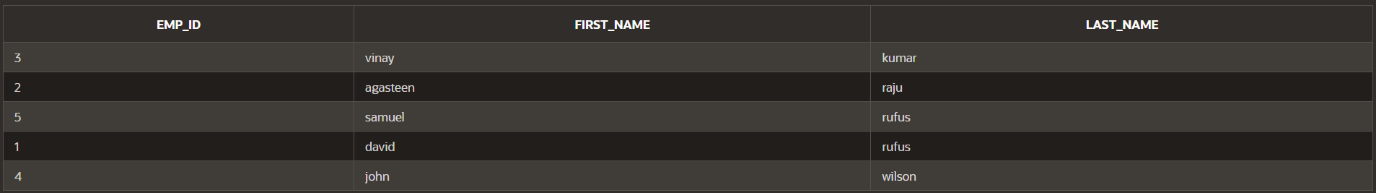
ORDER BY emp\_id DESC;



SELECT emp\_id,first\_name,last\_name

FROM employee

ORDER BY last\_name ;

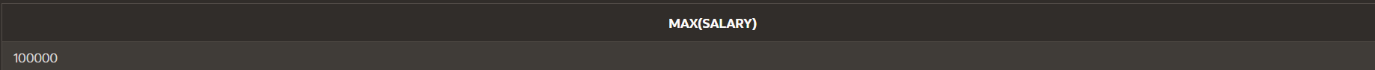


GROUP FUNCTIONS :

MAX :

SELECT MAX(salary)

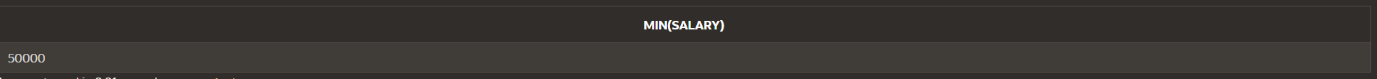
FROM Employee;



MIN:

SELECT MIN(salary)

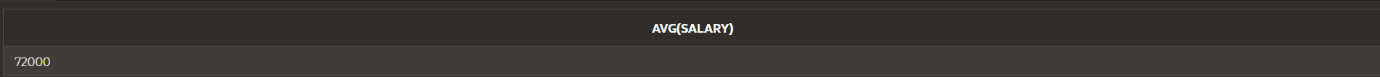
FROM Employee;



AVERAGE :

SELECT AVG(salary)

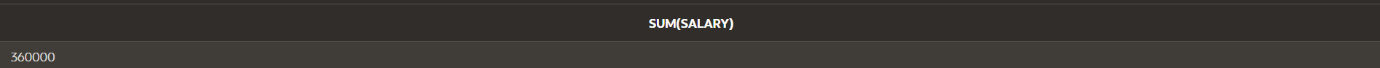
FROM Employee;



SUM :

SELECT SUM(salary)

FROM Employee;

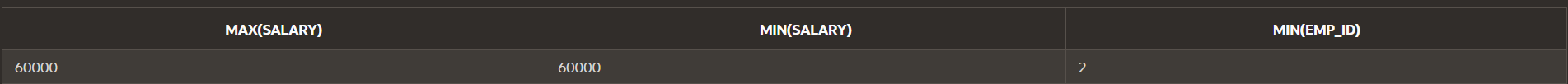


More than one GROUP function :

SELECT MAX(Salary),MIN(Salary),MIN(emp\_id)

FROM Employee

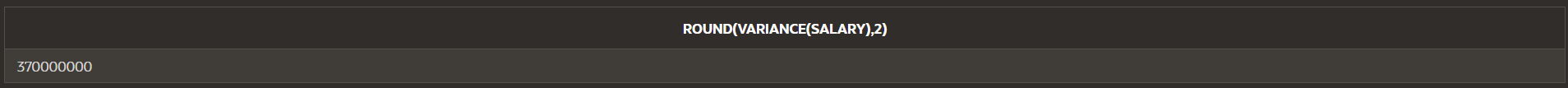
WHERE last\_name='raju';



VARIANCE :

SELECT ROUND(VARIANCE(salary),2)

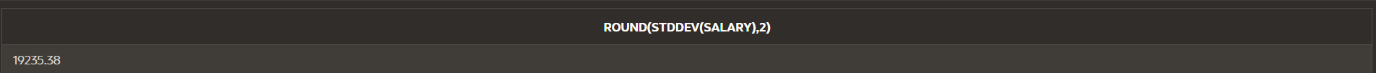
FROM Employee;



STANDARD DEVIATION :

SELECT ROUND(STDDEV(salary),2)

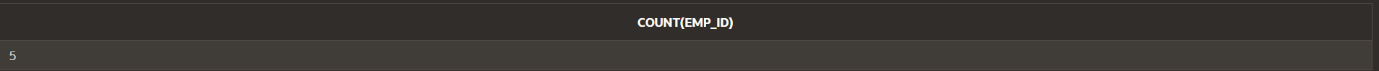
FROM Employee;



COUNT :

SELECT COUNT(emp\_id)

FROM Employee;

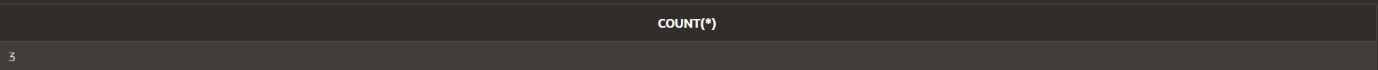


COUNT(\*)

SELECT COUNT(\*)

FROM Employee

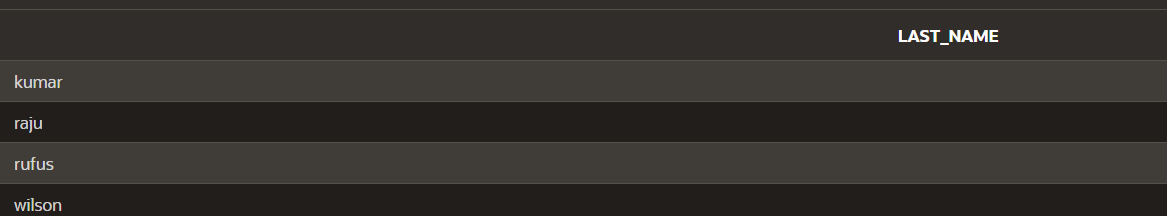
WHERE emp\_id > 2;



DISTINCT

SELECT DISTINCT last\_name

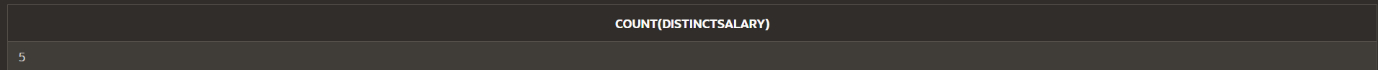
FROM Employee;





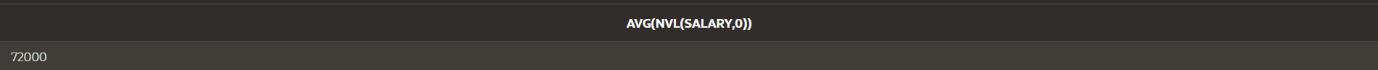
SELECT COUNT(DISTINCT salary)

FROM Employee;



SELECT AVG(NVL(salary,0))

FROM Employee;



**Section 9**

**DP 9.1:**

A screenshot of a computer

Description automatically generated

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

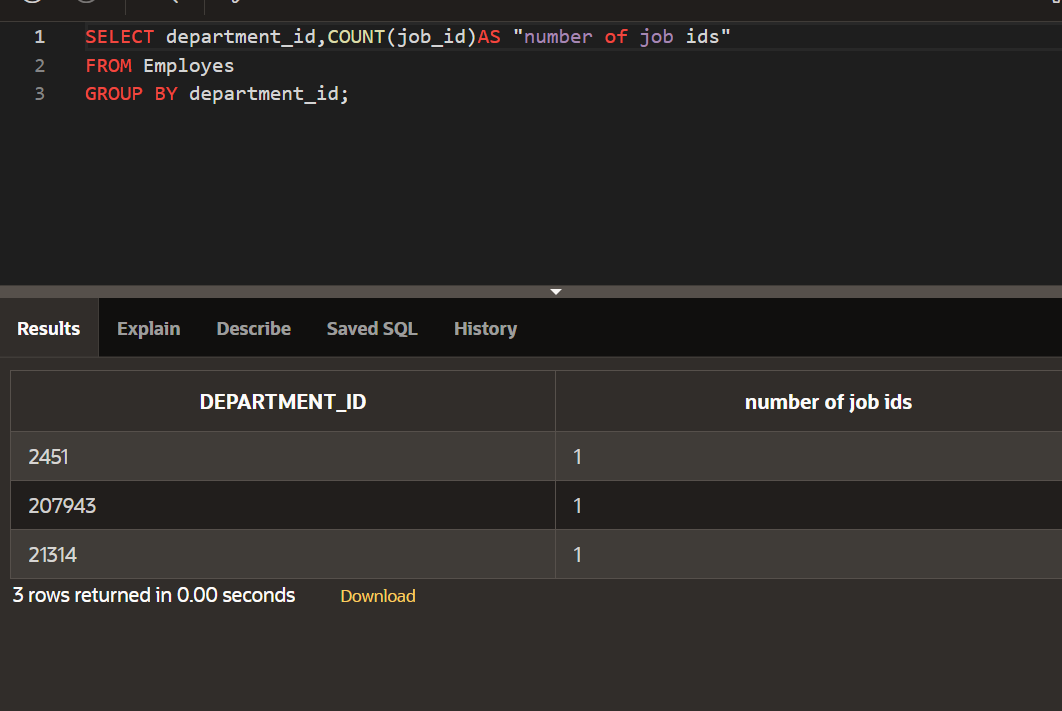
A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated



A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**DP 9.2:**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

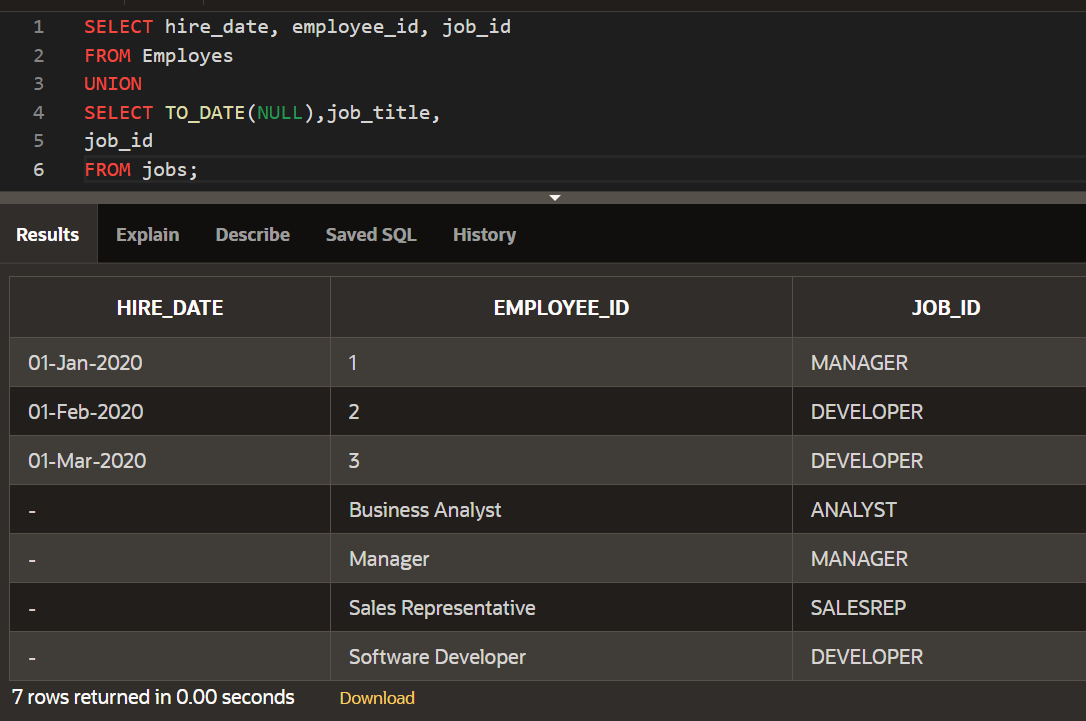
A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**DP 9.3:**

****

**A screenshot of a computer

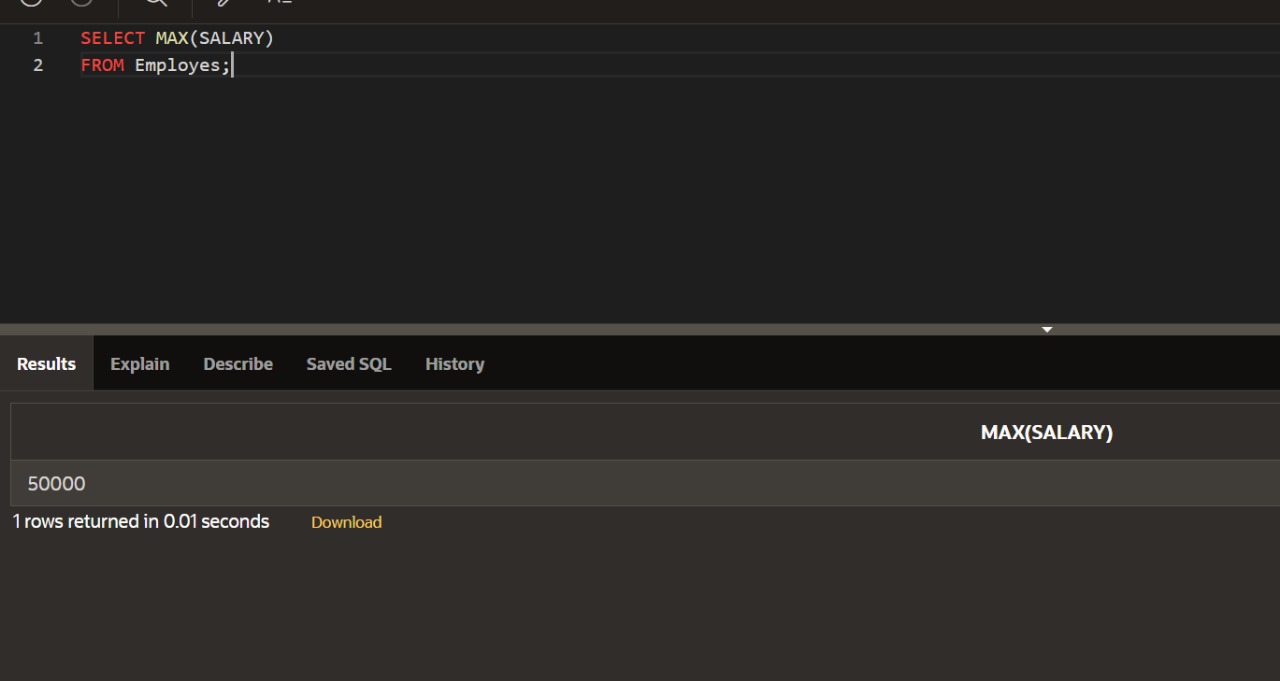
Description automatically generated**

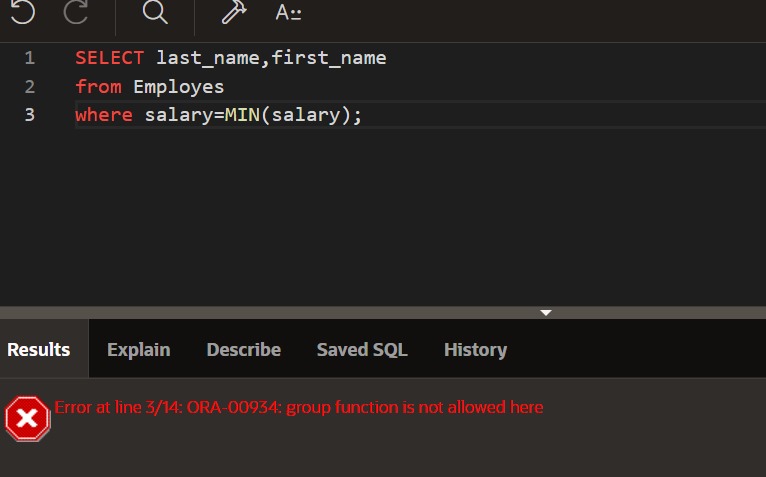
**A screenshot of a computer

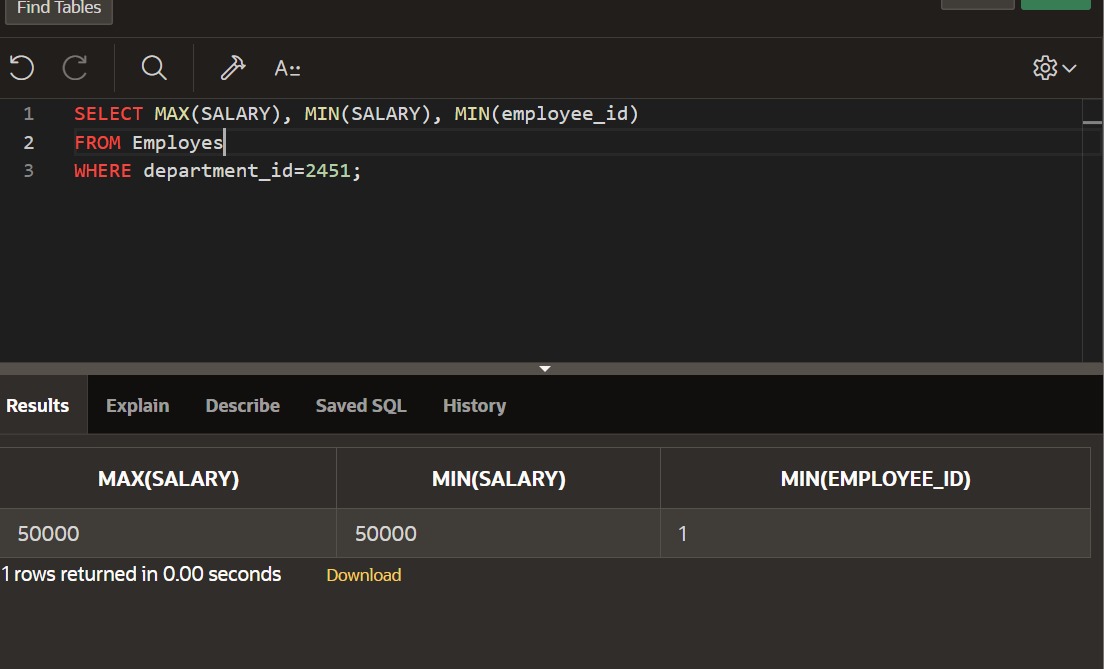
Description automatically generated**

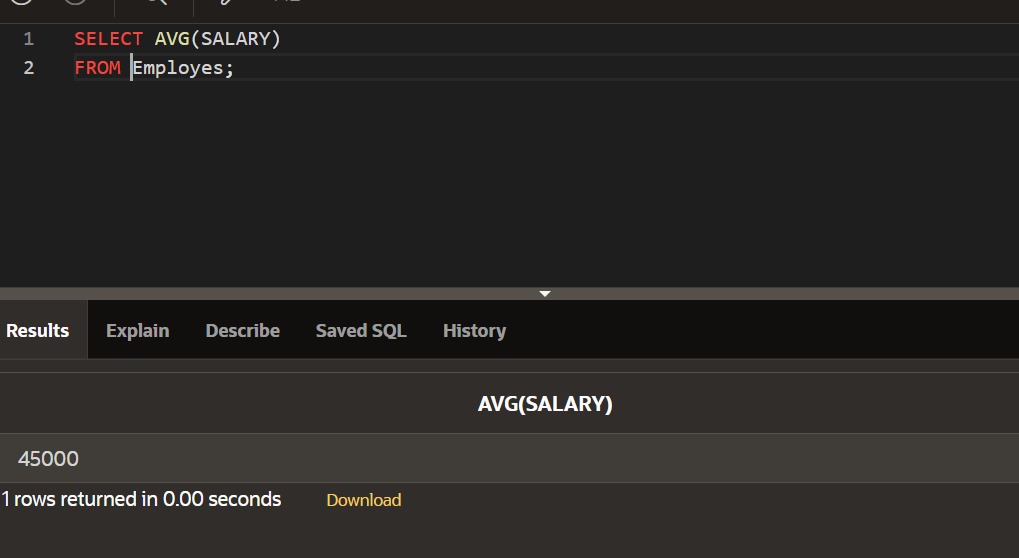
**SECTION 8**

**DP\_8.1:**

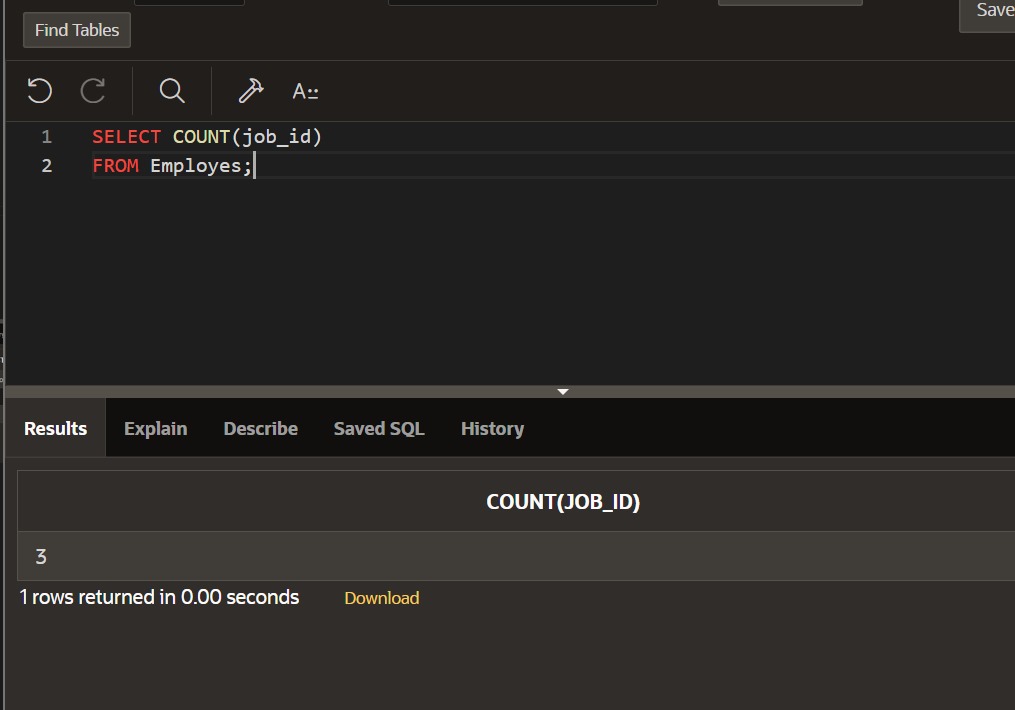
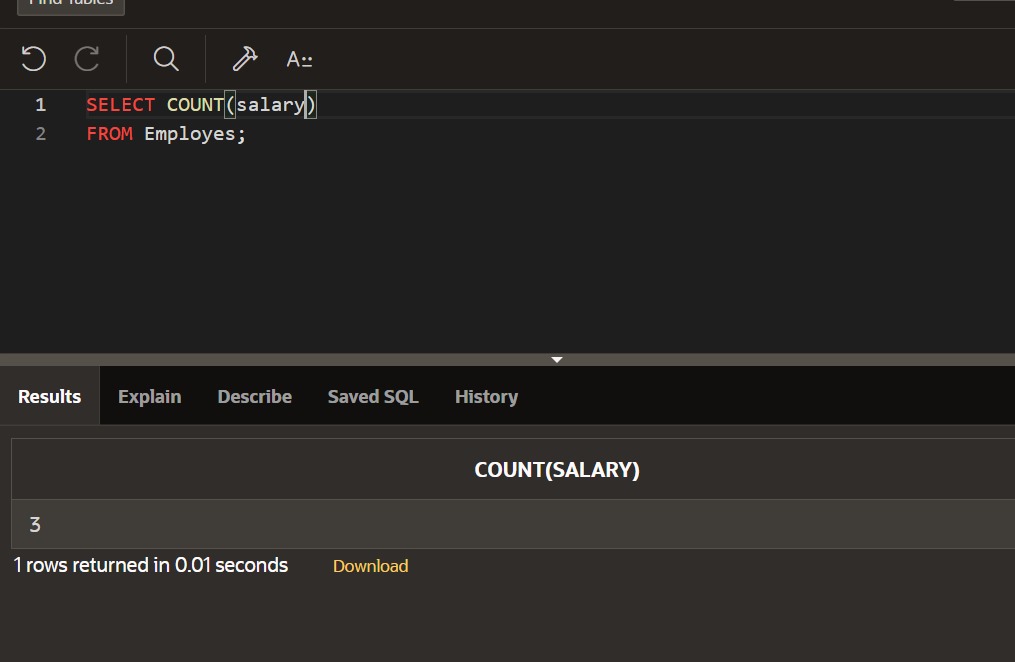
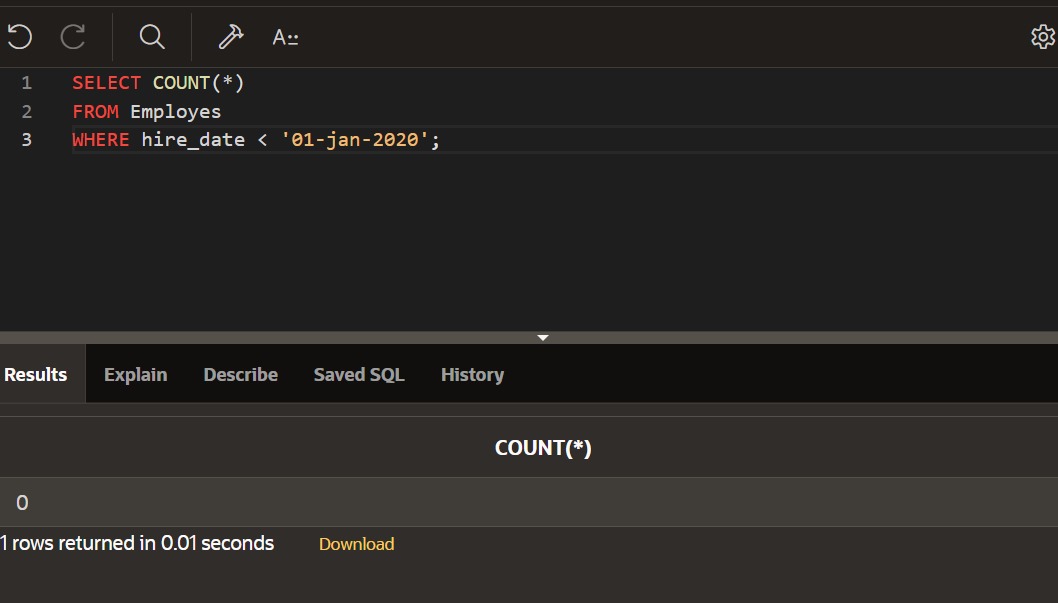
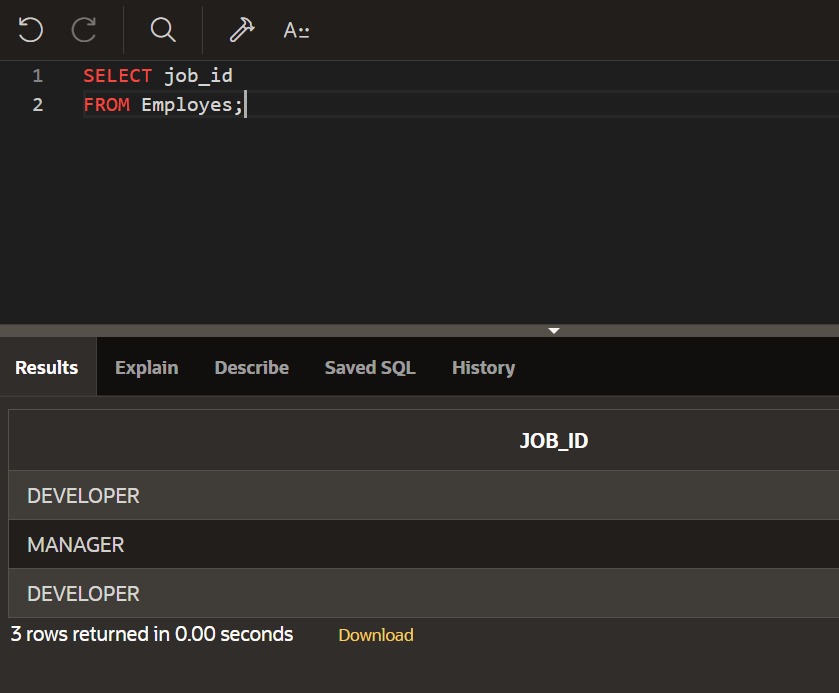
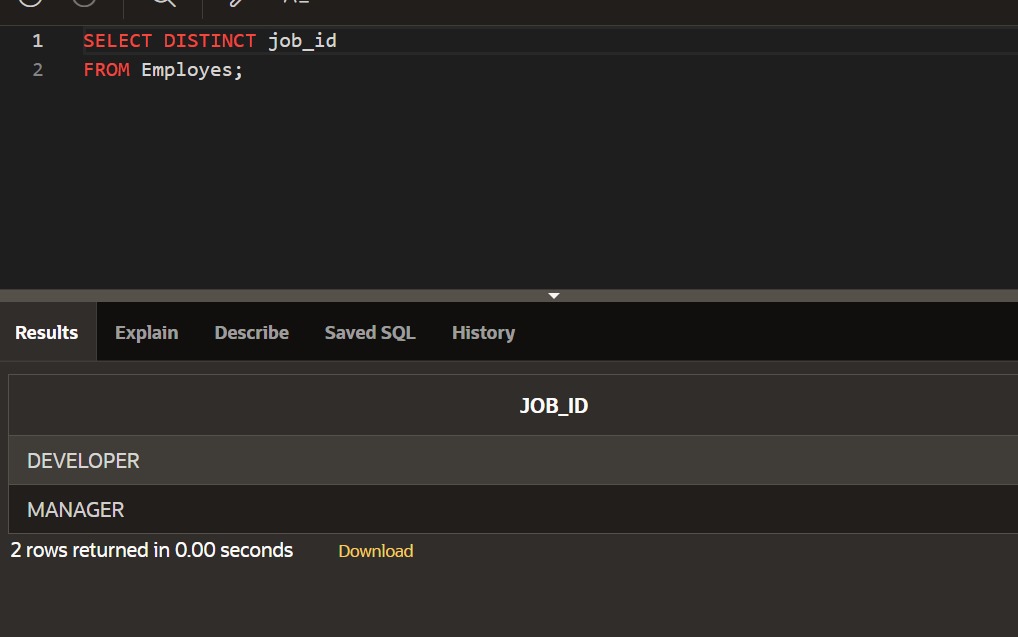
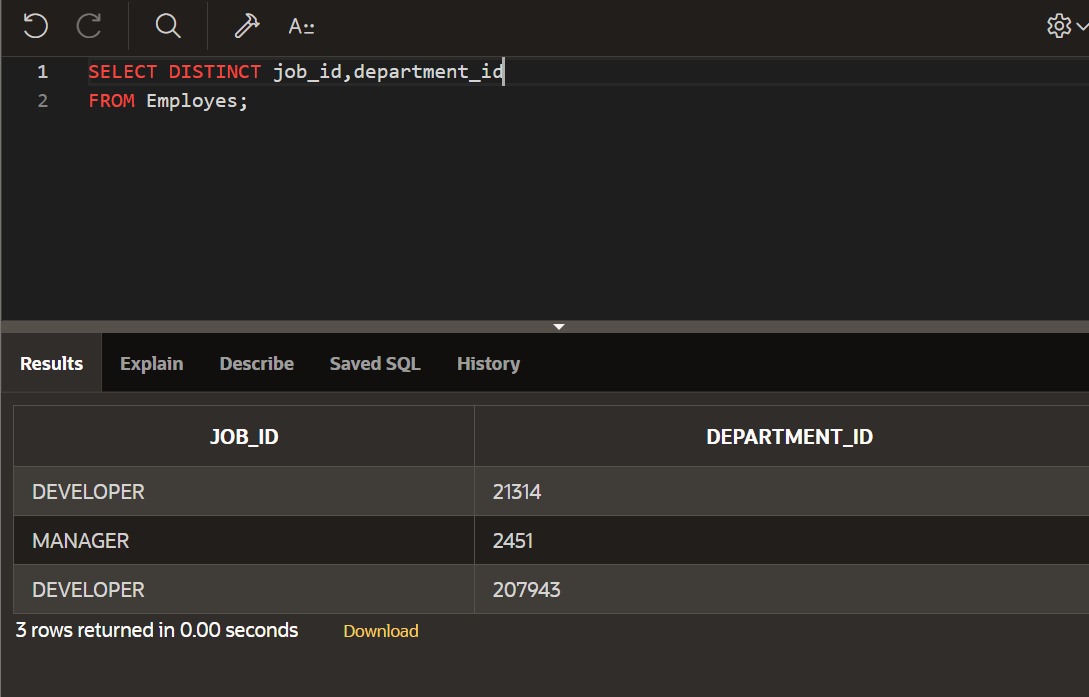
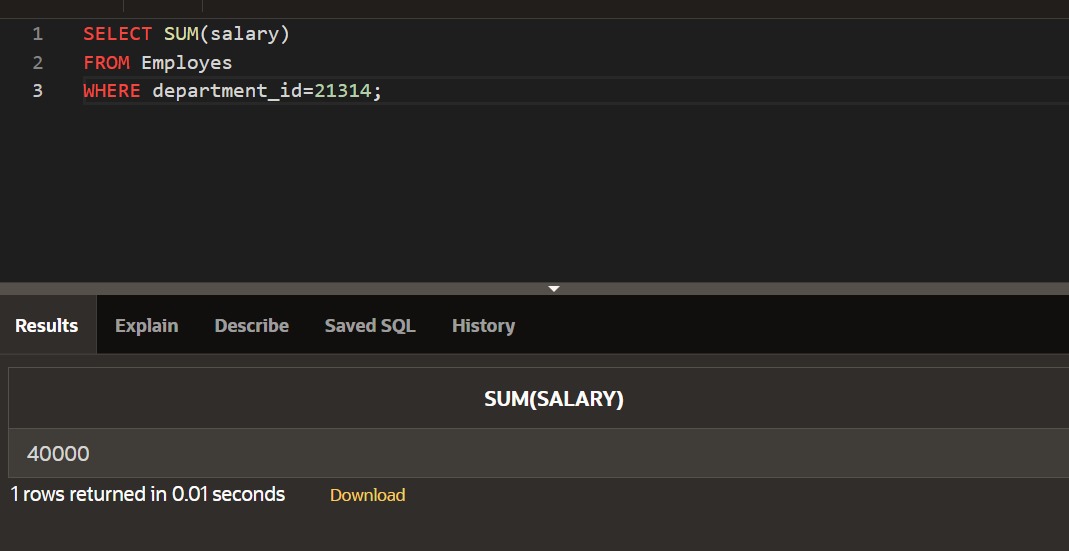
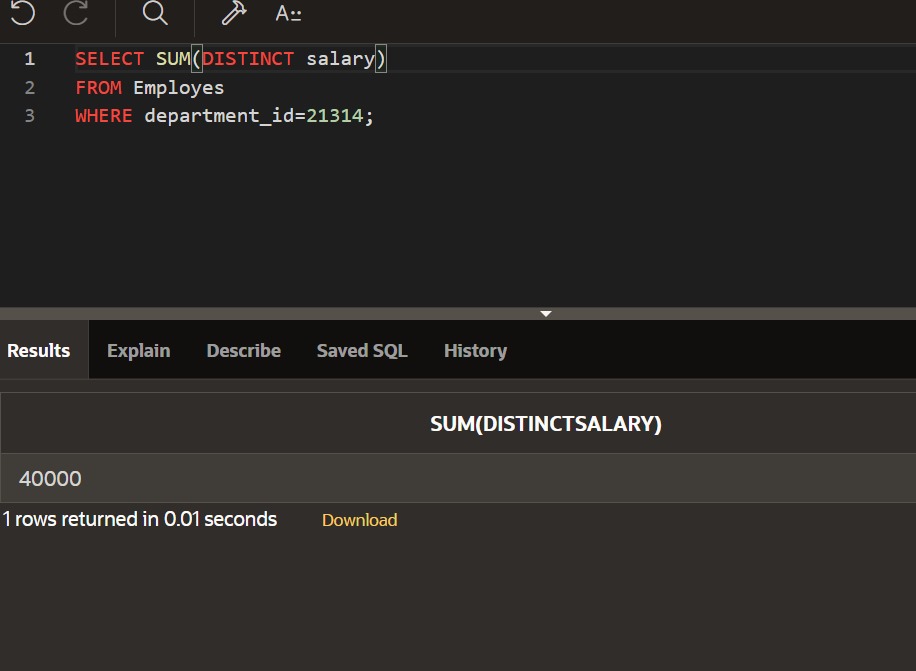
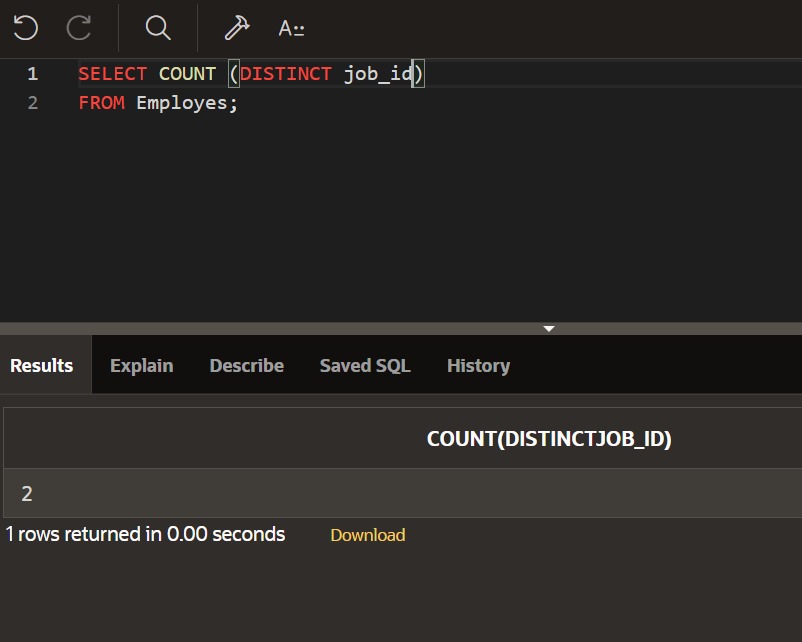
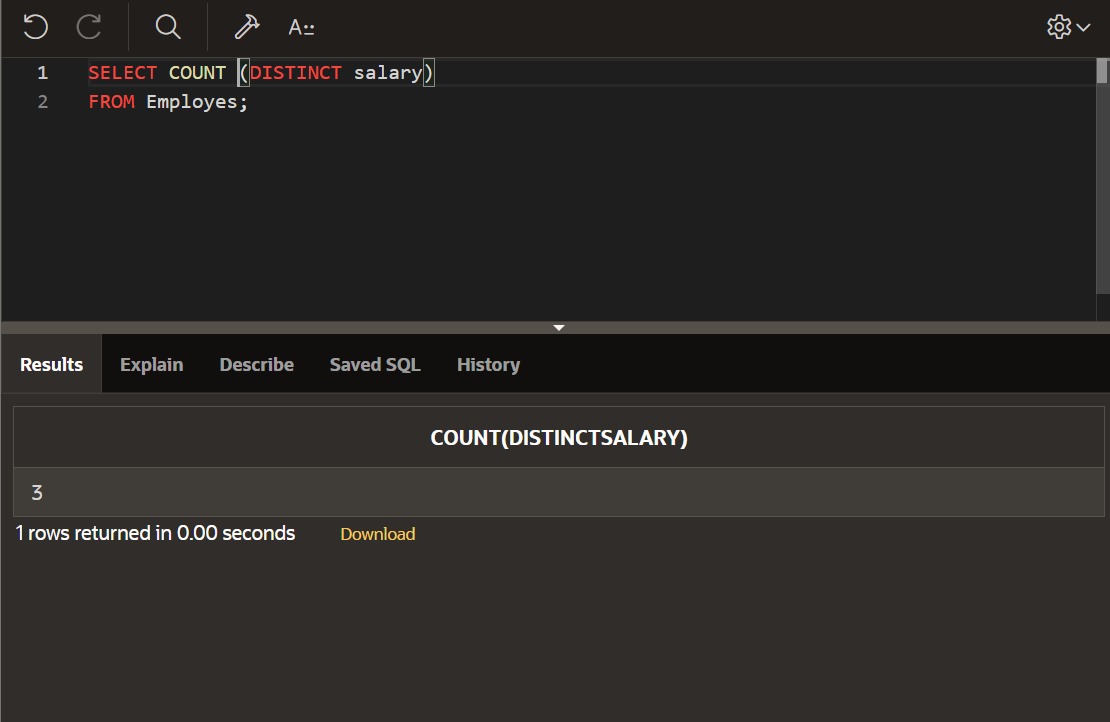
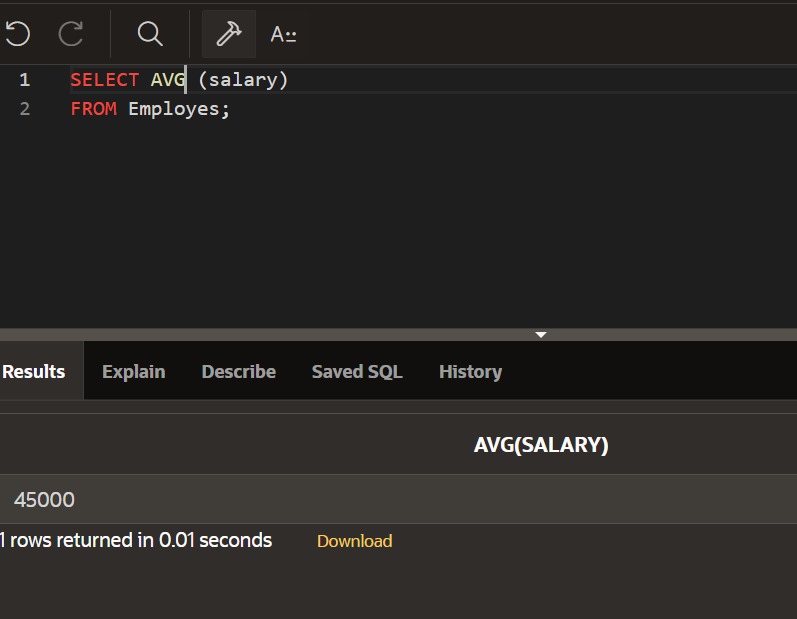


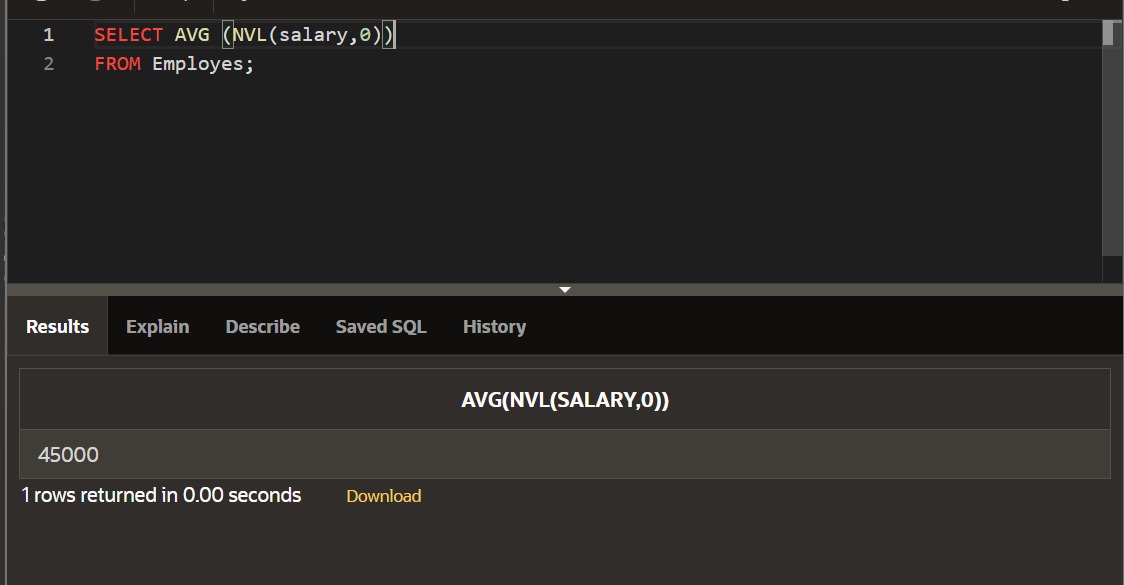


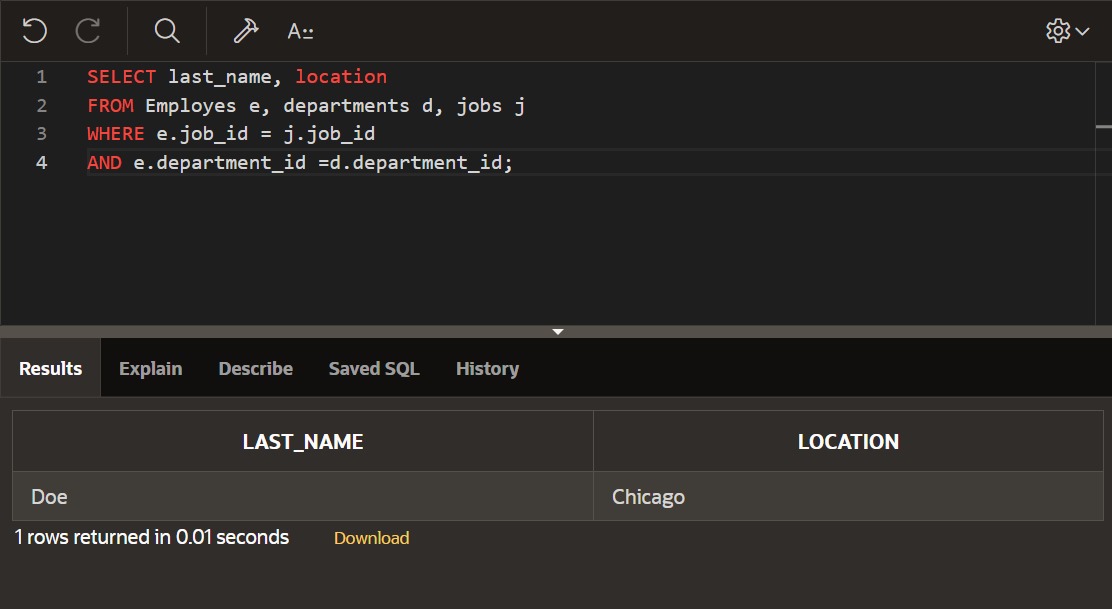




**DP\_8.2:**





**DATABASE PROGRAMMING WITH SQL**

**7.1 Oracle Equijoin and Cartesian Product**

CREATE TABLE employ(

eno VARCHAR(14),

ename VARCHAR(14),

eadhress VARCHAR(15),

epno VARCHAR(15),

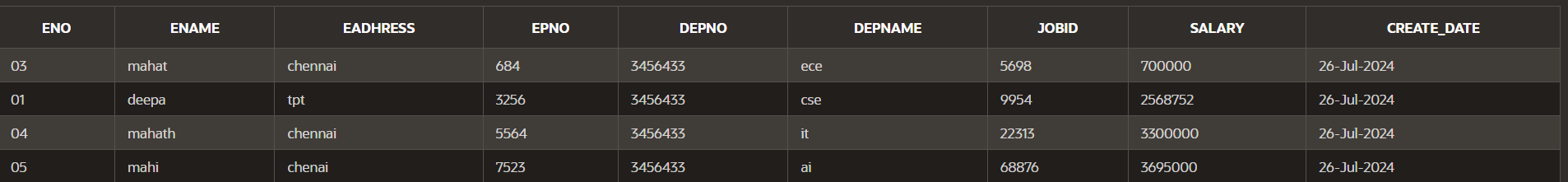
depno VARCHAR(14),

depname VARCHAR(14),

jobid VARCHAR(10),

salary VARCHAR(10),

create\_date DATE DEFAULT SYSDATE);



CREATE TABLE jobs (

job\_id VARCHAR(10) PRIMARY KEY,

job\_title VARCHAR(50) NOT NULL,

min\_salary DECIMAL(8, 2),

max\_salary DECIMAL(8, 2)

);

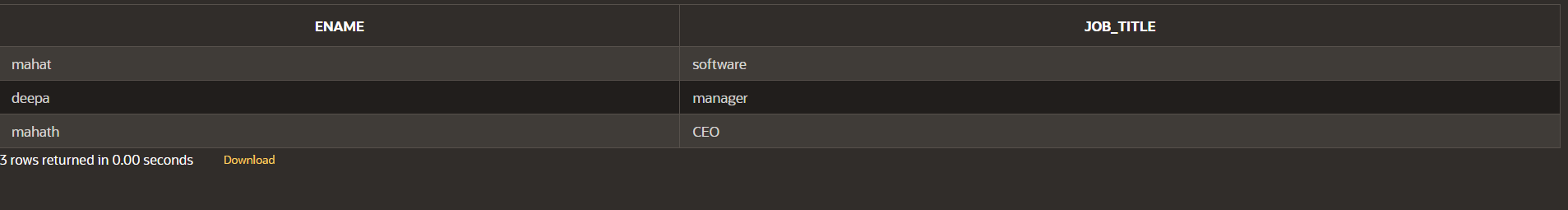


**PROPRIETARY JOINS:**

SELECT employ.ename, jobs.job\_title

FROM employ,jobs

WHERE employ.jobid=jobs.job\_id;

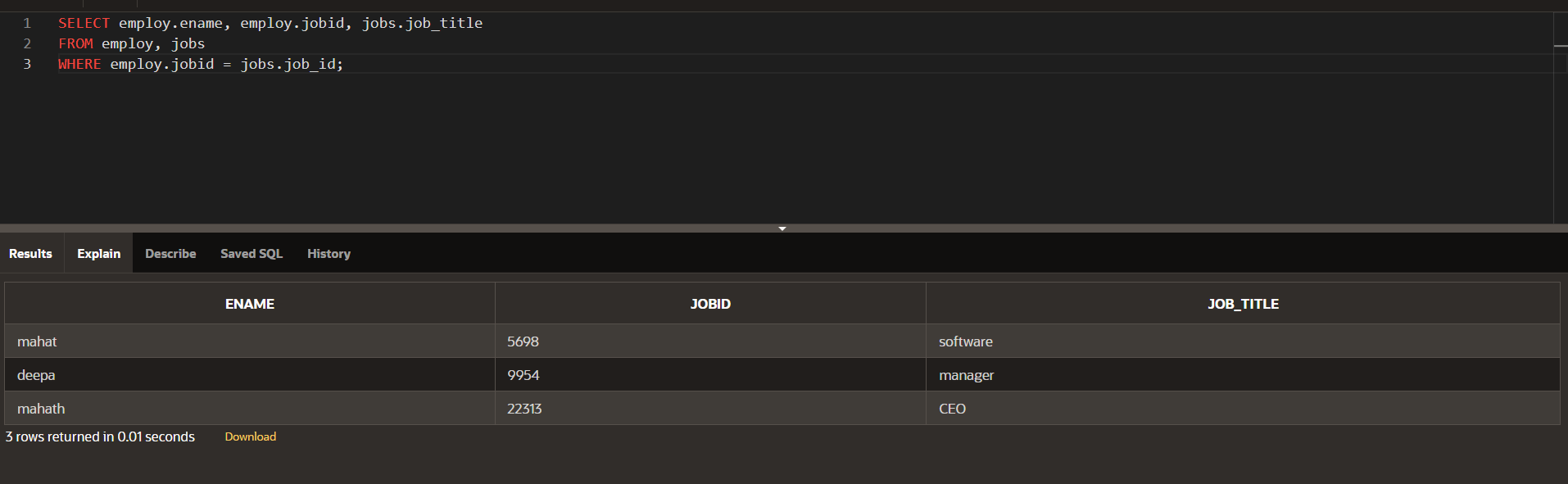


**EQUIJOIN:**

SELECT employ.ename, employ.jobid, jobs.job\_title

FROM employ, jobs

WHERE employ.jobid = jobs.job\_id;



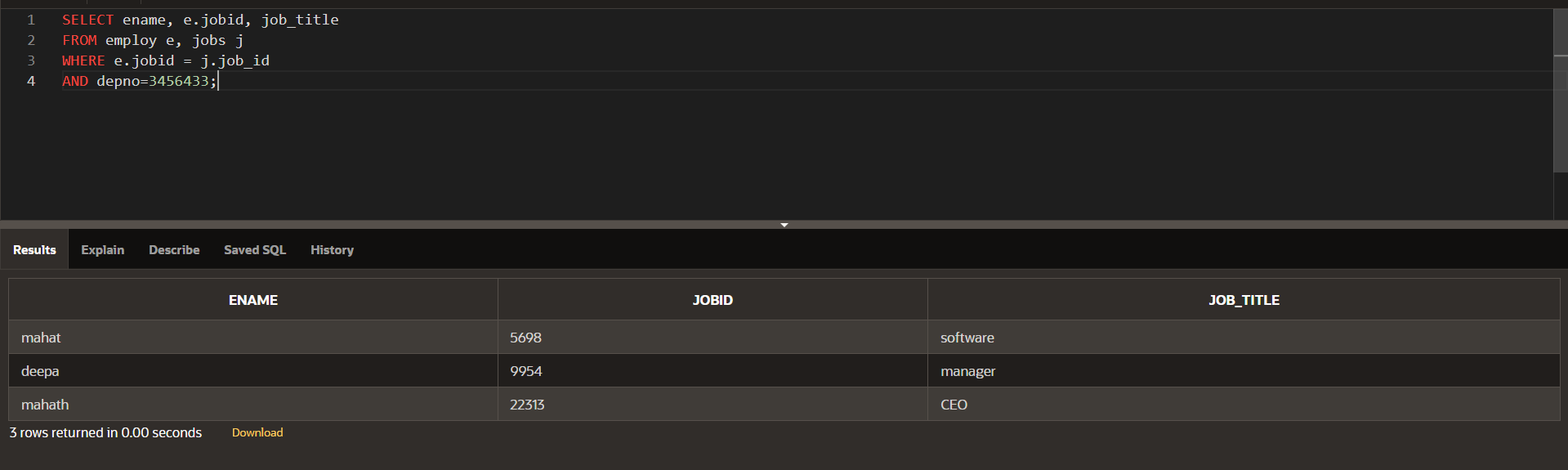
**ALIASES:**

SELECT ename, e.jobid, job\_title

FROM employ e, jobs j

WHERE e.jobid = j.job\_id

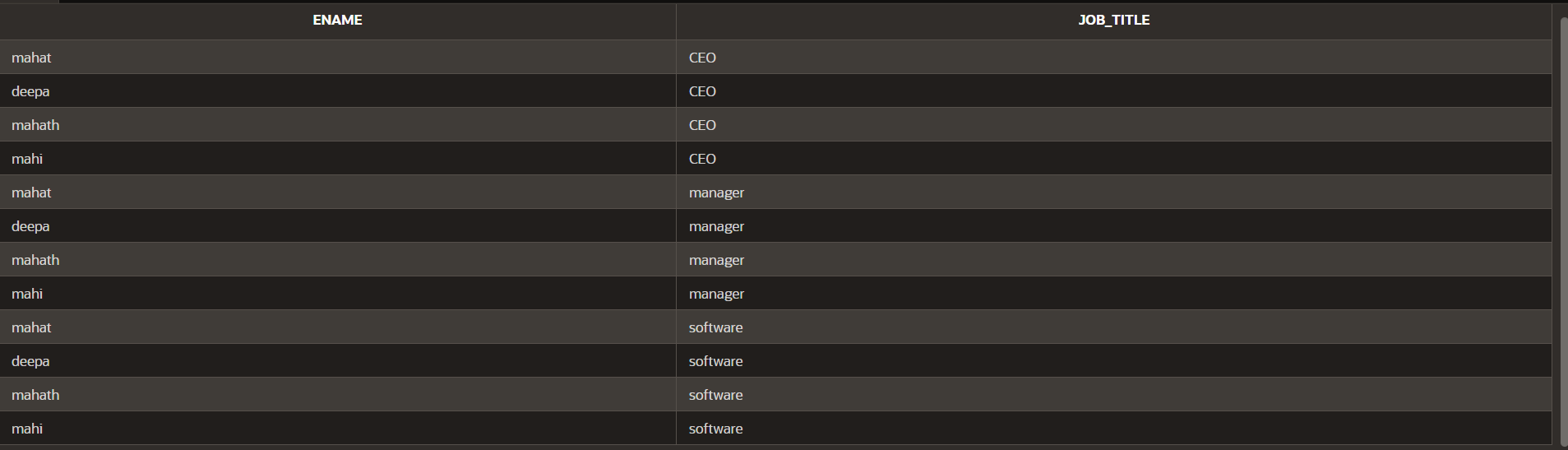
AND depno=3456433;



**CARTESIAN PRODUCT JOIN:**

SELECT employ.ename,jobs.job\_title

FROM employ,jobs;



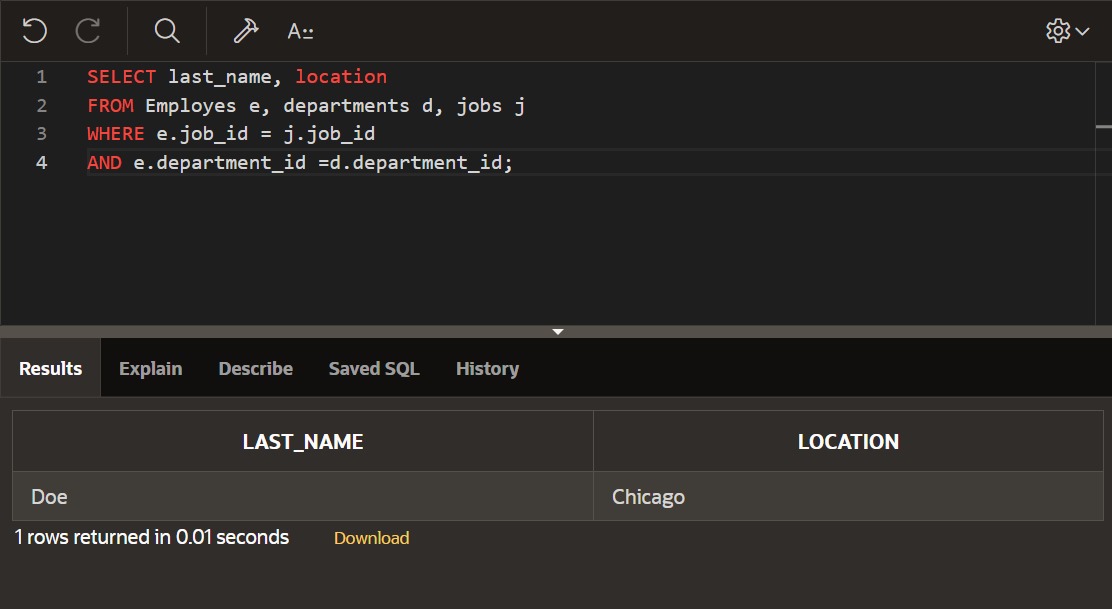
**JOIN:**

SELECT last\_name, location

FROM Employes e, departments d, jobs j

WHERE e.job\_id = j.job\_id

AND e.department\_id =d.department\_id;



**DP\_7.2:**

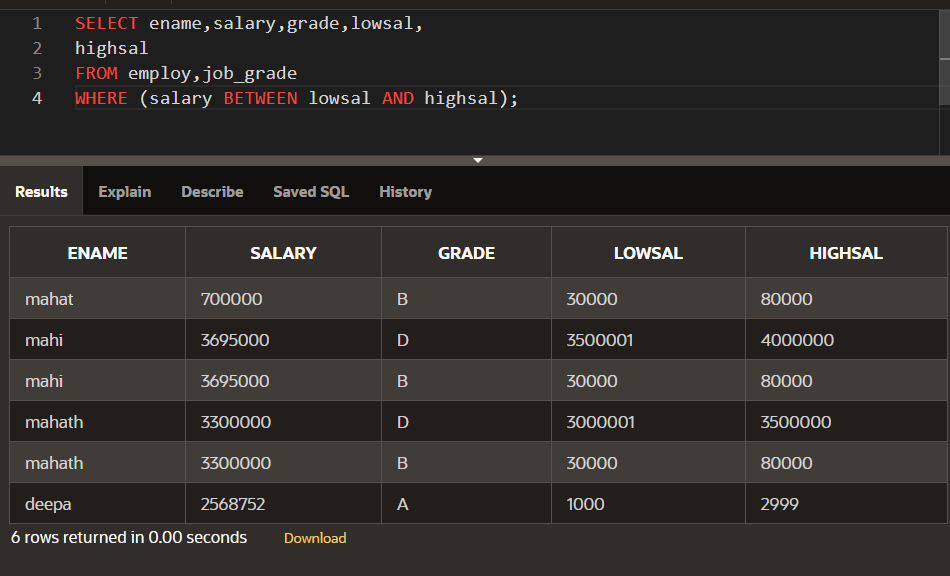
**Nonequijoin:**

SELECT ename,salary,grade,lowsal,

highsal

FROM employ,job\_grade

WHERE (salary BETWEEN lowsal AND highsal);



**Section 6**

**HEARIACHEY LEVEL**

select level id,name,deptid

from emp

start with id=100

connect by prior id=deptid;

A black and grey rectangular object

Description automatically generated with medium confidence

**HIERACHIAL USING START WITH KEYWORD**

select id,name,deptid

from emp

start with id=100

connect by prior id=deptid;

A black and grey rectangular object

Description automatically generated with medium confidence

**FULL OUTER JOIN**

select e.id,e.name,d.deptid,d.dept\_name

from emp e full outer join dept d

on (e.deptid=d.deptid);

A screenshot of a computer

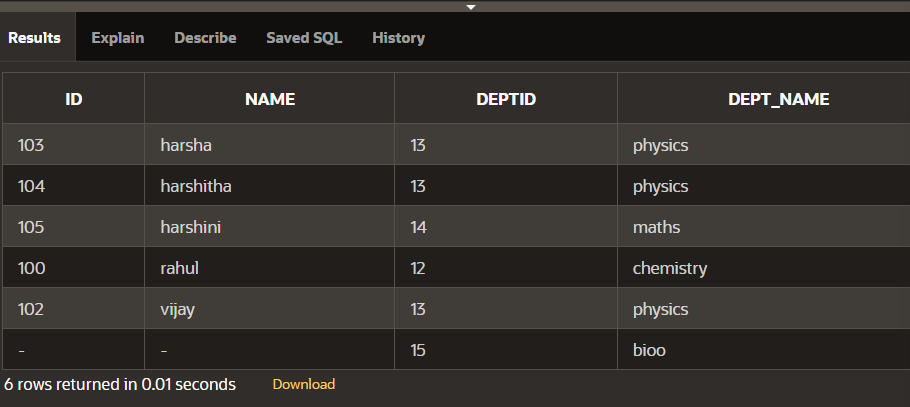
Description automatically generated

**RIGHT OUTER JOIN**

select e.id,e.name,d.deptid,d.dept\_name

from emp e right outer join dept d

on (e.deptid=d.deptid);



**LEFT OUTER JOIN**

select e.id,e.name,d.deptid,d.dept\_name

from emp e left outer join dept d

on (e.deptid=d.deptid);

A screenshot of a computer

Description automatically generated

**ON CLAUSE**

select id,name,dept\_name

from emp e join dept d

on(e.deptid=d.deptid);

A screenshot of a computer

Description automatically generated

**USING CLAUSE**

select id,name,deptid,dept\_name

from emp join dept using (deptid);

A screenshot of a computer

Description automatically generated

**CROSS JOIN**

select id,name,dept\_name

from emp cross join dept;

**A screenshot of a computer

Description automatically generated**

**NATURAL JOIN**

select id,name,deptid,dept\_name

from emp natural join dept;

A screenshot of a computer

Description automatically generated

**Section 5**

**NVL FUNCTION**

select id,nvl(dept,'not assigned')

from singer;

A screenshot of a computer

Description automatically generated

**NVL DATE**

SELECT NVL(TO\_CHAR(hiredate, 'YYYY-MM-DD'), 'no date')

FROM singer;

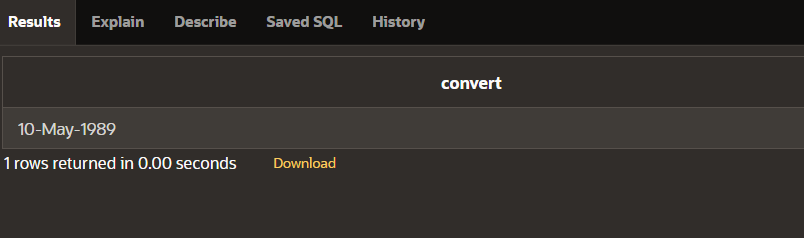
A screenshot of a computer

Description automatically generated

**CHARACTER TO DATE**

select to\_date('may10,1989','fxmondd,yyyy') as "convert"

from dual;



**NUMBER TO CHARACTER**

select to\_char(salary,'$99,999')

from singer;

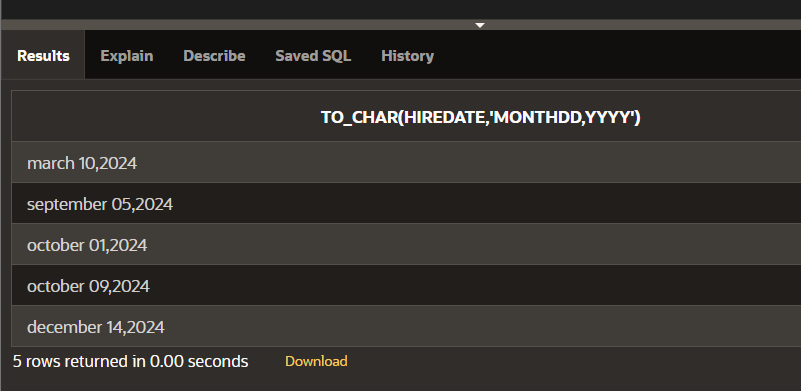
A screenshot of a computer

Description automatically generated

**DATE TO CHARACTER**

select to\_char(hiredate,'month dd,yyyy')

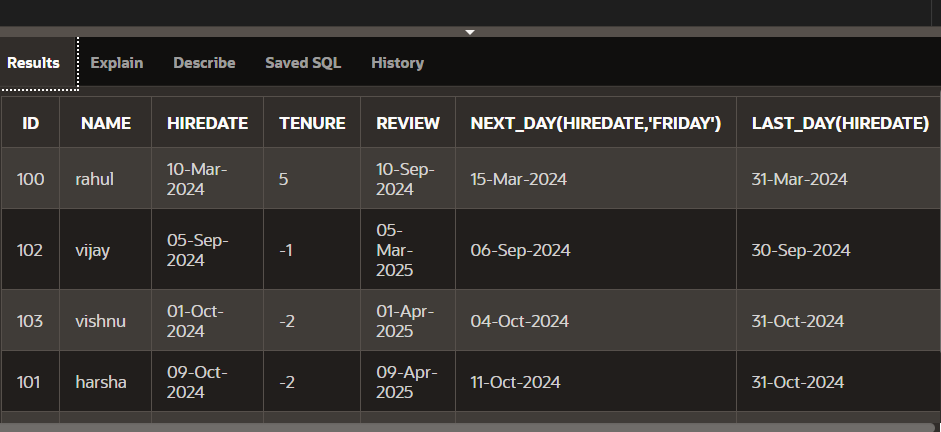
from singer;

****

**DATE FUNCTION**

select id,name,hiredate,round(months\_between(sysdate,hiredate)) as tenure, add\_months(hiredate,6) as review, next\_day(hiredate,'friday'),last\_day(hiredate)

from singer;



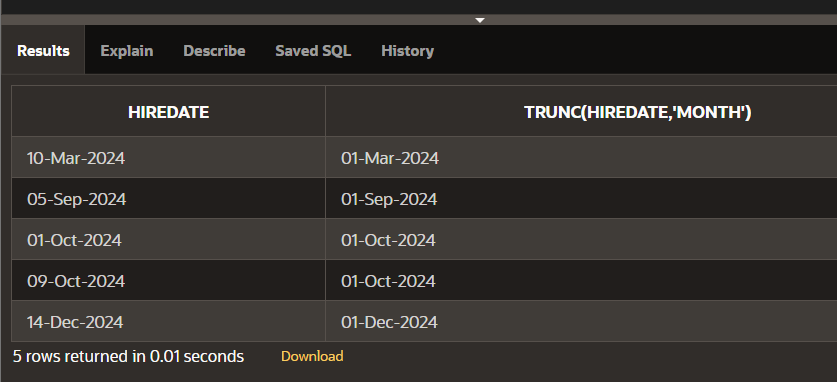
**Section 4**

**TRUNC**

select hiredate,

trunc(hiredate,'month')

from singer;

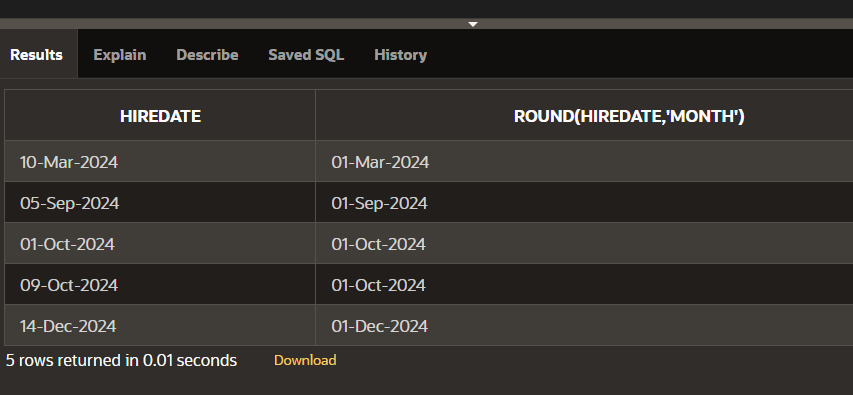


**ROUND TO HIREDATE**

select hiredate,

round(hiredate,'month')

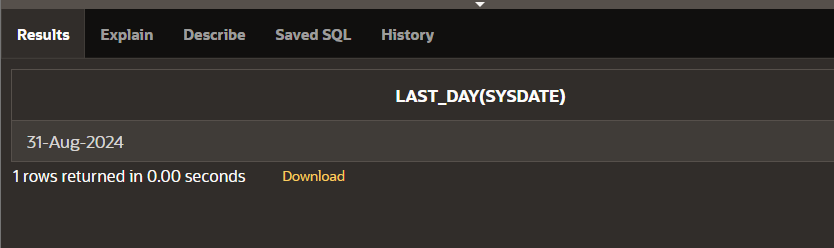
from singer;



**LAST DAY**

select last\_day(sysdate)

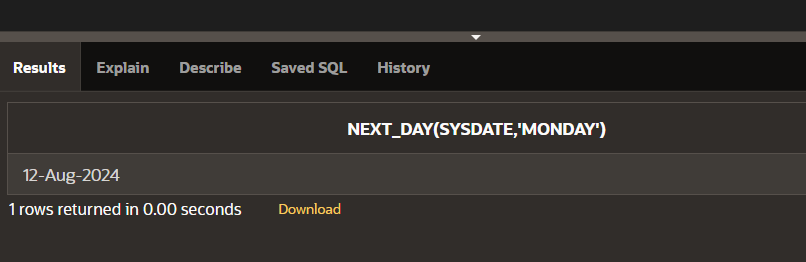
from dual;



**NEXT DAY**

select next\_day(sysdate,'monday')

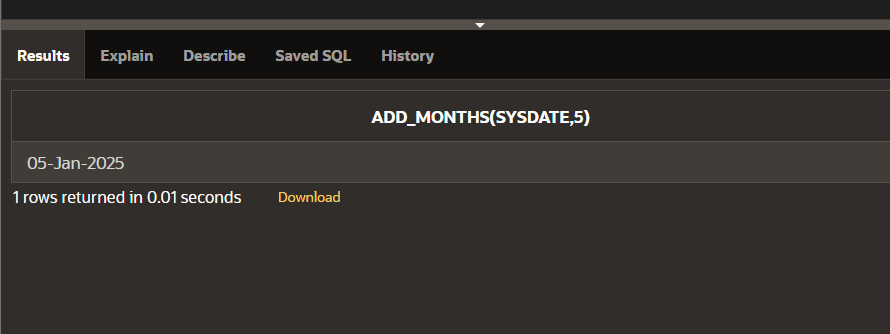
from dual;



**ADD MONTHS**

select add\_months(sysdate,5)

from dual;



**MONTHS BETWEEN**

select id,name

from singer

where months\_between

(sysdate,hiredate)<100;

