DATABASE MANAGEMENT SYSTEMS
LAB ASSSIGNMENT-1
SUBMITTED BY- RIMJHIM MITTAL
ROLL NUMBER- 102103430
CLASS- 2CO16

---

1. Create table Student (Rno, Name, DOB, Gender, Class, College, City, Marks)

```
CREATE TABLE Student(
    Rno NUMBER(9) NOT NULL,
    Stu_Name VARCHAR2(50) NOT NULL,
    DOB DATE NOT NULL,
    Gender VARCHAR2(10),
    Stu_Class VARCHAR2(10),
    College VARCHAR2(100),
    City VARCHAR2(15),
    Marks NUMBER(3));
```

+ Code + Text

```
Table created.
```

2. Insert 5 records in student table

```
INSERT
INTO Student
    VALUES( 102103430,'Rimjhim Mittal','27-JUN-03','FEMALE','2CO16','TIET','Patiala',80);
INSERT
INTO Student
    VALUES( 102103432,'Harsh Jain','13-JUN-03','MALE','2CO16','TIET','Patiala',80);
INSERT
INTO Student
    VALUES( 102103447,'Shreeya CHatterji','15-JUN-03','FEMALE','2CO16','TIET','Patiala',80);
INSERT
INTO Student
    VALUES( 102103424,'Lakshaya Aggarwal','06-NOV-03','MALE','2CO16','TIET','Patiala',80);
INSERT
INTO Student
    VALUES( 102103443,'Hrsh Dhingra','14-MAR-03','MALE','2CO16','TIET','Patiala',80);
```

```
1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

3. Display the information of all the students

```
SELECT * FROM Student;
```

RNO	STU_NAME	DOB	GENDER	STU_CLASS	COLLEGE	CITY	MARKS
102103430	Rimjhim Mittal	27-JUN-03	FEMALE	2CO16	TIET	Patiala	80
102103432	Harsh Jain	13-JUN-03	MALE	2CO16	TIET	Patiala	80
102103447	Shreeya CHatterji	15-JUN-03	FEMALE	2CO16	TIET	Patiala	80
102103424	Lakshaya Aggarwal	06-NOV-03	MALE	2CO16	TIET	Patiala	80
102103443	Hrsh Dhingra	14-MAR-03	MALE	2CO16	TIET	Patiala	80

Download CSV

5 rows selected.

4. Display the detail structure of student table

```
DESCRIBE Student;
```

TABLE STUDENT

Column	Null?	Type
RNO	NOT NULL	NUMBER(9,0)
STU_NAME	NOT NULL	VARCHAR2(50)
DOB	NOT NULL	DATE
GENDER	-	VARCHAR2(10)
STU_CLASS	-	VARCHAR2(10)
COLLEGE	-	VARCHAR2(100)
CITY	-	VARCHAR2(15)
MARKS	-	NUMBER(3,0)

Download CSV

5. Display Rno, Name and Class information of ‘Patiala’ students.

```
SELECT Rno, stu_name, stu_class
FROM STUDENT
WHERE CITY= 'Patiala';
```

RNO	STU_NAME	STU_CLASS
102103430	Rimjhim Mittal	2CO16
102103432	Harsh Jain	2CO16
102103447	Shreeya CHatterji	2CO16
102103424	Lakshaya Aggarwal	2CO16
102103443	Hrsh Dhingra	2CO16

Download CSV

5 rows selected.

6. Display information on ascending order of RNo

```
SELECT *
FROM Student
ORDER BY Rno;
```

RNO	STU_NAME	DOB	GENDER	STU_CLASS	COLLEGE	CITY	MARKS
102103424	Lakshaya Aggarwal	06-NOV-03	MALE	2CO16	TIET	Patiala	80
102103430	Rimjhim Mittal	27-JUN-03	FEMALE	2CO16	TIET	Patiala	80
102103432	Harsh Jain	13-JUN-03	MALE	2CO16	TIET	Patiala	80
102103443	Hrsh Dhingra	14-MAR-03	MALE	2CO16	TIET	Patiala	80
102103447	Shreeya CHatterji	15-JUN-03	FEMALE	2CO16	TIET	Patiala	80

Download CSV

5 rows selected.

7. Change the marks of Rno 102103432 to 89.

```
UPDATE Student
SET Marks = 89
WHERE RNo= 102103432;
SELECT * FROM Student
WHERE RNo= 102103432;
```

RNO	STU_NAME	DOB	GENDER	STU_CLASS	COLLEGE	CITY	MARKS
102103432	Harsh Jain	13-JUN-03	MALE	2CO16	TIET	Patiala	89

Download CSV

8. Change the name and city of Rno 102103424

```
UPDATE Student
SET City= 'Delhi',
Stu_Name= 'Laccha Aggarwal'
WHERE RNo= 102103424;

SELECT * FROM Student
WHERE RNo= 102103424;
```

1 row(s) updated.

RNO	STU_NAME	DOB	GENDER	STU_CLASS	COLLEGE	CITY	MARKS
102103424	Laccha Aggarwal	06-NOV-03	MALE	2CO16	TIET	Delhi	80

Download CSV

9. Delete the information of ‘Delhi’ city records

```
DELETE FROM Student
WHERE City= 'Delhi';
SELECT * FROM Student;
```

RNO	STU_NAME	DOB	GENDER	STU_CLASS	COLLEGE	CITY	MARKS
102103430	Rimjhim Mittal	27-JUN-03	FEMALE	2CO16	TIET	Patiala	80
102103432	Harsh Jain	13-JUN-03	MALE	2CO16	TIET	Patiala	89
102103447	Shreeya CHatterji	15-JUN-03	FEMALE	2CO16	TIET	Patiala	80
102103443	Hrsh Dhingra	14-MAR-03	MALE	2CO16	TIET	Patiala	80

Download CSV

4 rows selected.

10. Delete the records of student where marks<30.

```
DELETE FROM Student
WHERE Marks>85;

SELECT * FROM Student;
```

1 row(s) deleted.

RNO	STU_NAME	DOB	GENDER	STU_CLASS	COLLEGE	CITY	MARKS
102103430	Rimjhim Mittal	27-JUN-03	FEMALE	2CO16	TIET	Patiala	80
102103447	Shreeya CHatterji	15-JUN-03	FEMALE	2CO16	TIET	Patiala	80
102103443	Hrsh Dhingra	14-MAR-03	MALE	2CO16	TIET	Patiala	80

Download CSV

3 rows selected.

DATABASE MANAGEMENT SYSTEMS
LAB ASSSIGNMENT-2
SUBMITTED BY- RIMJHIM MITTAL
ROLL NUMBER- 102103430
CLASS- 2CO16

Based on Emp table
Columns are EmpNo, Ename, Job, Salary, Commission, DeptNO
Insert 5 records by storing Null value in some records for commission column.

```
CREATE TABLE EMP
       (EMPNO NUMBER(4) NOT NULL,
        ENAME VARCHAR2(10),
        EJOB VARCHAR2(9),
        SAL NUMBER(7, 2),
        COMM NUMBER(7, 2),
        DEPTNO NUMBER(2));

INSERT INTO EMP VALUES
        (7369, 'SMITH',  'CLERK',  800, NULL, 20);
INSERT INTO EMP VALUES
        (7499, 'ALLEN',  'SALESMAN', 1600,  300, 30);
INSERT INTO EMP VALUES
        (7521, 'ALARIC',   'SALESMAN', 1250,  500, 10);
INSERT INTO EMP VALUES
        (7566, 'JONES',  'MANAGER',  2975, NULL, 20);
INSERT INTO EMP VALUES
        (7654, 'MARTIN', 'SALESMAN', 1250, 1400, 30);
INSERT INTO EMP VALUES
        (7698, 'RAJ',  'CLERK',  2850, NULL, 30);
INSERT INTO EMP VALUES
        (7782, 'CLARK',  'MANAGER',  2450, NULL, 10);
INSERT INTO EMP VALUES
        (7788, 'RAMADA',  'ANALYST', 3000, NULL, 20);

SELECT * FROM EMP;
```

EMPNO	ENAME	EJOB	SAL	COMM	DEPTNO
7369	SMITH	CLERK	800	-	20
7499	ALLEN	SALESMAN	1600	300	30
7521	ALARIC	SALESMAN	1250	500	10
7566	JONES	MANAGER	2975	-	20
7654	MARTIN	SALESMAN	1250	1400	30
7698	RAJ	CLERK	2850	-	30
7782	CLARK	MANAGER	2450	-	10
7788	RAMADA	ANALYST	3000	-	20

Q1) Get employee no and employee name who works in dept no 10 ?

```
SELECT EMPNO, ENAME
FROM EMP
WHERE DEPTNO= 10;
```

EMPNO	ENAME
7521	ALARIC
7782	CLARK

Q2) Display the employee names of those clerks whose salary is greater than 2000 ?

```
SELECT ENAME
FROM EMP
WHERE EJOB='CLERK'
AND SAL>2000;
```

ENAME
RAJ

Q3) Display name and job of Salesperson & Clerks ?

```
SELECT ENAME , EJOB
FROM EMP
WHERE EJOB='CLERK'
OR EJOB='SALESMAN';
```

ENAME	EJOB
SMITH	CLERK
ALLEN	SALESMAN
ALARIC	SALESMAN
MARTIN	SALESMAN
RAJ	CLERK

Q4) Display all details of employees whose salary between 2000 and 3000 ?

```
SELECT *
FROM EMP
WHERE SAL
BETWEEN 2000 AND 3000;
```

EMPNO	ENAME	EJOB	SAL	COMM	DEPTNO
7566	JONES	MANAGER	2975	-	20
7698	RAJ	CLERK	2850	-	30
7782	CLARK	MANAGER	2450	-	10
7788	RAMADA	ANALYST	3000	-	20

Q5) Display all details of employees whose dept no is 10, 20, or 30 ?

```
SELECT *
FROM EMP
WHERE DEPTNO IN (10,20,30);
```

EMPNO	ENAME	EJOB	SAL	COMM	DEPTNO
7369	SMITH	CLERK	800	-	20
7499	ALLEN	SALESMAN	1600	300	30
7521	ALARIC	SALESMAN	1250	500	10
7566	JONES	MANAGER	2975	-	20
7654	MARTIN	SALESMAN	1250	1400	30
7698	RAJ	CLERK	2850	-	30
7782	CLARK	MANAGER	2450	-	10
7788	RAMADA	ANALYST	3000	-	20

Q6) Display name of those employees whose commission is NULL ?

```
SELECT ENAME
FROM EMP
WHERE COMM IS NULL;
```

ENAME
SMITH
JONES
RAJ
CLARK
RAMADA

Q7) Display dept no & salary in ascending order of dept no and with in each dept no salary should be in descending order ?

```
SELECT DEPTNO, SAL
FROM EMP
ORDER BY DEPTNO, SAL DESC;
```

DEPTNO	SAL
10	2450
10	1250
20	3000
20	2975
20	800
30	2850
30	1600
30	1250

Q8) Display name of employees having two ‘a’ or ‘A’ chars in the name ?

```
SELECT ENAME
FROM EMP
WHERE  ENAME LIKE '%A%A%'
OR ENAME LIKE '%a%a%' ;
```

ENAME
ALARIC
RAMADA

Q9) Display the name of the employees whose second char is ‘l’ or ‘L’ ?

```
SELECT ENAME
FROM EMP
WHERE  ENAME LIKE '_L%'
OR ENAME LIKE '_l%' ;
```

ENAME
ALLEN
ALARIC
CLARK

Q10) Display the name of the employees whose first or last char is ‘a’ or ‘A’ ?

```
SELECT ENAME
FROM EMP
WHERE  ENAME LIKE 'a%'
OR ENAME LIKE 'A%'
OR ENAME LIKE '%A'
OR ENAME LIKE '%a' ;
```

ENAME
ALLEN
ALARIC
RAMADA

Q11) Display maximum, minimum, average salary of deptno 10 employees.

```
SELECT AVG(SAL), MAX(SAL), MIN(SAL)
FROM emp
WHERE DEPTNO =  10;
```

AVG(SAL)	MAX(SAL)	MIN(SAL)
1850	2450	1250

Q12) Display total number of employees working in deptno 20

```
SELECT COUNT(*)
FROM EMP
WHERE DEPTNO= 20;
```

COUNT(*)

Q13) Display total salary paid to clerks

```
SELECT SUM(SAL)
FROM EMP
WHERE EJOB= 'CLERK';
```

SUM(SAL)
3650

Q14) Display system date

```
SELECT TO_CHAR(SYSDATE, 'MM-DD-YYYY HH24:MI:SS') "NOW"
FROM DUAL;
```

NOW
01-24-2023 18:02:26

Q15) Display the result of (12*12)/13

```
SELECT (12*12)/13
FROM DUAL;
```

(12*12)/13
11.07692307692307692307692307692307692308

Q16) Display info of ‘raj’ irrespective to the case in which the data is stored.

```
SELECT *
FROM EMP
WHERE UPPER(ENAME) ='RAJ';
```

EMPNO	ENAME	EJOB	SAL	COMM	DEPTNO
7698	RAJ	CLERK	2850	-	30

DATABASE MANAGEMENT SYSTEMS
LAB ASSIGNMENT-3
SUBMITTED BY- RIMJHIM MITTAL
ROLL NUMBER- 102103430
CLASS- 2CO16

Q1) Use the following functions

1. **chr (n):**

```
SELECT CHR(82) || CHR(73)|| CHR(77)|| CHR(74)|| CHR(72)|| CHR(73)|| CHR(77) FROM
```

CHR(82)||CHR(73)||CHR(77)||CHR(74)||CHR(72)||CHR(73)||CHR(77)
RIMJHIM

2. **concat(char1,char2):**

```
SELECT CONCAT(CONCAT('Hello', ' '), 'World') FROM DUAL;
```

CONCAT(CONCAT('HELLO',''),'WORLD')
Hello World

3. **instr(string,char):**

```
SELECT INSTR('RIMJHIM', 'I', 2) FROM DUAL;
```

INSTR('RIMJHIM','I',2)
2

4. **length(n):**

```
SELECT LENGTH('PL/SQL') FROM DUAL;
```

LENGTH('PL/SQL')
6

5. **lpad(char1 ,n [,char2]):**

```
SELECT LPAD('MITTAL', 14, 'RIMJHIM ') from dual;
```

LPAD('MITTAL',14,'RIMJHIM')
RIMJHIM MITTAL

6. **ltrim(string [,char(s)]):**

```
SELECT LTRIM('HELLO','HE') from dual;
```

LTRIM('HELLO','HE')
LLO

7. **rpad(char1 ,n [,char2]):**

```
SELECT RPAD('MITTAL', 14, 'RIMJHIM ') from dual;
```

RPAD('MITTAL',14,'RIMJHIM')
MITTALRIMJHIM

8. **rtrim(string [,char(s)]):**

```
SELECT RTRIM('HELLO','OL') from dual;
```

RTRIM('HELLO','OL')
HE

9. **replace(char ,search_string , replacement_string):**

```
SELECT replace('I AM A GOOD GIRL' ,'GOOD' , 'BAD') from dual;
```

REPLACE('IAMAGOODGIRL','GOOD','BAD')
I AM A BAD GIRL

10. **substr(string ,position ,substring length):**

```
SELECT SUBSTR('ABCDEFG',-5,4) "Substring" FROM DUAL;
```

Substring
CDEF

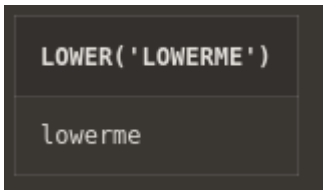
11. **initcap(char):**

```
SELECT INITCAP('the HELLO soap') "Capitals"
FROM DUAL;
```

Capitals
The Hello Soap

12. **lower(string):**

```
SELECT LOWER('LOWERME')
FROM DUAL;
```



13. **upper(string):**

```
SELECT UPPER('uPpeRCase') "Name"
FROM DUAL;
```

14. **translate(char ,from string ,to string):**

```
SELECT TRANSLATE('FATHER', 'FA', 'MO') FROM DUAL;
```

15. **abs(n):**

```
SELECT ABS(-28) "Absolute" FROM DUAL;
```

16. **ceil(n):**

```
SELECT CEIL(3.545) FROM DUAL;
```

CEIL(3.545)
4

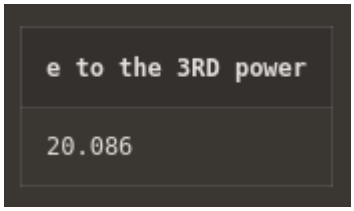
17. **cos(n):**

```
SELECT FLOOR(COS(3.14159265359)) "Cosine of 180 degrees" FROM DUAL;
```

Cosine of 180 degrees
-1

18. **exp(n):**

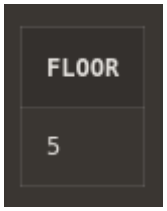
```
SELECT ROUND(EXP(3),3) "e to the 3RD power" FROM DUAL;
```



e to the 3RD power
20.086

19. **floor(n):**

```
SELECT FLOOR(5.34) "FLOOR" FROM DUAL;
```



FLOOR
5

20. **mod(m ,n):**

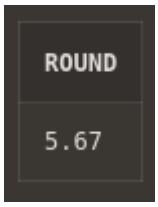
```
SELECT MOD(38,5) "MOD" FROM DUAL;
```

21. **power(x ,y):**

```
SELECT power(5 ,3) "POWER" FROM DUAL;
```

22. **round(x [,y]):**

```
SELECT round(5.67483648, 2) "ROUND" FROM DUAL;
```



23.**sign(n):**

```
SELECT SIGN(-15) FROM DUAL;
SELECT SIGN(8) FROM DUAL;
SELECT SIGN(-0) FROM DUAL;
```

SIGN(-15)
-1

Download CSV

SIGN(8)
1

Download CSV

SIGN(-0)
0

24. **sqrt(n);**

```
SELECT Sqrt(256) FROM DUAL;
```

SQRT(256)
16

25. **trunc(x ,n):**

```
SELECT trunc(25465.57895, 2), trunc(25465.57895, -1), trunc(25465.57895, 0)  FROM D
```

TRUNC(25465.57895,2)	TRUNC(225465.57895,-1)	TRUNC(25465.57895,0)
25465.57	225460	25465

26. **sysdate:**

```
SELECT TO_CHAR(SYSDATE, 'MM-DD-YYYY HH24:MI:SS') "NOW" FROM DUAL;
```

NOW
01-25-2023 15:56:11

27.**add_months(d ,n):**

```
SELECT ADD_MONTHS('27-JAN-03', 5)
FROM DUAL;
```

ADD_MONTHS('27-JAN-03',5)
27-JUN-03

28. **last_day():**

```
SELECT last_day('20-FEB-2013')
FROM DUAL;
```

LAST_DAY('20-FEB-2013')
28-FEB-13

29.**months_between(date1 ,date2):**

```
SELECT months_between('15-AUG-1947', '30-JAN-1949')
FROM DUAL;
```

MONTHS_BETWEEN('15-AUG-1947','30-JAN-1949')
-17.483870967741935483870967741935483870967

30. **next_day(date ,char):**

```
SELECT next_day('28-FEB-12', 2), NEXT_DAY('24-JUN-22', 'MONDAY')
FROM DUAL;
--This assumes 2 to be monday
```

NEXT_DAY('28-FEB-12',2)	NEXT_DAY('24-JUN-22','MONDAY')
05-MAR-12	27-JUN-22

31. **greatest(expr):**

```
SELECT GREATEST('HARRY', 'HARRIOT', 'HAROLD') "Greatest",
GREATEST(1, '3.925', '2.4') "Greatest"
FROM DUAL;
```

--If the first expr is numeric, then Oracle determines the argument with the highest numeric precedence, implicitly converts the remaining arguments to that data type before the comparison, and returns that data type. If the first expr is not numeric, then each expr after the first is implicitly converted to the data type of the first expr before the comparison.Character comparison is based on the numerical codes of the characters in the database character set and is performed on whole strings treated as one sequence of bytes, rather than character by character.

Greatest	Greatest
HARRY	3.925

32.**least(expr):**

```
SELECT LEAST('HARRY', 'HARRIOT', 'HAROLD') "Least",
LEAST(1, '3.925', '2.4') "LEAST"
FROM DUAL;
```

Least	LEAST
HAROLD	1

Q2) Display current time in hour : min : sec format

```
select to_char(sysdate, 'HH24:MI:SS') as "Current Time"
from dual;
```

Current Time
06:31:55

Q3) Display salary + commission of emp table

```
SELECT ENAME, COMM, SAL, (SAL+NVL(COMM, 0)) FROM EMP;
```

ENAME	COMM	SAL	(SAL+NVL(COMM,0))
ALARIC	500	1250	1750
JONES	-	2975	2975
MARTIN	1400	1250	2650
RAJ	-	2850	2850
CLARK	-	2450	2450
RAMADA	-	3000	3000
SMITH	-	800	800
ALLEN	300	1600	1900

Q4) Store any date value in hiredate column of table ?

```
ALTER TABLE EMP
ADD hire_date DATE;
UPDATE EMP
SET hire_date='21-feb-16'
WHERE ENAME = 'ALARIC';
UPDATE EMP
SET hire_date='01-mar-85'
WHERE ENAME = 'JONES';
UPDATE EMP
SET hire_date='05-DEC-20'
WHERE ENAME = 'MARTIN';
UPDATE EMP
SET hire_date='30-JAN-00'
```

```
WHERE ENAME = 'RAJ';
UPDATE EMP
SET hire_date='05-DEC-85'
WHERE ENAME = 'CLARK';
UPDATE EMP
SET hire_date='05-JAN-23'
WHERE ENAME = 'SMITH';

SELECT ENAME, hire_date from EMP;
```

ENAME	HIRE_DATE
ALARIC	21-FEB-16
JONES	01-MAR-85
MARTIN	05-DEC-20
RAJ	30-JAN-00
CLARK	05-DEC-85
RAMADA	-
SMITH	05-JAN-23
ALLEN	-

Q5) Display name of employee(s) who join the company in 1985 ?

```
SELECT ENAME, HIRE_DATE FROM EMP WHERE EXTRACT(YEAR FROM HIRE_DATE)=1985;
```

ENAME	HIRE_DATE
JONES	01-MAR-85
CLARK	05-DEC-85

Q6) Display name of the employee(s) who join the company this year ?

```
SELECT ENAME, HIRE_DATE FROM EMP WHERE EXTRACT(YEAR FROM HIRE_DATE)=EXTRACT(YEAR FR
```

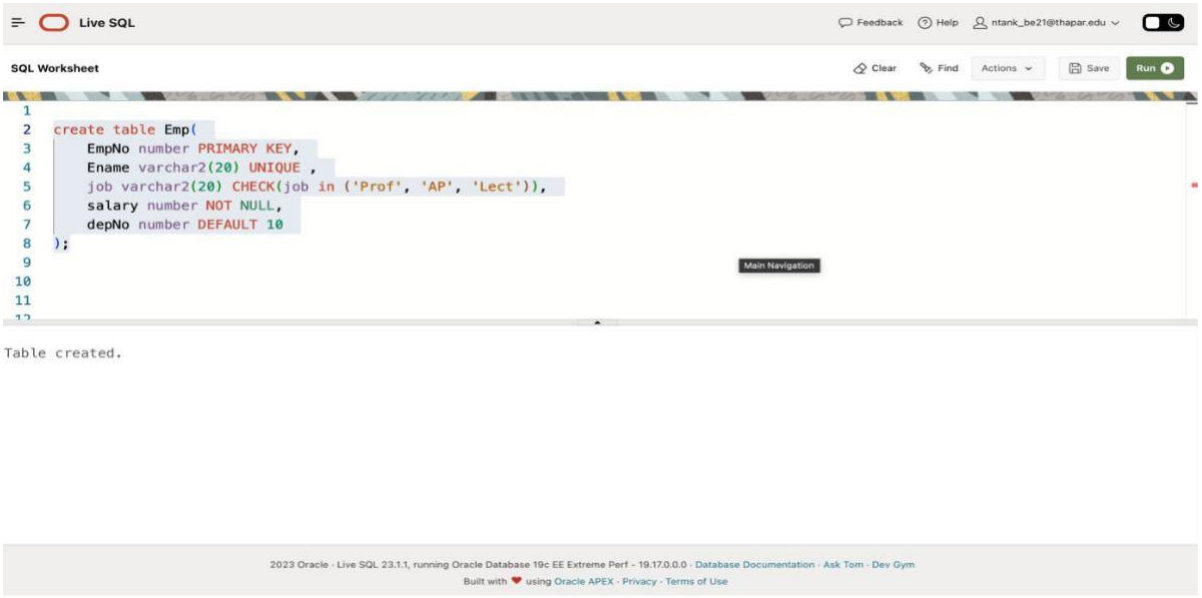
ENAME	HIRE_DATE
SMITH	05-JAN-23

Lab Assignment-4

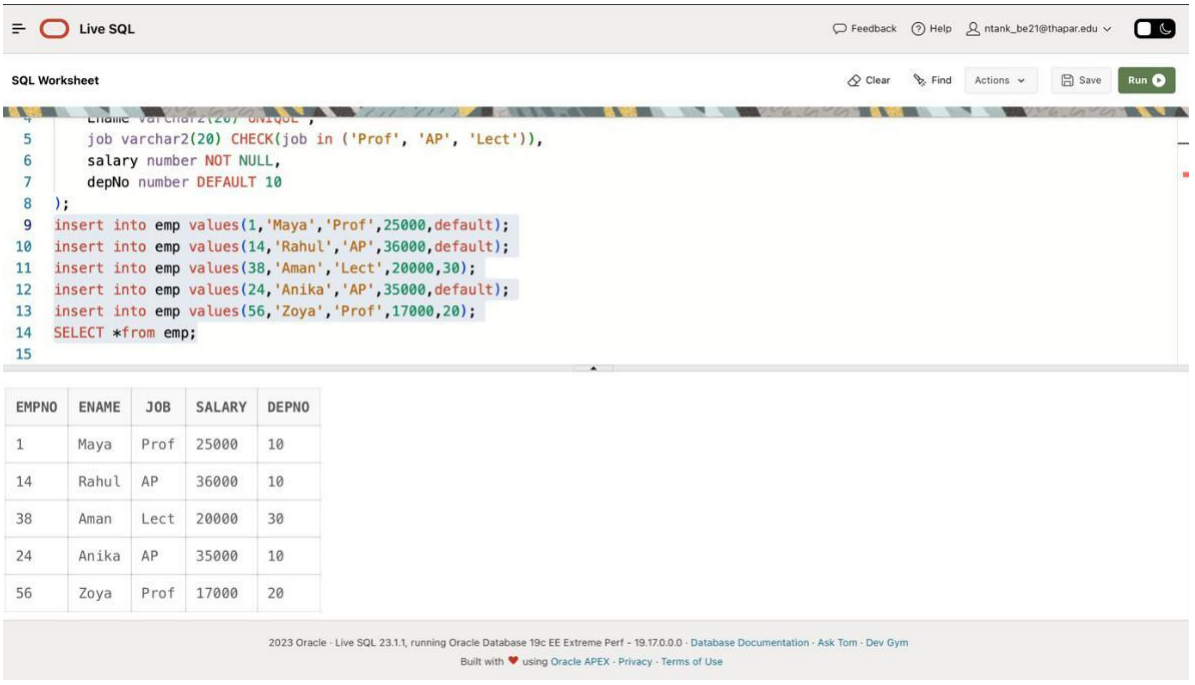
1. Create table emp which has the following attributes (employee table)

   (@empno, ename, job, sal, deptno)

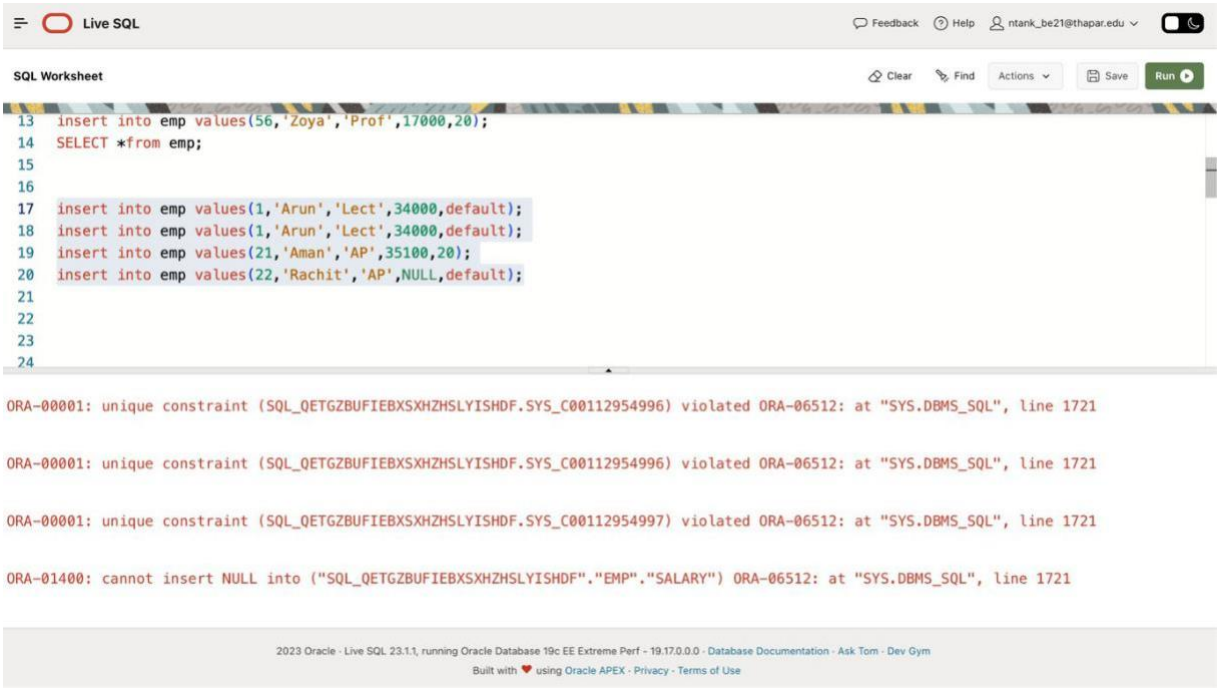
   Where empno is primary key, ename is unique, job in (Prof, AP, and Lect), sal is not NULL, and deptno default is 10.



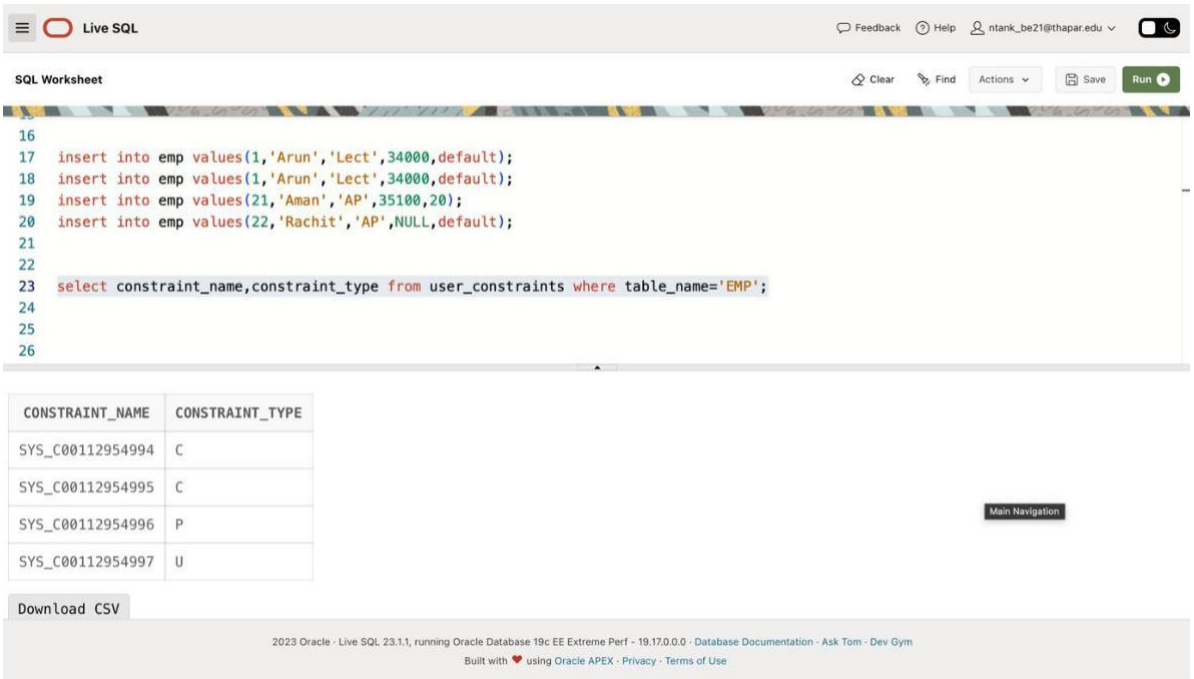
Insert appropriate records



Check error messages in case of violation



and list all the constraint names for given table.



2. Create table book:

   Rno number—PK

   DOI-date
   DOR-date

   DOR>DOI
   Insert appropriate records, check error messages in case of violation and list all the constraint names for given table.

   Create table book:

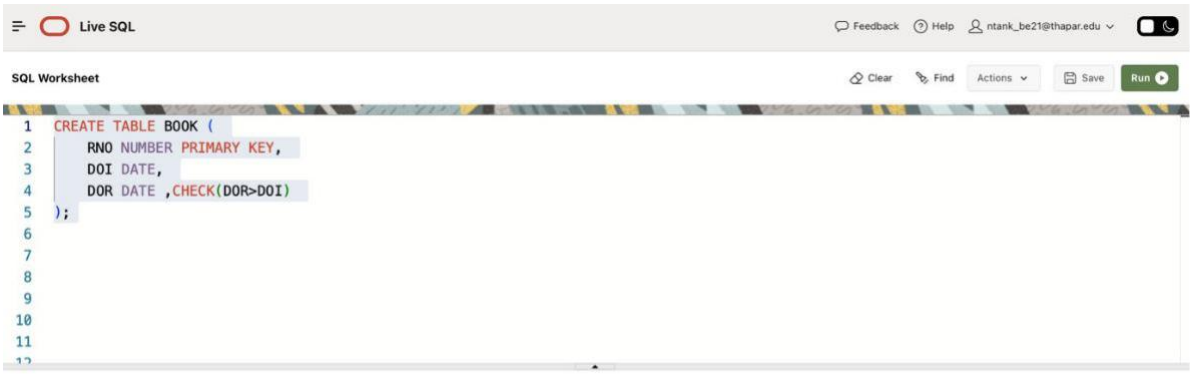
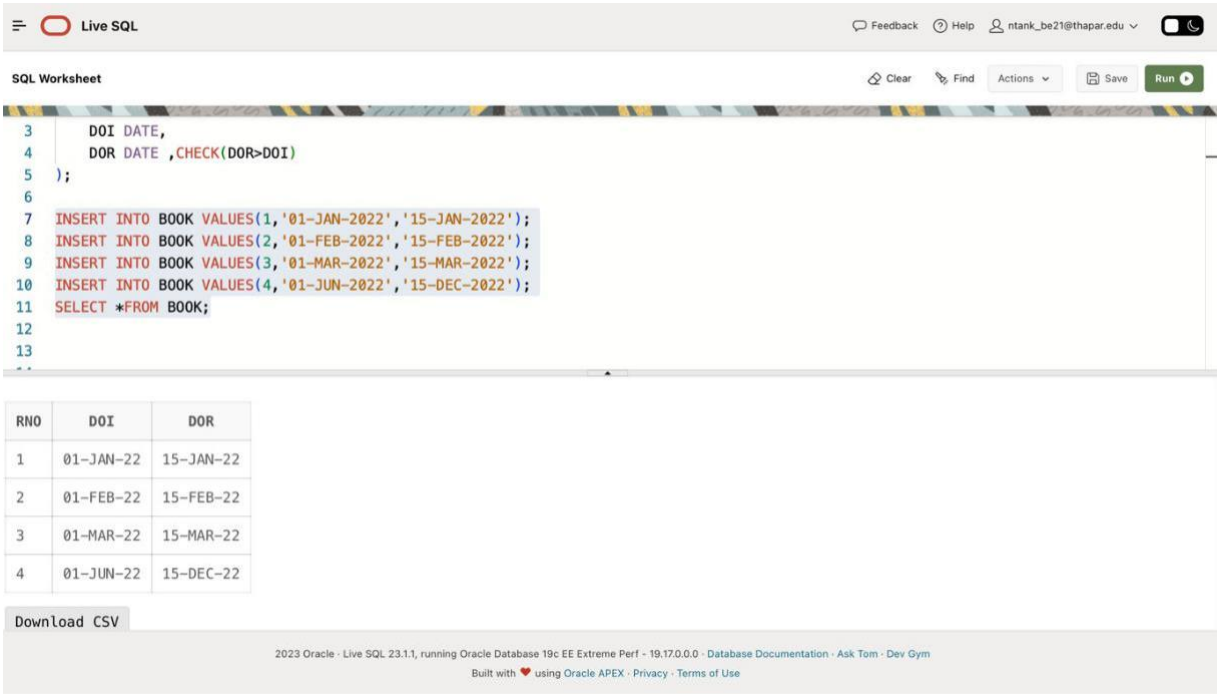
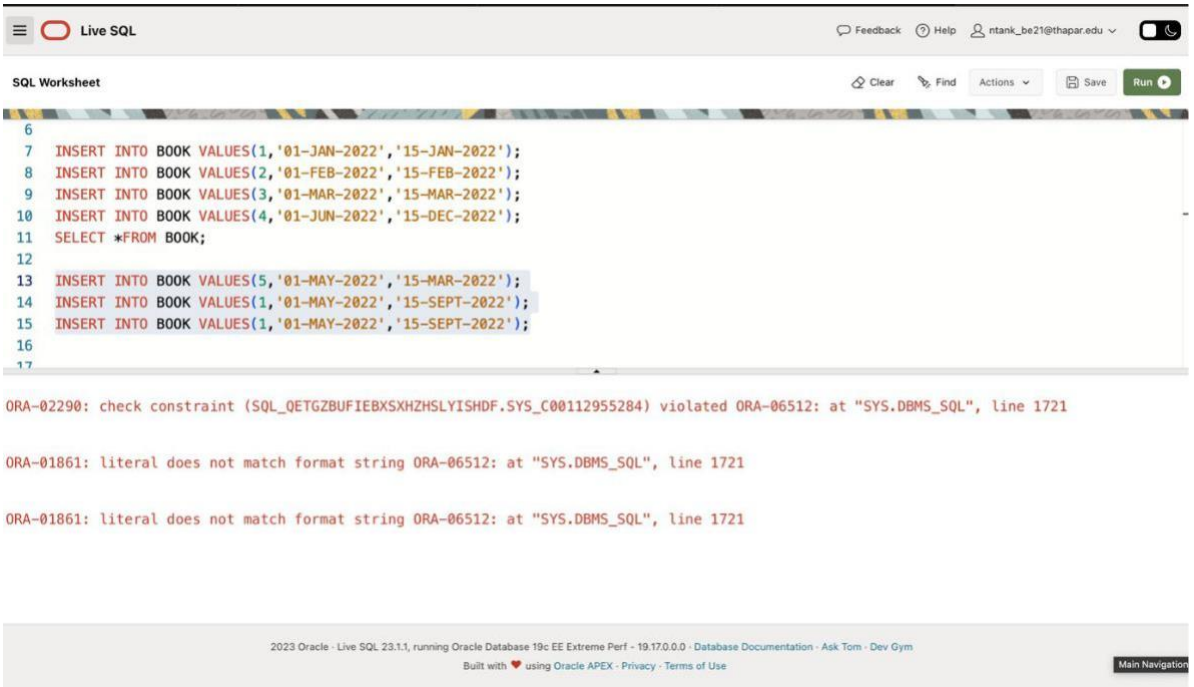


Table created.

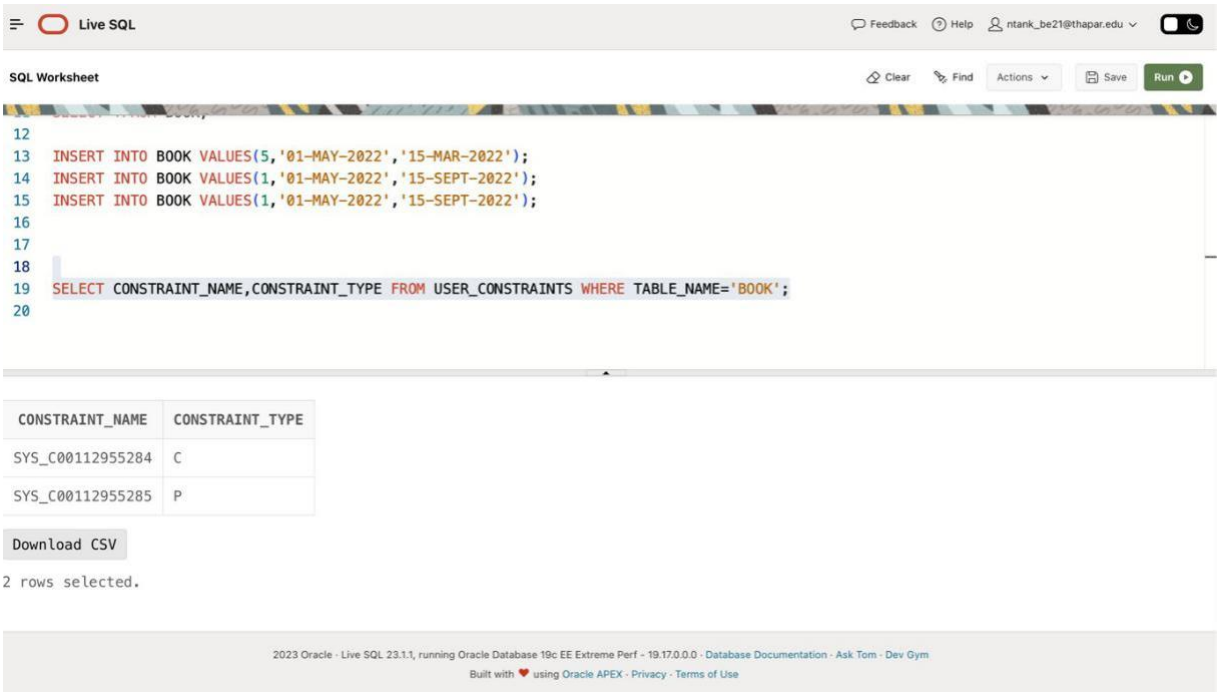
Insert appropriate records:



Check error messages in case of violation :

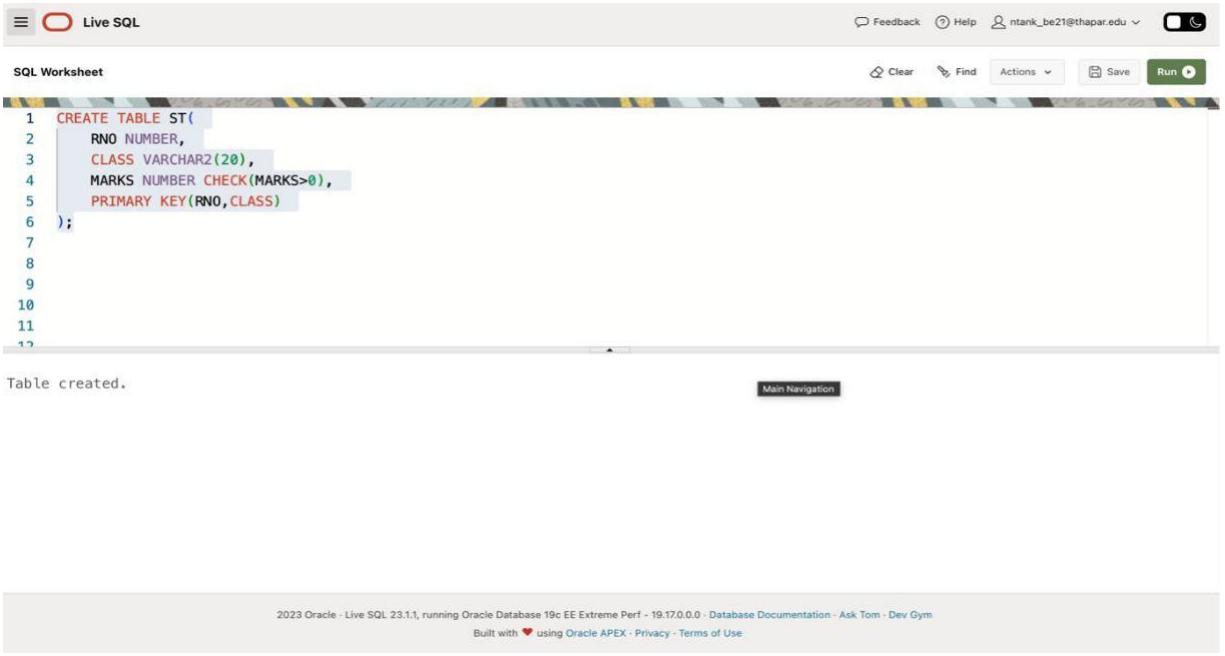


List all the constraint names for given table:

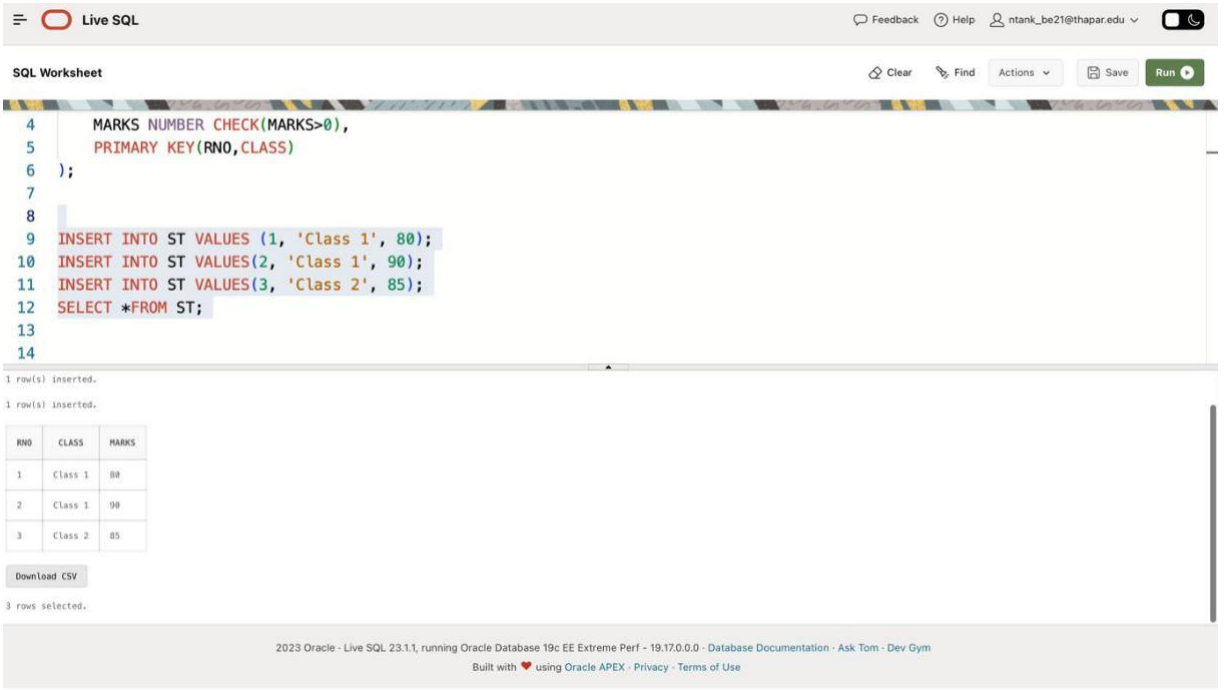


3. Create table st Rno-Number ,Class-Char Marks-Number
   Primary key(rno,class) Marks>0
   Insert appropriate records, check error messages in case of violation and list all the constraint names for given table.

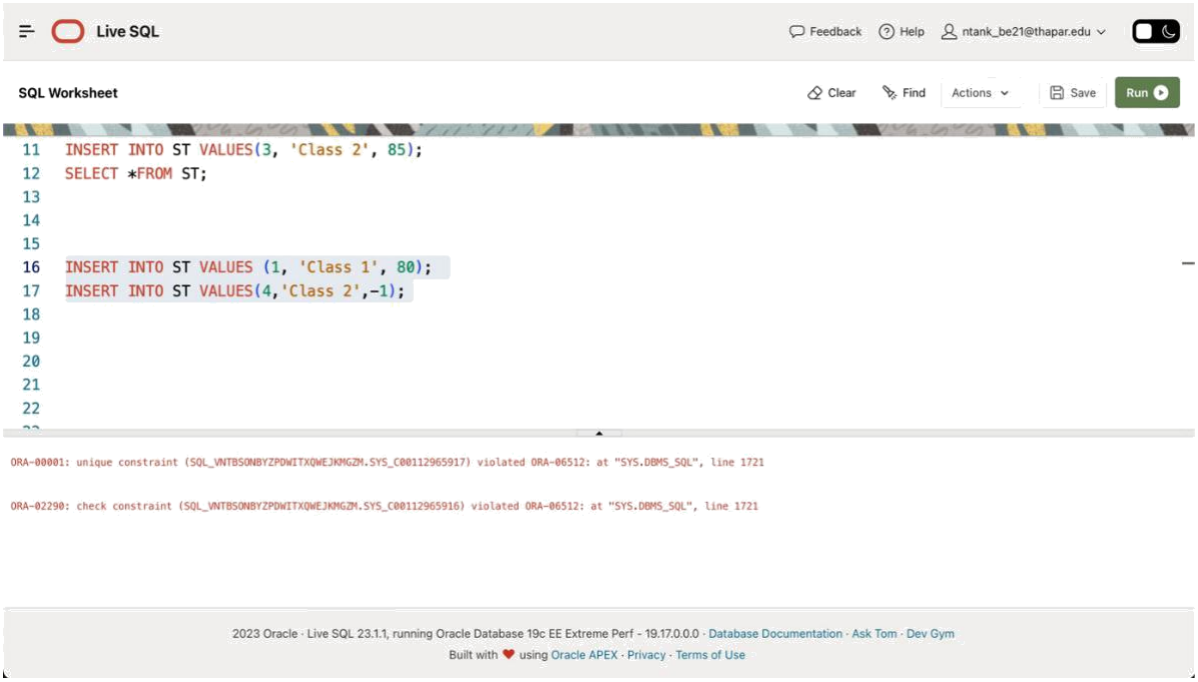
Create table:



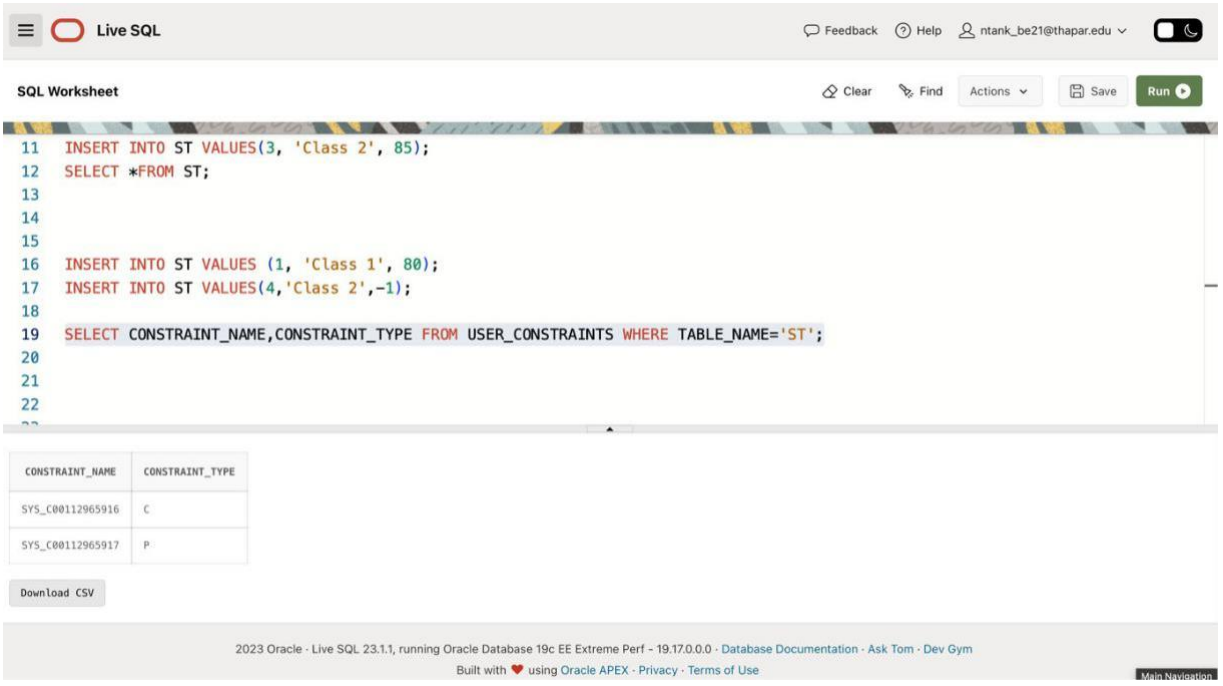
Insert appropriate records:



Check error messages in case of violation:



List all the constraint names for given table:

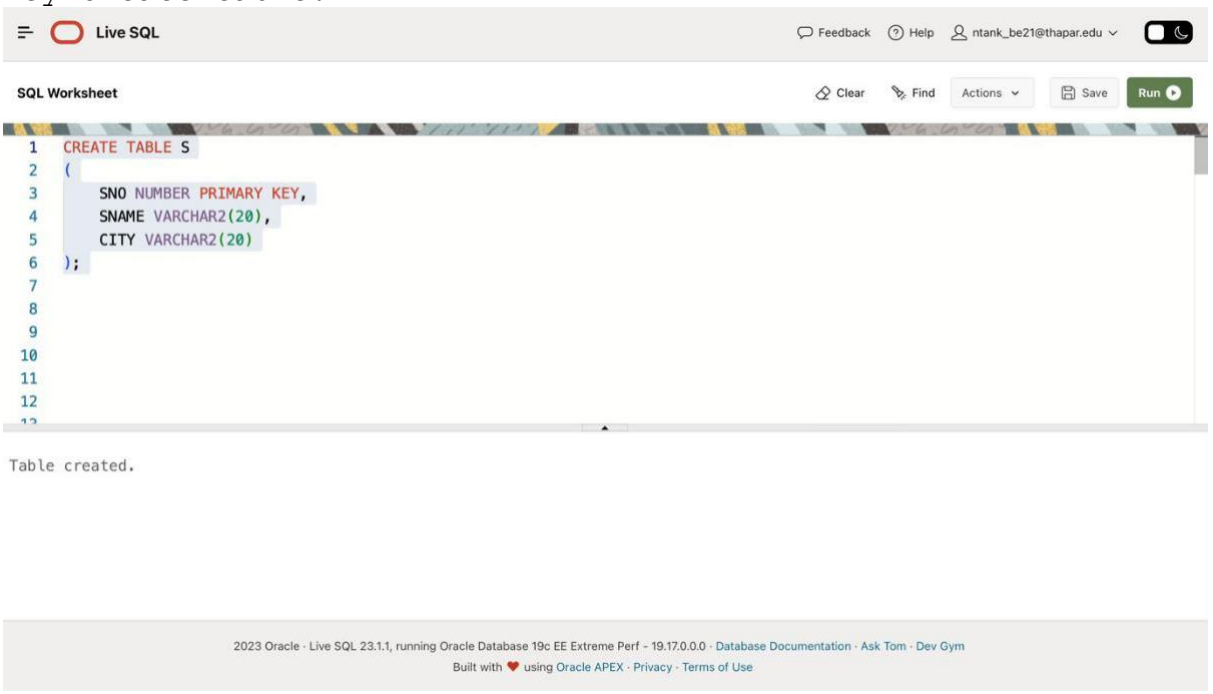


4. Create table S which has the following attributes (Salesperson table)

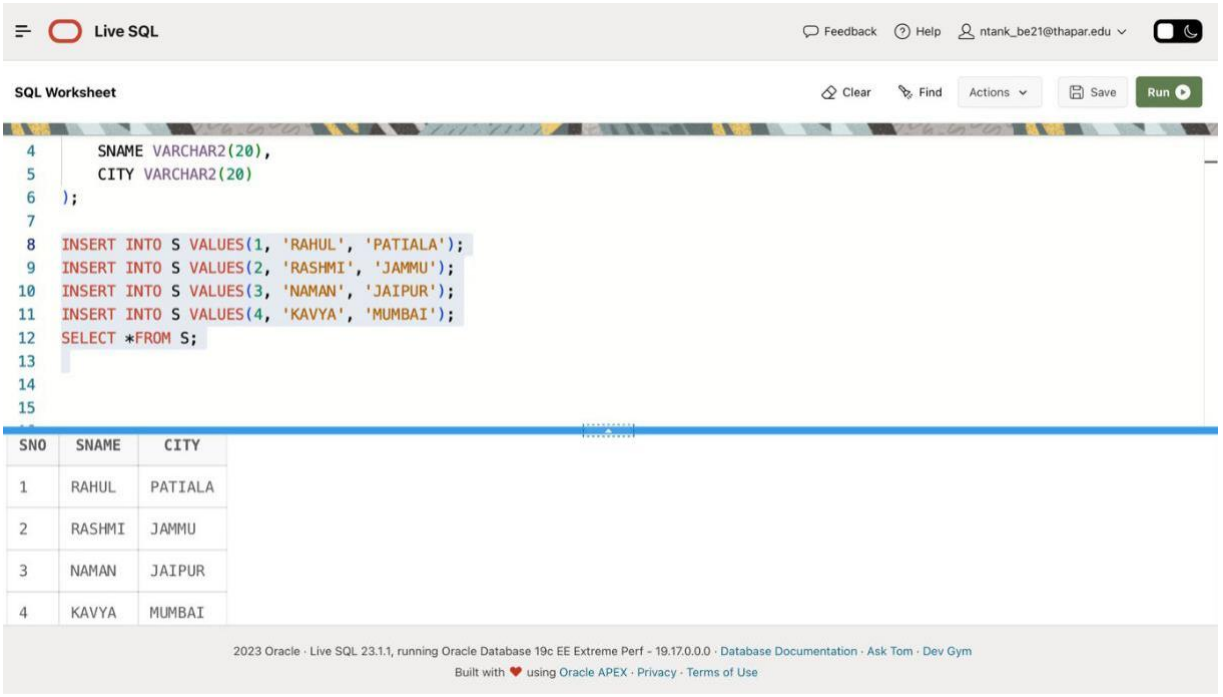
(sno, sname, city)

Where sno is primary

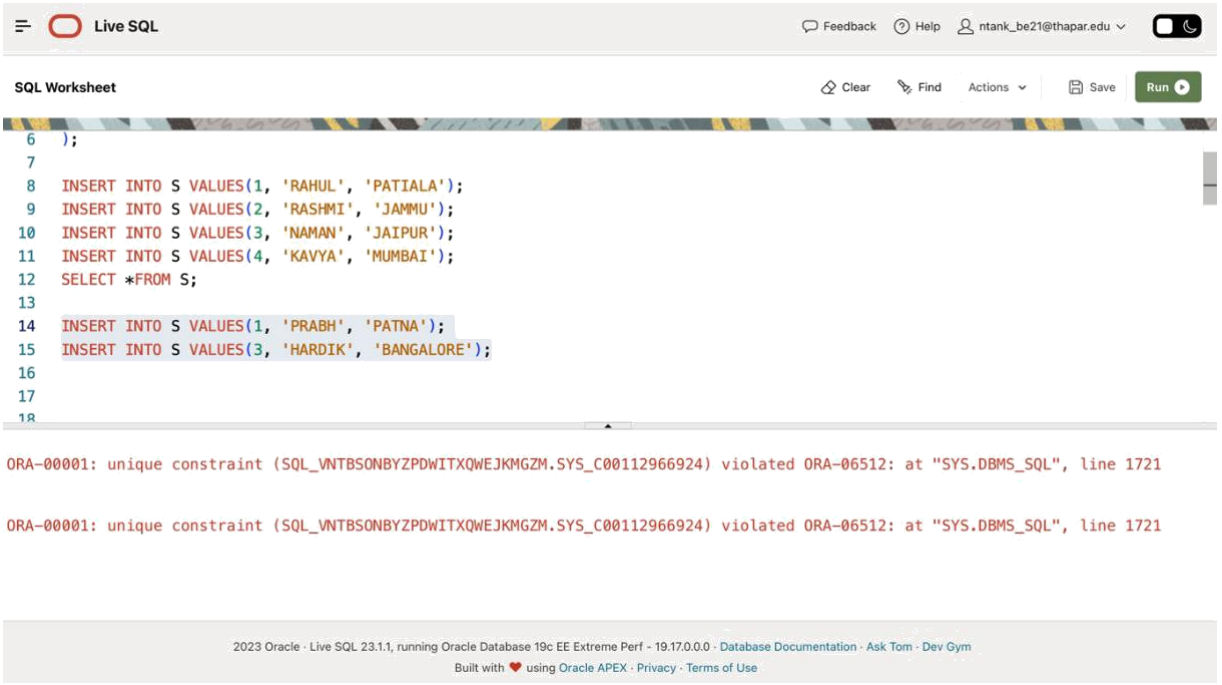
key Create table:



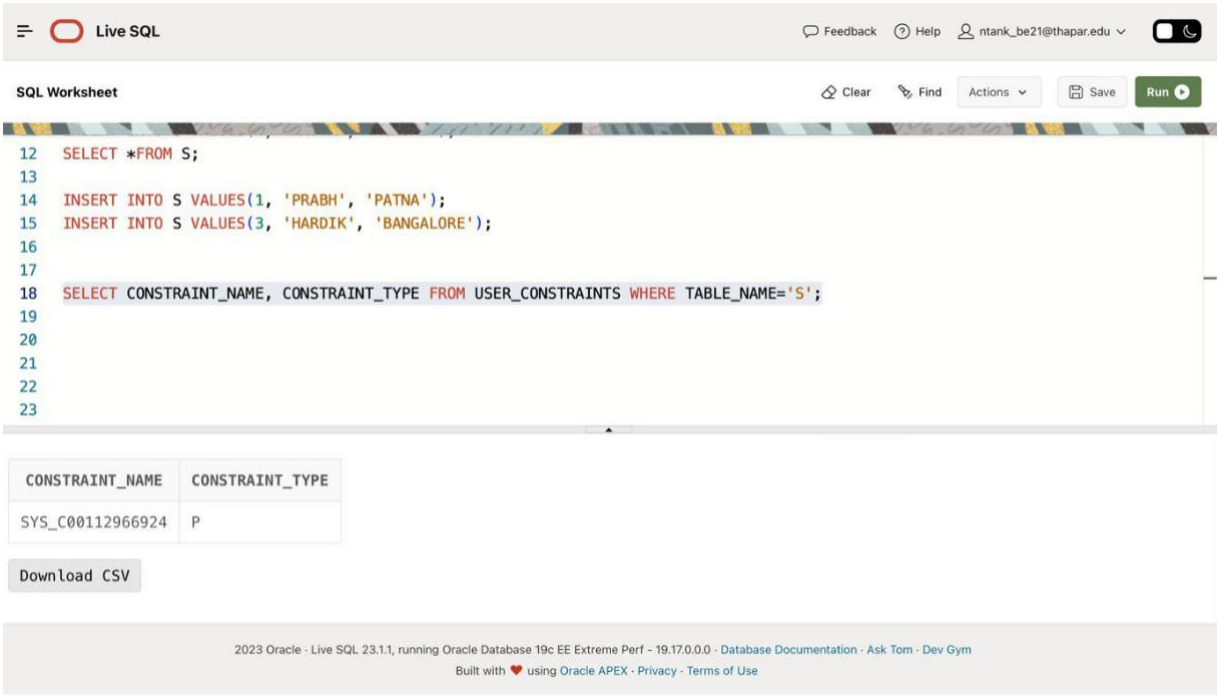
Insert appropriate records:



Check error messages in case of violation:

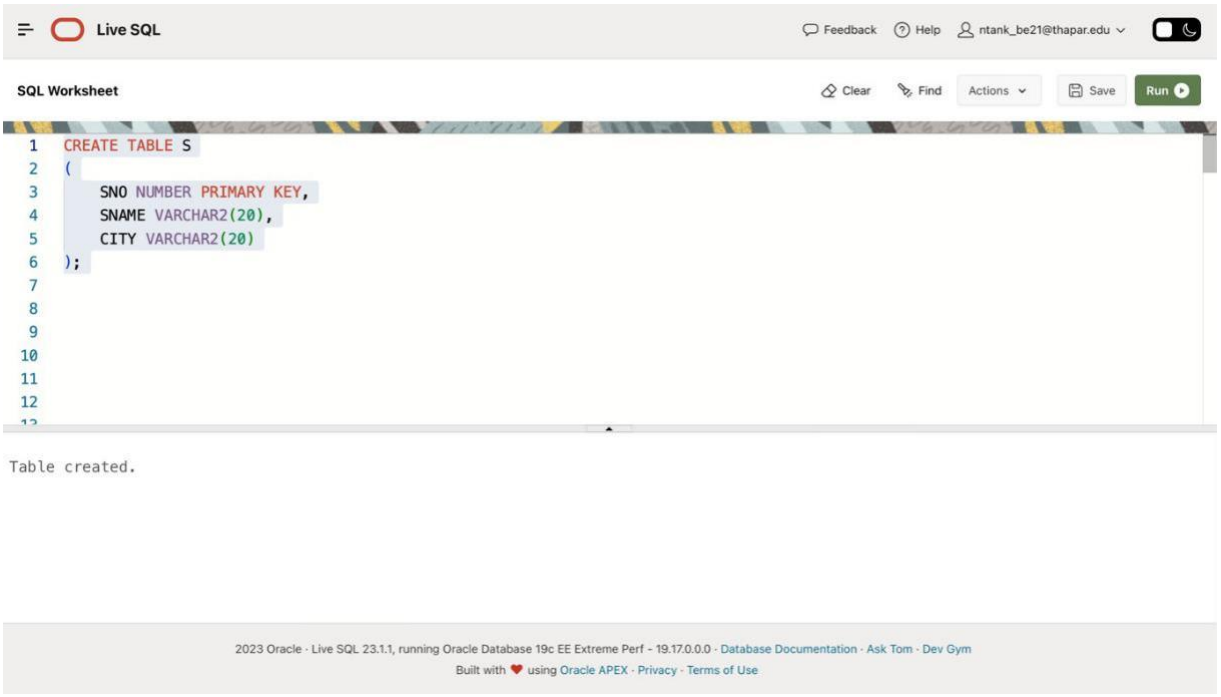


List all the constraint names for given table:

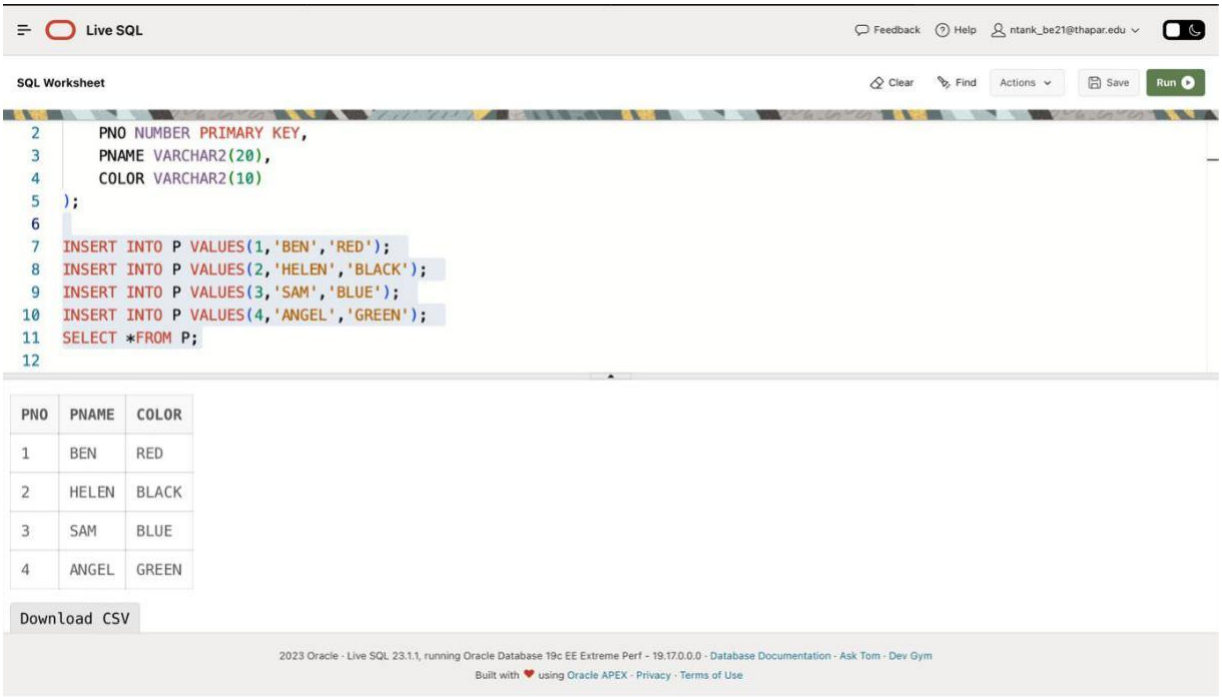


5. Create table P which has the following attributes (Part table)

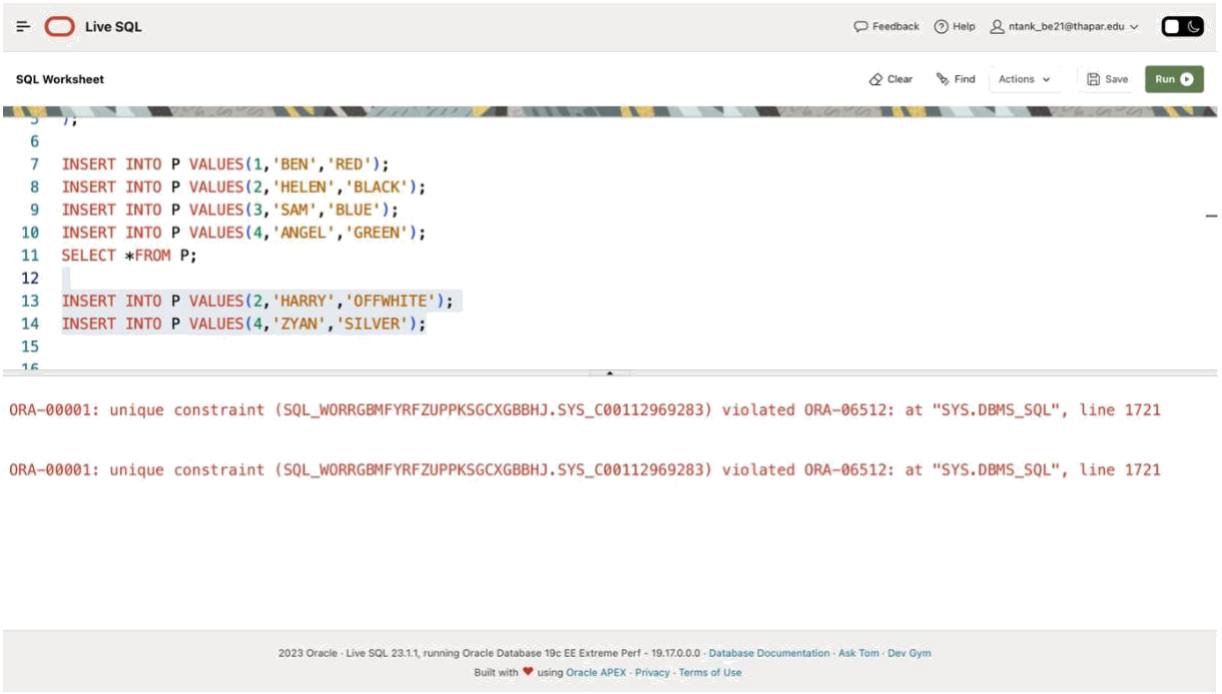
(pno, pname, color) Where pno is primary key Create table:



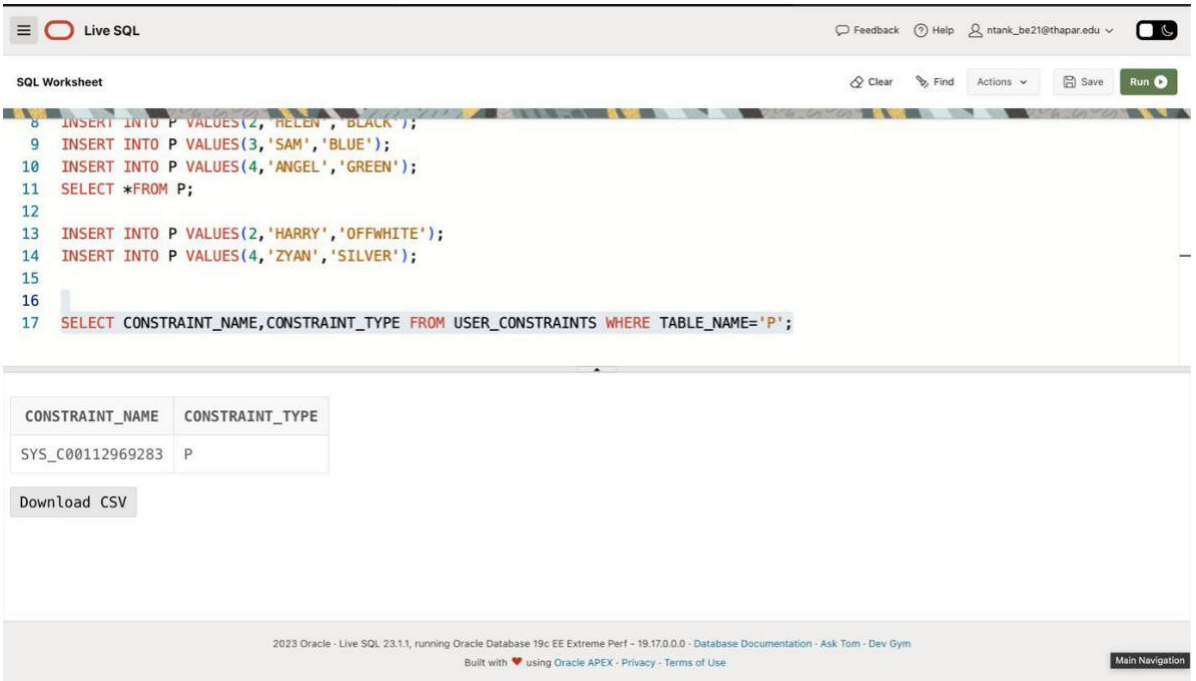
Insert appropriate records:



Check error messages in case of violation:



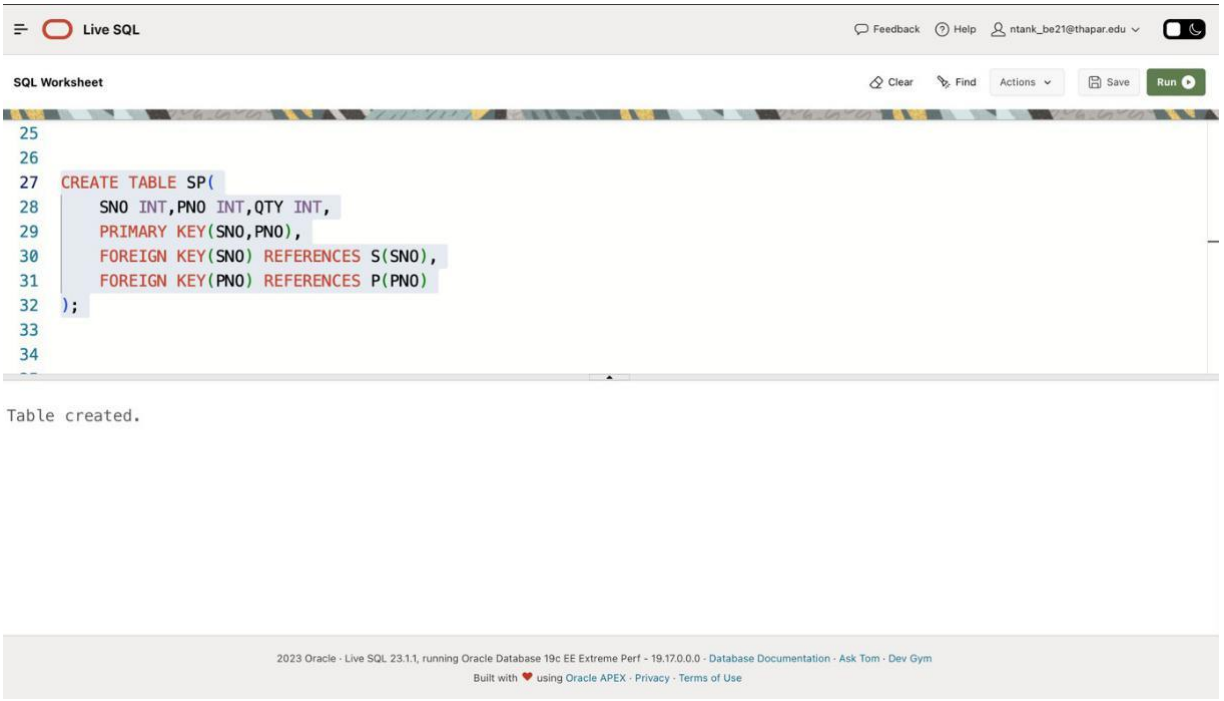
List all the constraint names for given table:



6. *Create table SP which has the following attributes (sno, pno qty)*

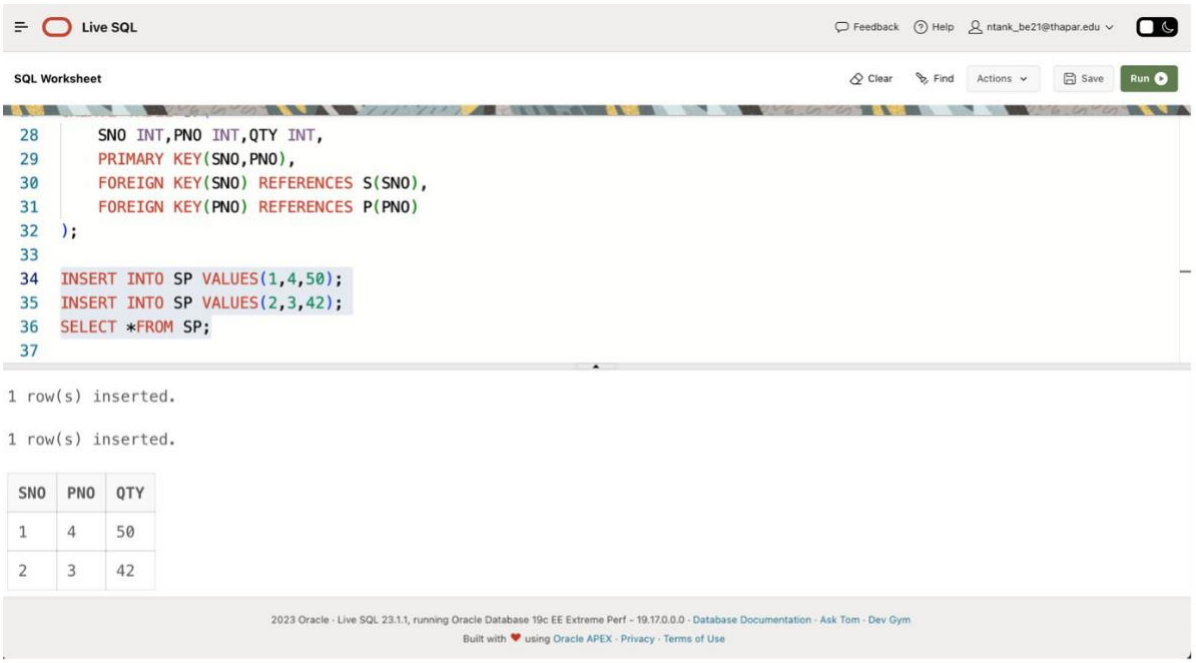
Where combination of (sno, pno) is primary key, also sno and pno are foreign keys

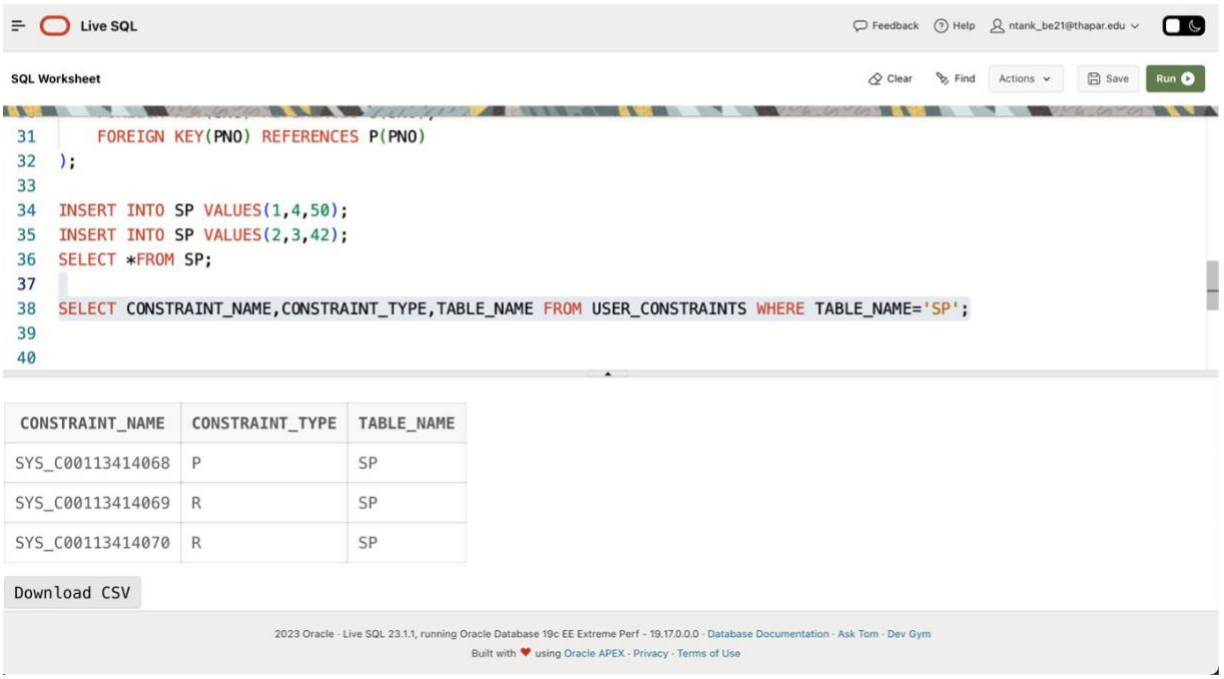
Create table:



Insert appropriate records:

List all the constraint names for given table:





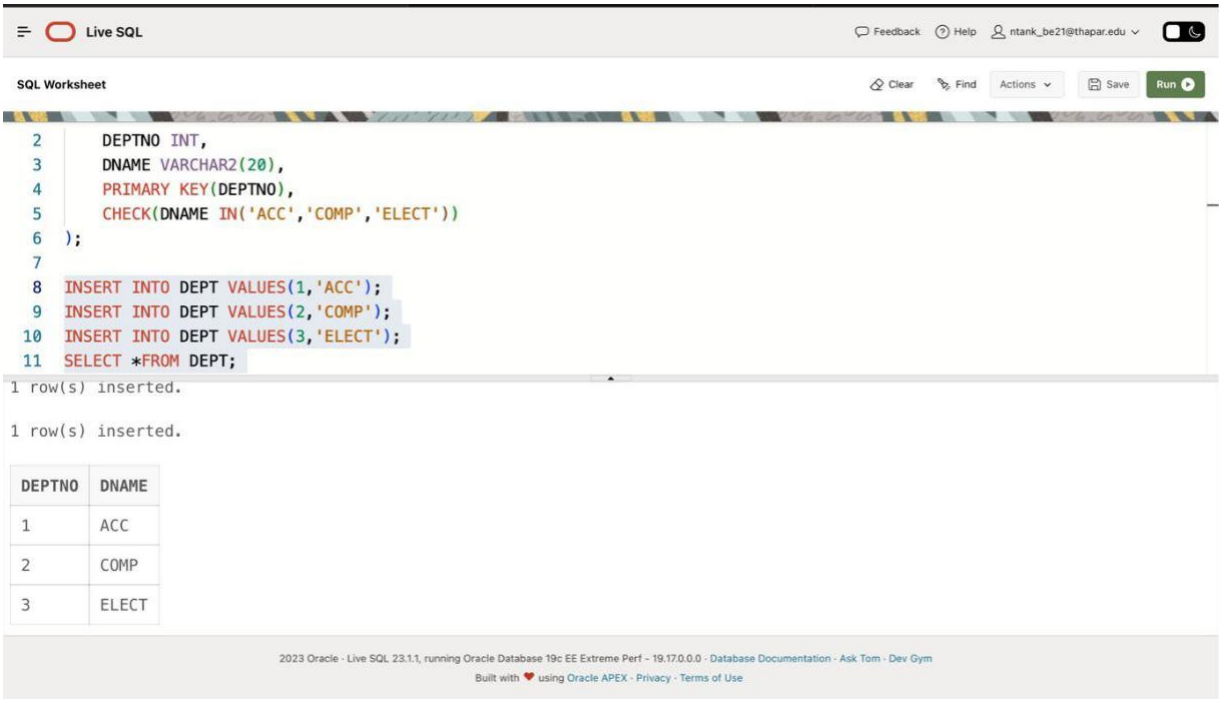
7.Create table dept which has the following attributes (department table)

(deptno, dname)

Where deptno is primary key, dname in (Acc, comp,

elect) Create table:

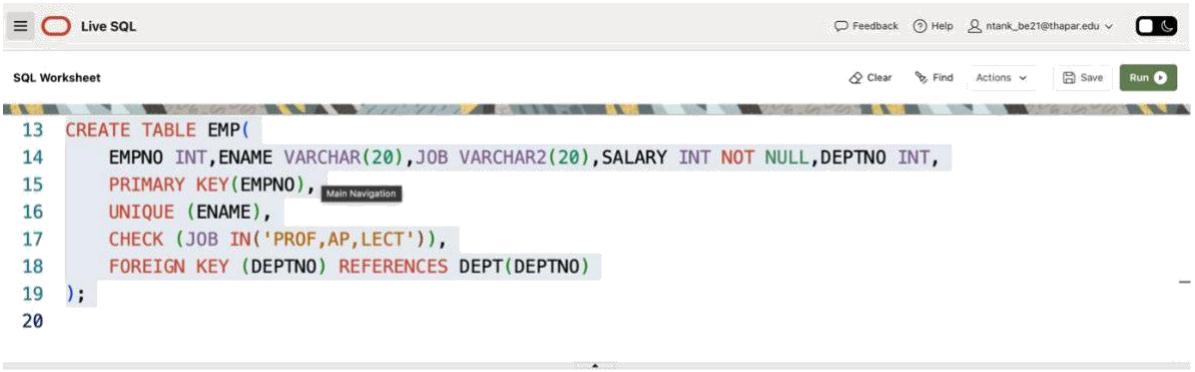
Insert appropriate records:



8. Create table emp which has the following attributes (employee table)

(@empno, ename, job, sal, deptno)

Where empno is primary key, ename is unique, job in (Prof, AP, and Lect), sal is not NULL, and deptno is foreign key




```
Table created.
```

LAB ASSIGNMENT-5

1. Create the following tables and insert some tuples in these tables shown below. Where sid is the primary key for the Sailors table, bid is the primary key for the Boats table and sid and bid are the foreign keys for the Reserves table referencing to the Sailors and Boats table, respectively.

Sailors(sid: integer, sname: string, rating: integer, age: real) Boats(bid: integer, bname: string, color: string) Reserves(sid: integer, bid: integer, day: date)

After inserting the records in these tables, the instances should look like as follows:

Sailors

sid	*sname*	*rating*	*age*
22	Dustin	7	45.0
29	Brutus	1	33.0
31	Lubber	8	55.5
32	Andy	8	25.5
58	Rusty	10	35.0
64	Horatio	7	35.0
71	Zorba	10	16.0
74	Horatio	9	35.0
85	Art	3	25.5
95	Bob	3	63.5

Reserves

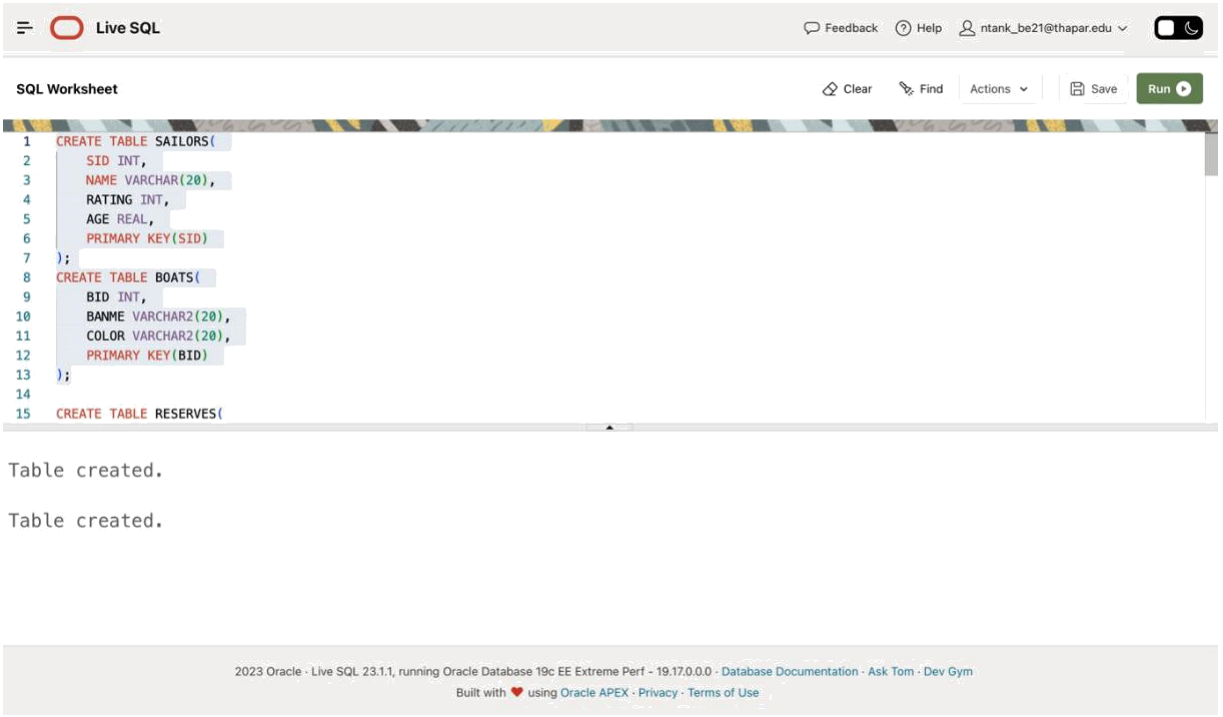
sid	*bid*	*day*
22	101	10/10/98
22	102	10/10/98
22	103	10/8/98
22	104	10/7/98
31	102	11/10/98
31	103	11/6/98
31	104	11/12/98
64	101	9/5/98
64	102	9/8/98
74	103	9/8/98

Boats

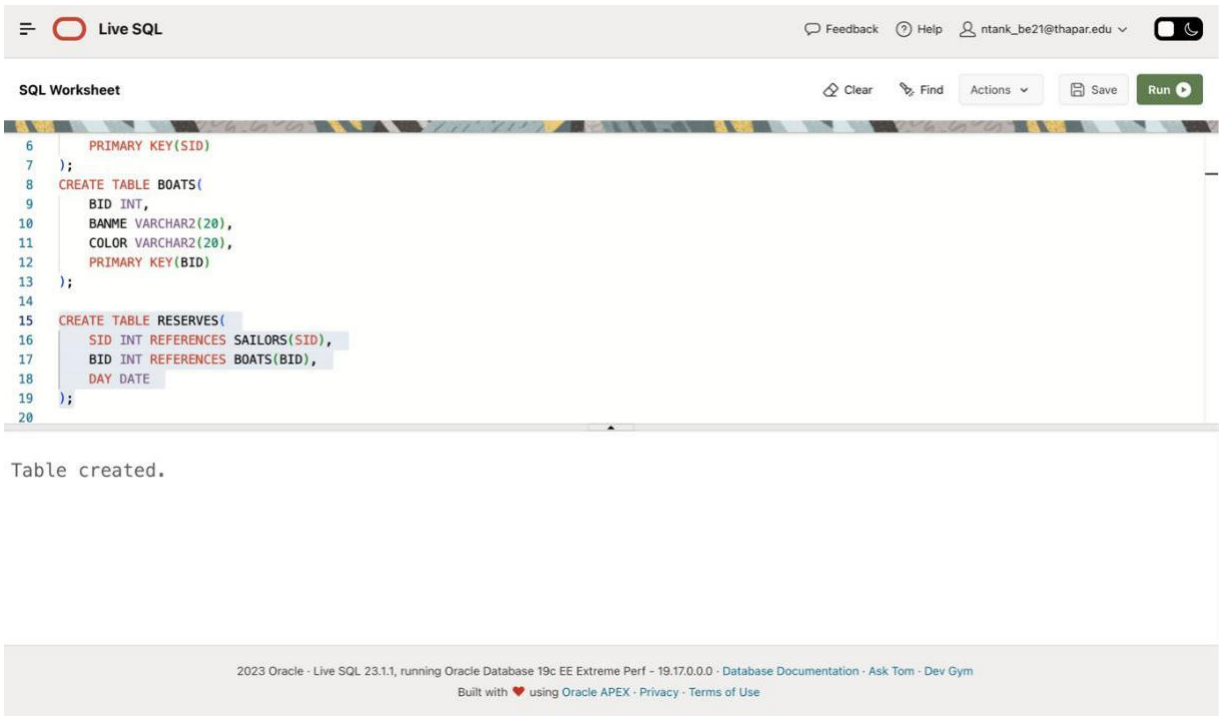
bid	*bname*	*color*
101	Interlake	blue
102	Interlake	red
103	Clipper	green
104	Marine	red

>> Sailors(sid: integer, sname: string, rating: integer, age: real)

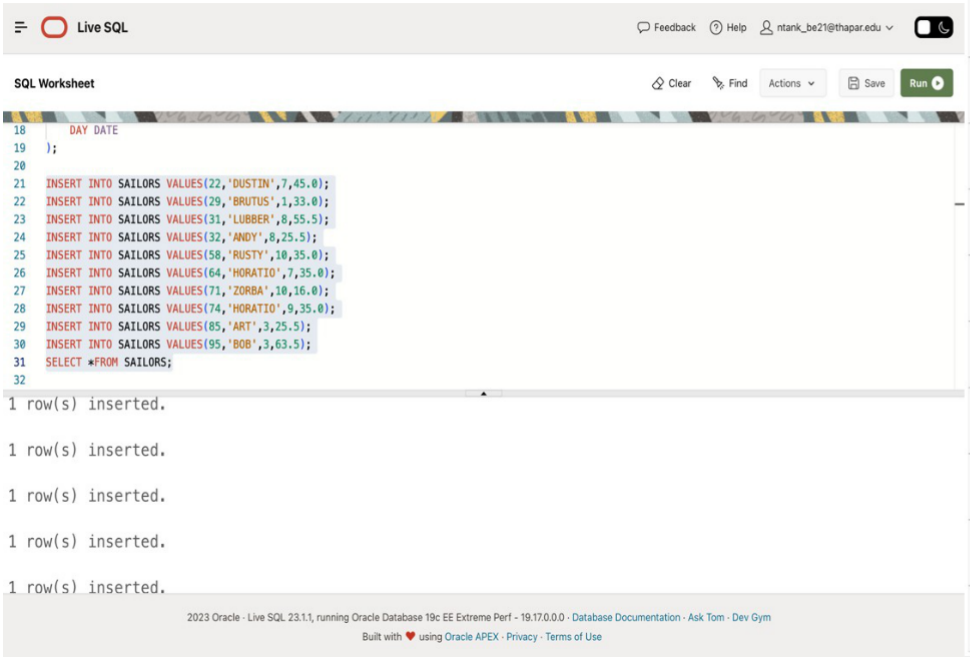
>> Boats(bid: integer, bname: string, color: string)



>> Reserves(sid: integer, bid: integer, day: date)

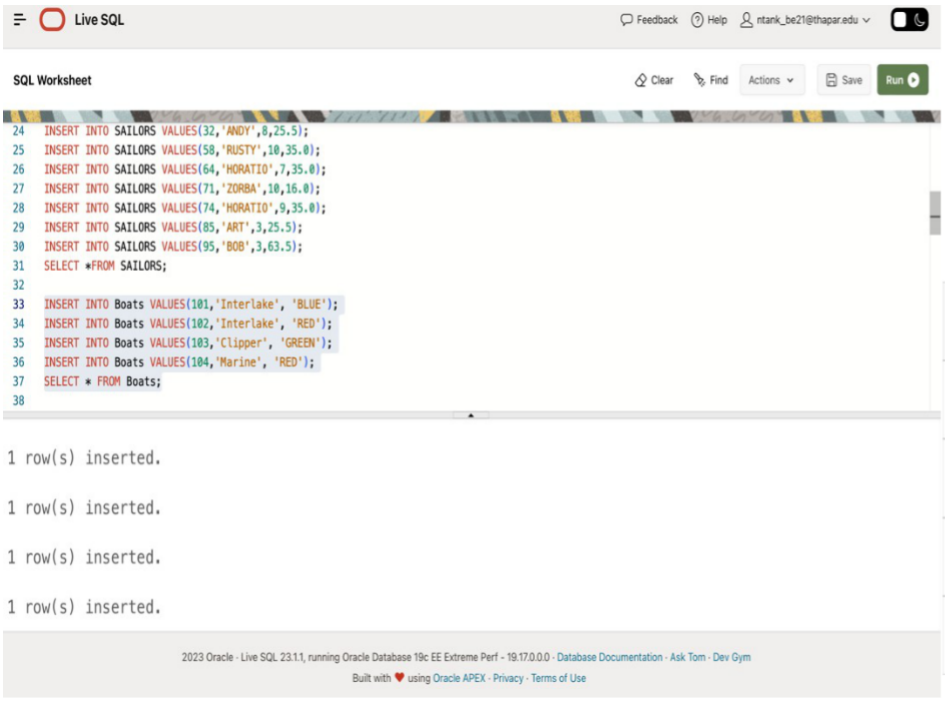


>> Inserting Record in Sailors table:



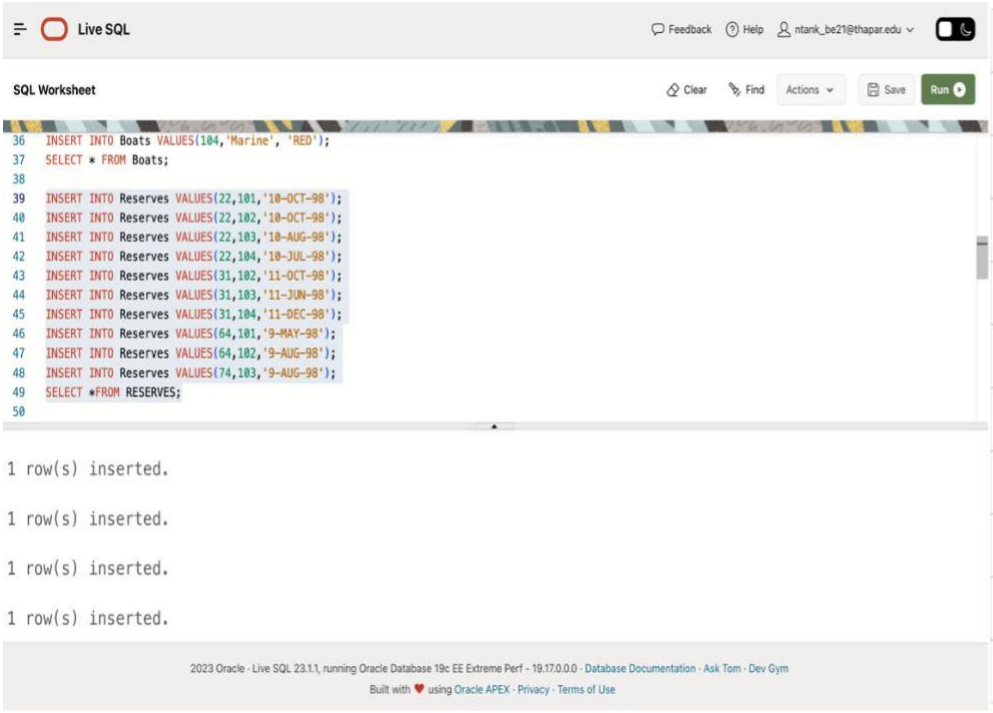
SID	NAME	RATING	AGE
22	DUSTIN	7	45
29	BRUTUS	1	33
31	LUBBER	8	55.5
32	ANDY	8	25.5
58	RUSTY	10	35
64	HORATIO	7	35
71	ZORBA	10	16
74	HORATIO	9	35
85	ART	3	25.5
95	BOB	3	63.5

>> Inserting Record in Reserves table:



BID	BANME	COLOR
101	Interlake	BLUE
102	Interlake	RED
103	Clipper	GREEN
104	Marine	RED

>>Inserting Record in Boats table:



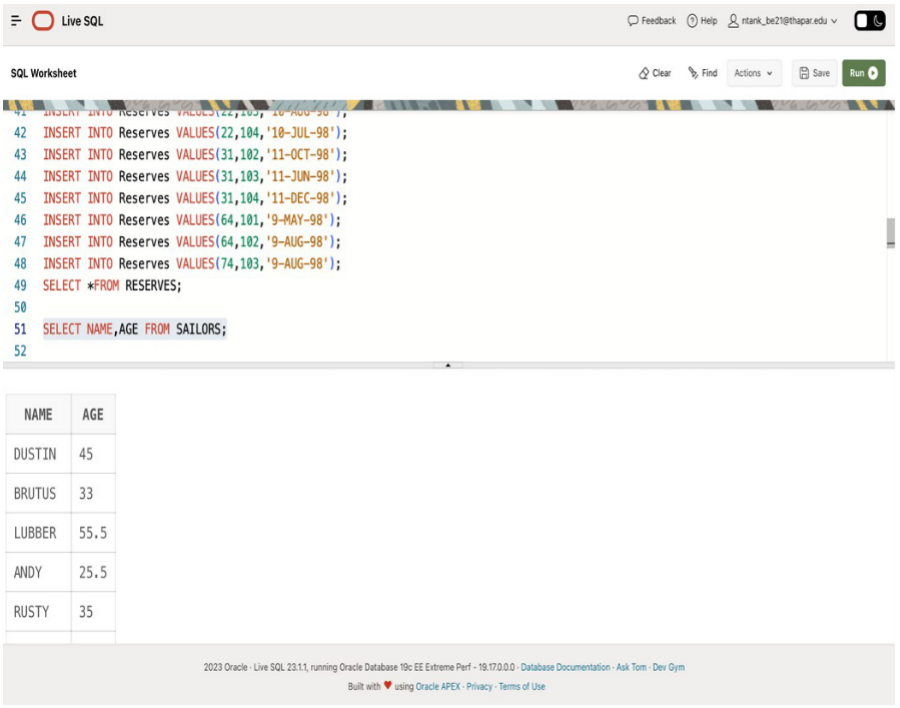
SID	BID	DAY
22	101	10-OCT-98
22	102	10-OCT-98
22	103	10-AUG-98
22	104	10-JUL-98
31	102	11-OCT-98
31	103	11-JUN-98
31	104	11-DEC-98
64	101	09-MAY-98
64	102	09-AUG-98
74	103	09-AUG-98

2. Write SQL command for the following:

1. i) Show the names and ages of all sailors.
2. ii) Show the details of the boats which are red and blue in color.
3. iii) Find the oldest and youngest sailors' age.
4. iv) Find the ages of sailors whose name begins and ends with B and has at leastthree characters.
5. v) Show the average rating of the sailors.
6. vi) Find all sailors with a rating above 7.
7. vii) Find the number of boats reserved by the sailor namedHoratio.
8. viii) Find the colors of boats reserved by Lubber.
9. ix) Show the details of the sailors who have reserved the boat with bid 102.
10. x) Find the sid of sailors who have reserved green boats.
11. xi) Find the names of sailors who have reserved boat number 103.
12. xii) Find the sids and names of sailors who have reserved a red boat.
13. xiii) Find the names of the sailors who have reserved a green or a blue boat.
14. xiv) Find the names of sailors who have reserved both a red and a green boat.
15. xv) Find the names of sailors who have reserved at least one boat.

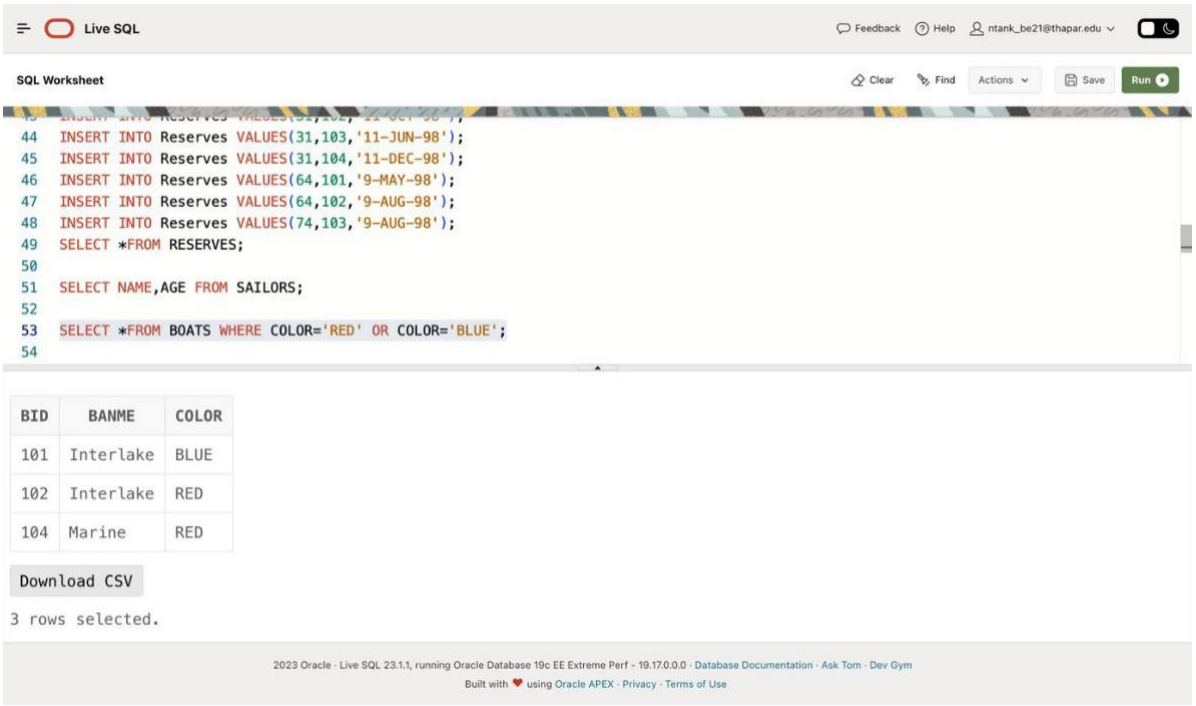
Solution:

i) Show the names and ages of all sailors:

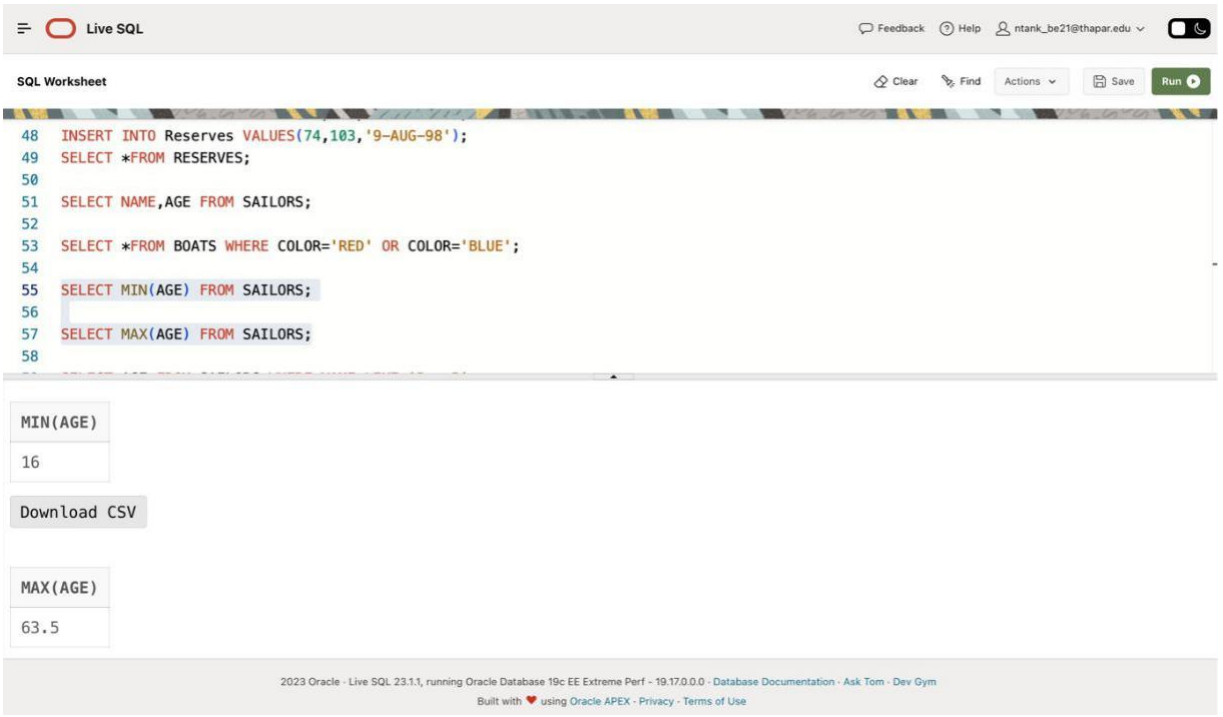


NAME	AGE
DUSTIN	45
BRUTUS	33
LUBBER	55.5
ANDY	25.5
RUSTY	35
HORATIO	35
ZORBA	16
HORATIO	35
ART	25.5
BOB	63.5

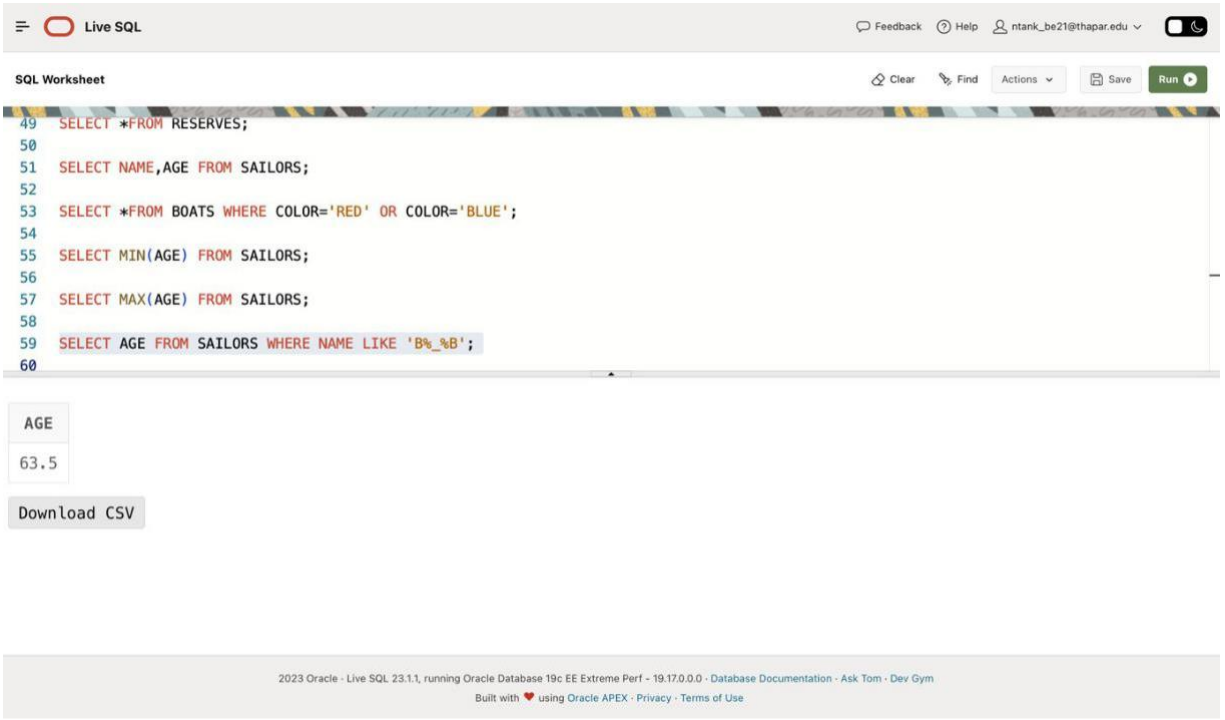
ii) Show the details of the boats which are red and blue in color.



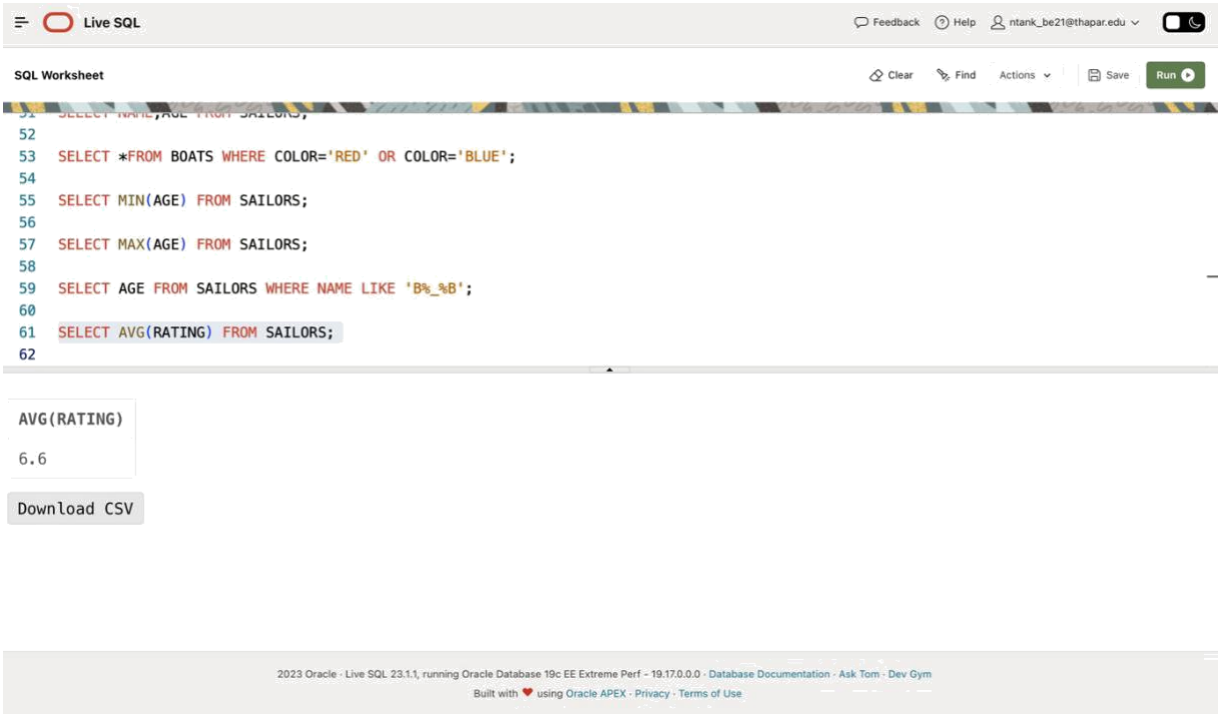
iii) Find the oldest and youngest sailors' age.



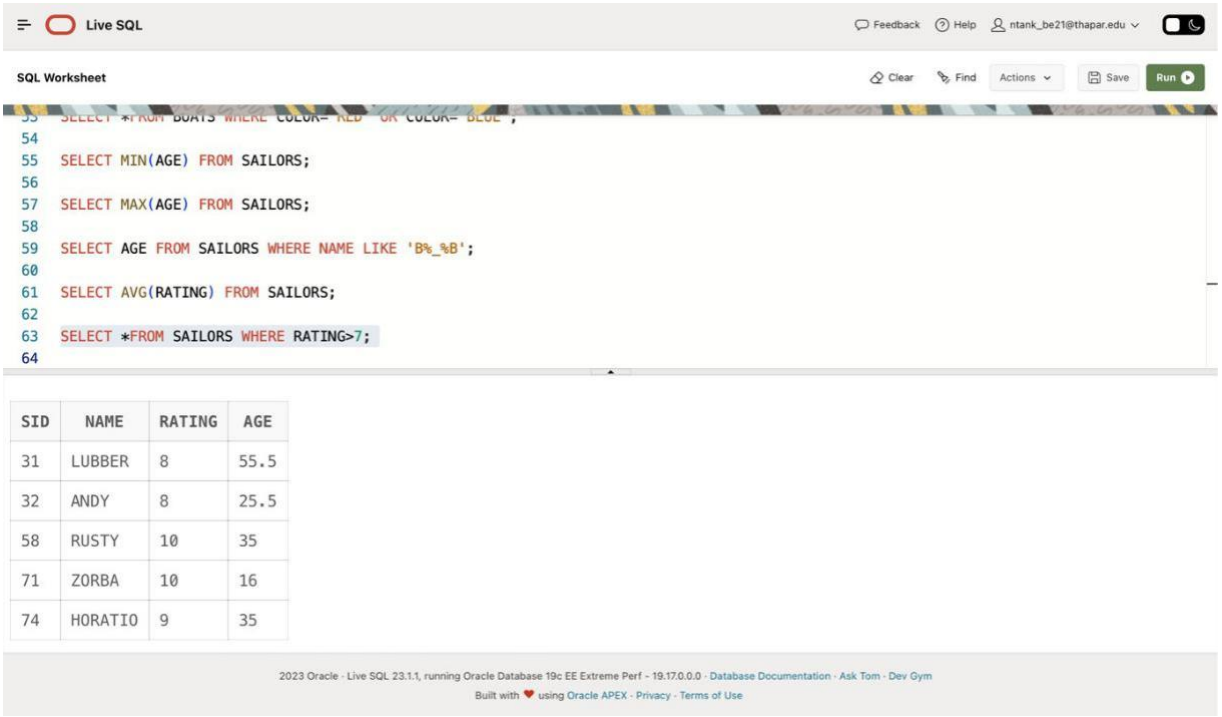
iv) Find the ages of sailors whose name begins and ends with B and has at leastthree characters.



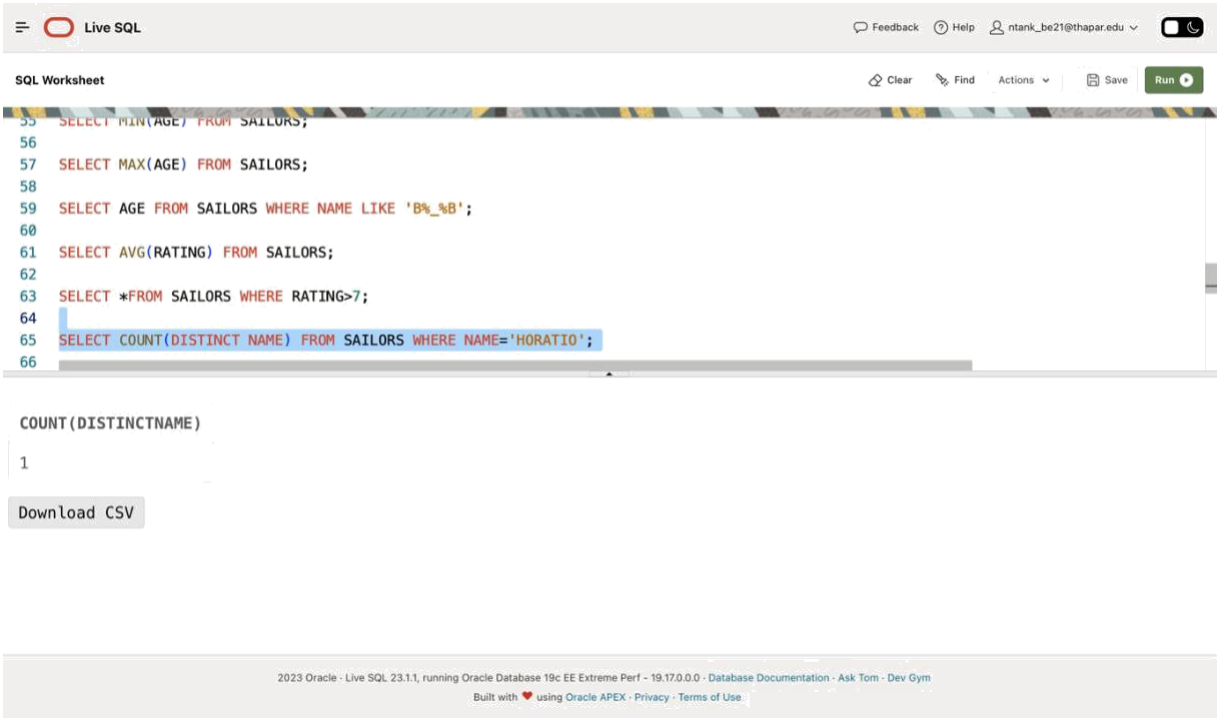
v) Show the average rating of the sailors.



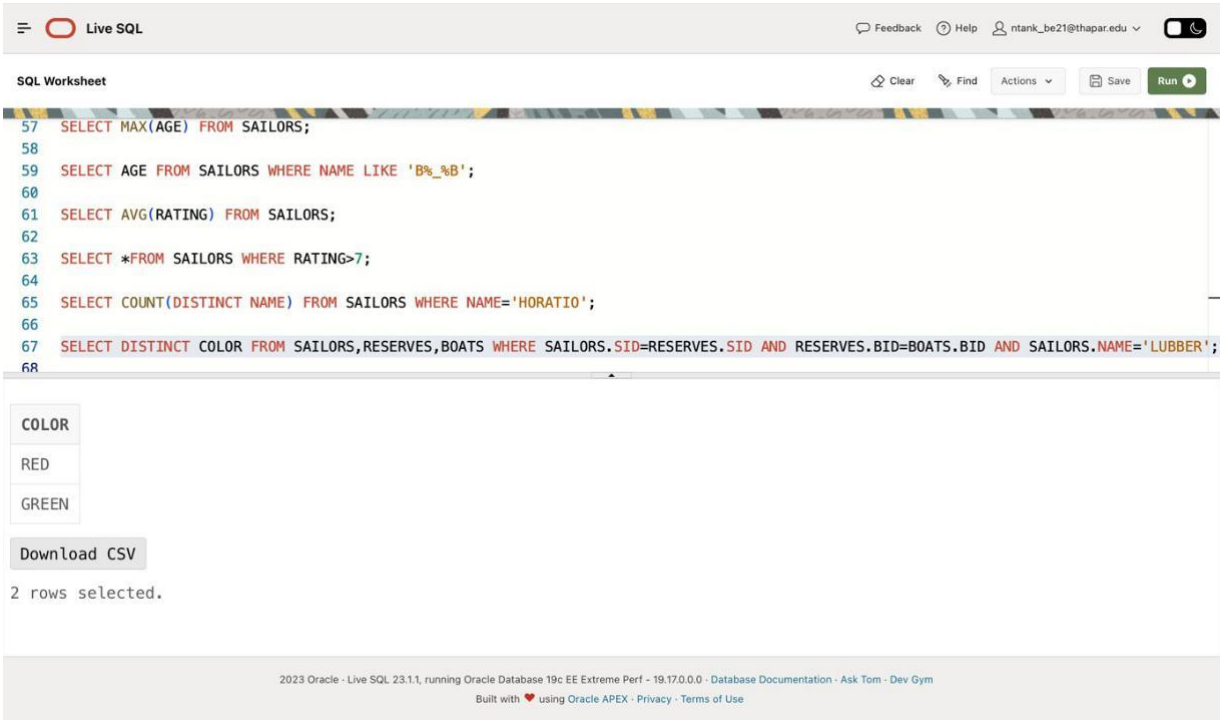
vi) Find all sailors with a rating above 7.



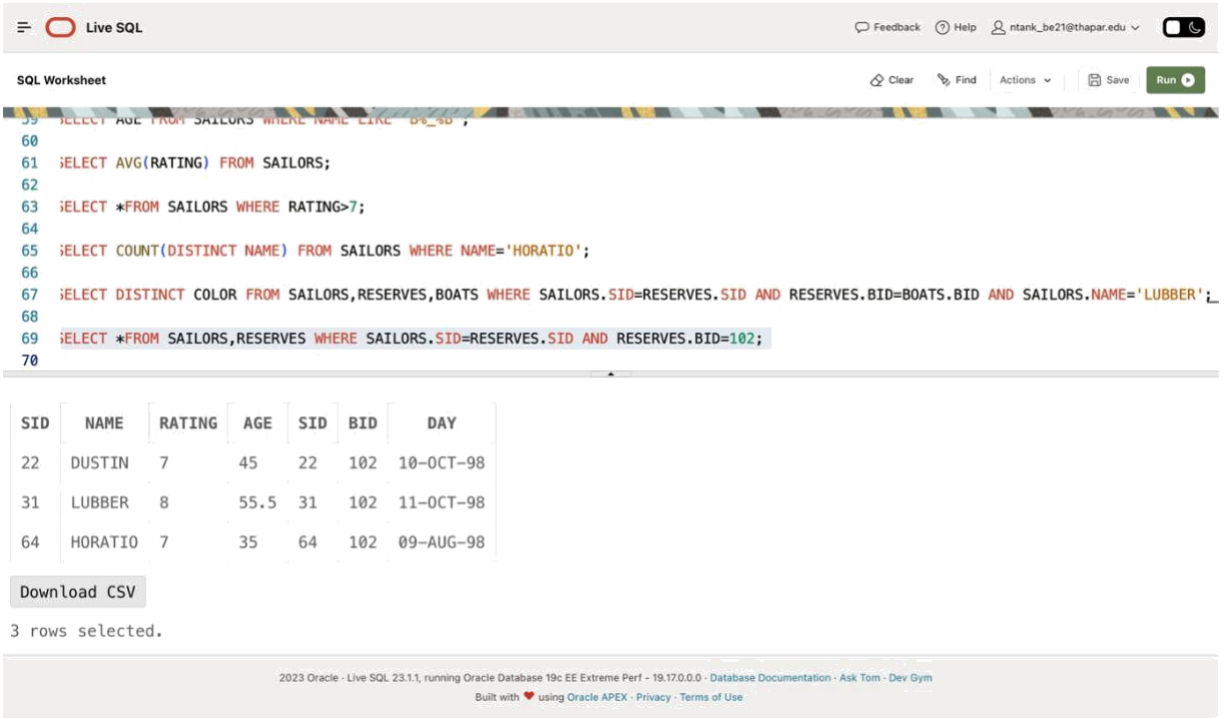
vii) Find the number of boats reserved by the sailor namedHoratio.



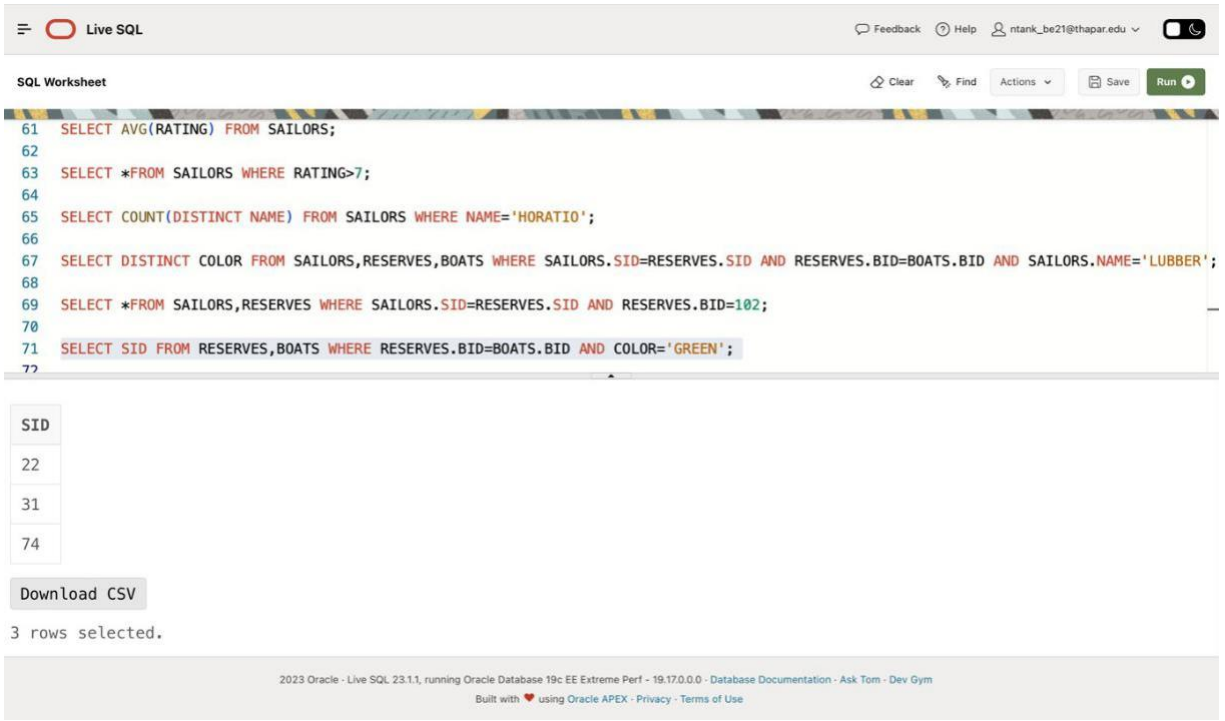
viii) Find the colors of boats reserved by Lubber.



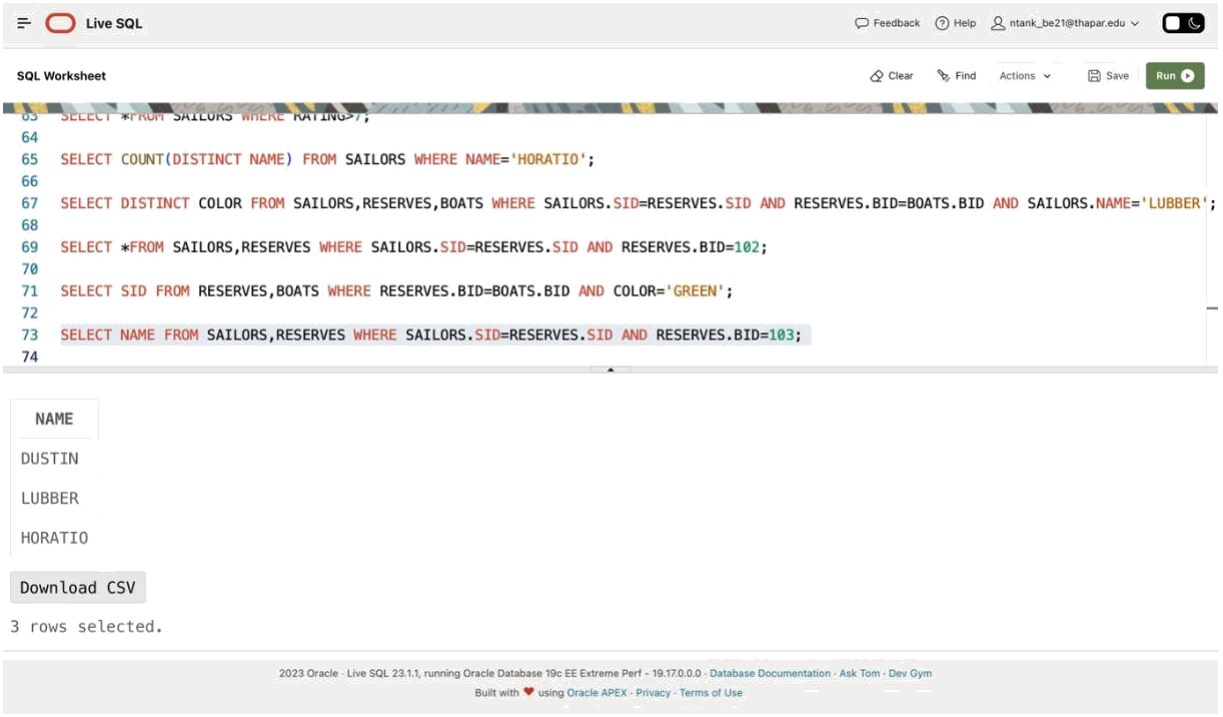
ix) Show the details of the sailors who have reserved the boat with bid 102.



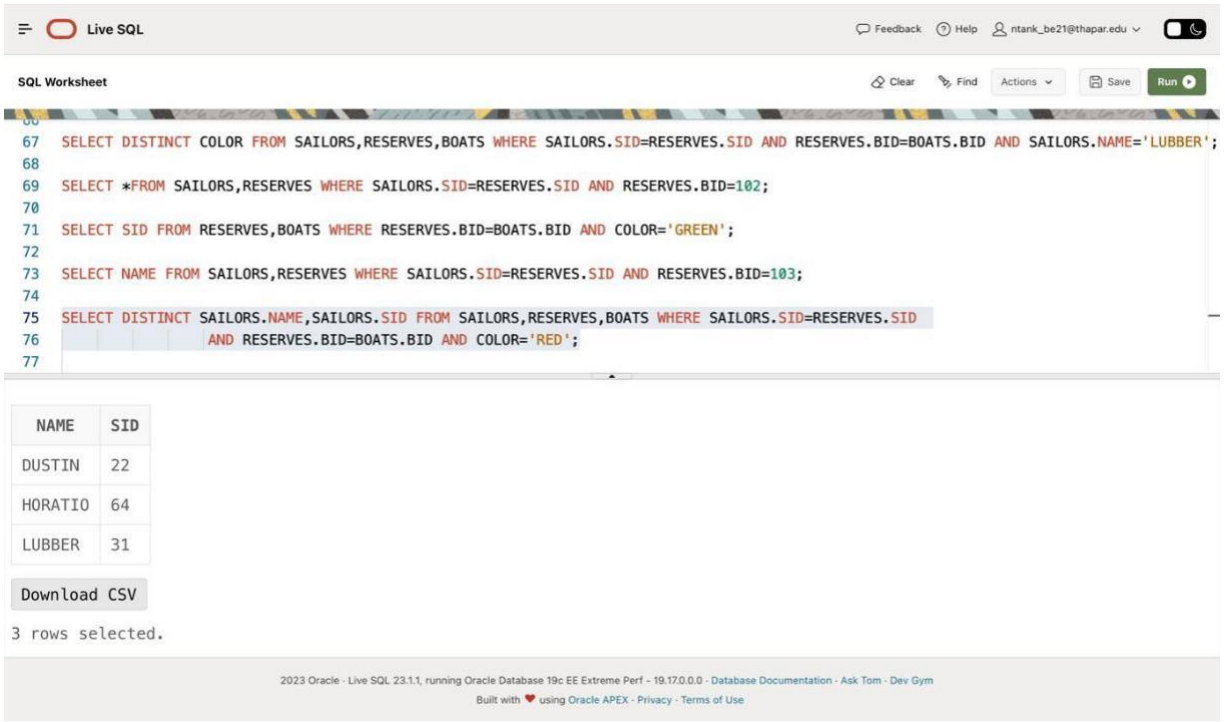
x) Find the sid of sailors who have reserved green boats.



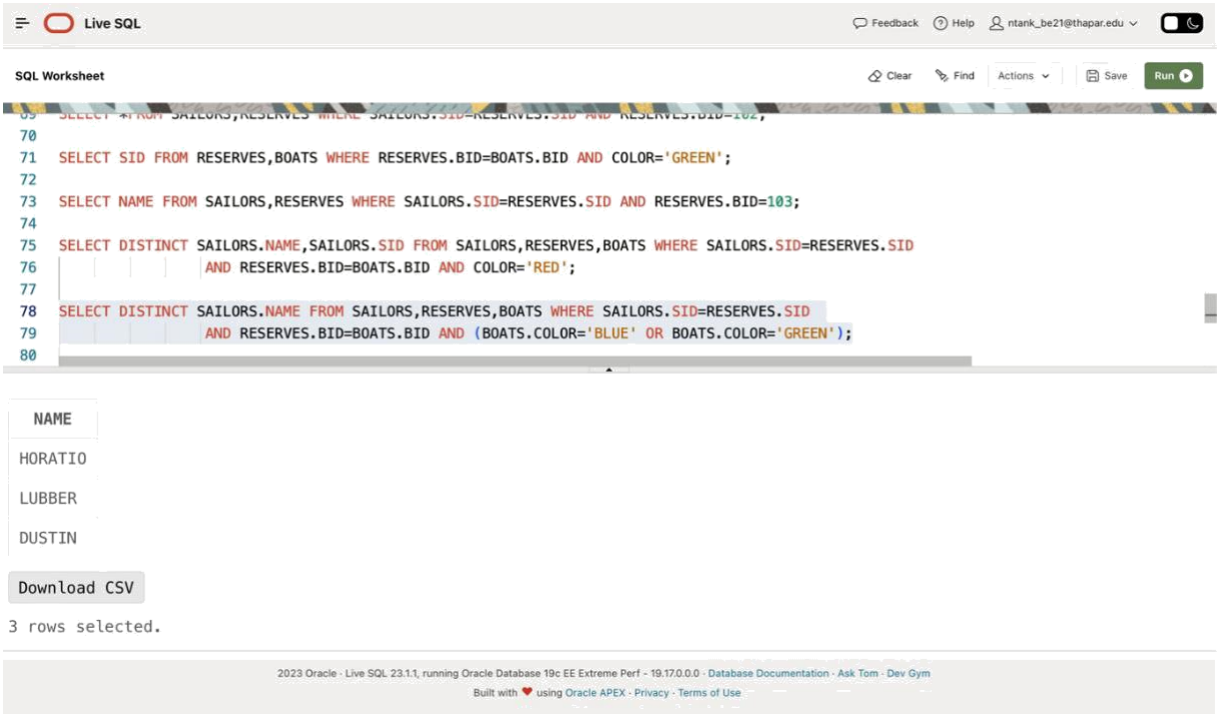
xi) Find the names of sailors who have reserved boat number 103.



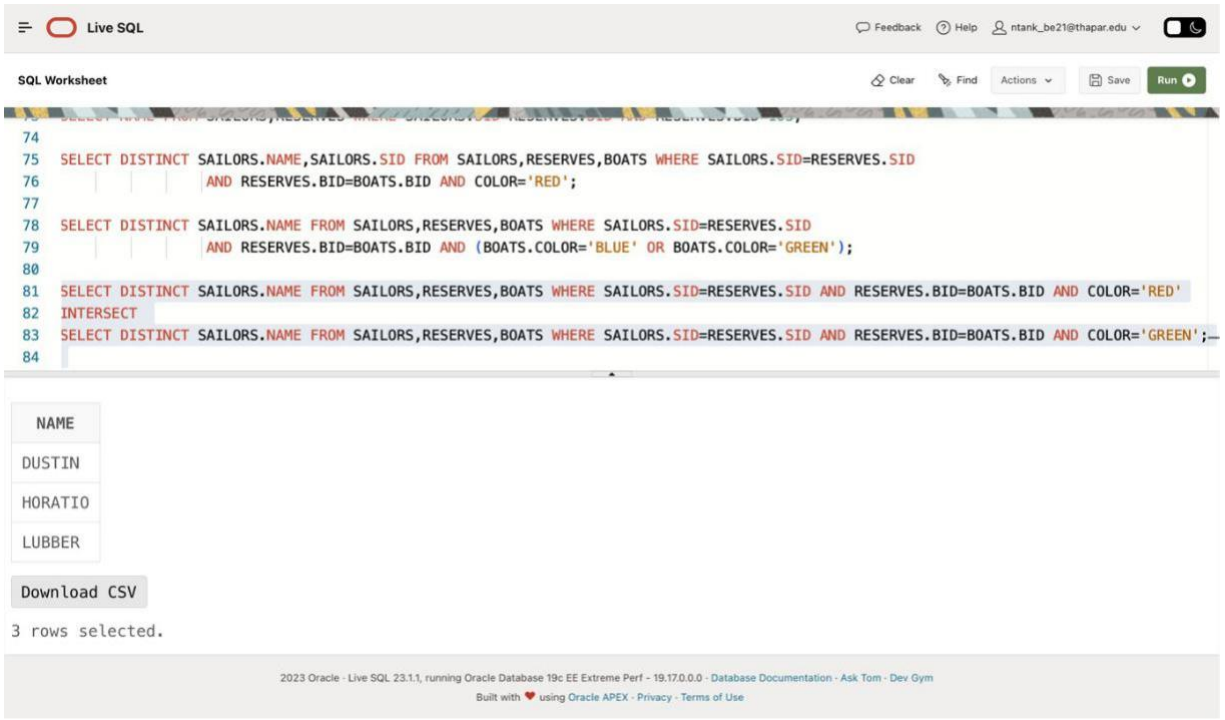
xii) Find the sids and names of sailors who have reserved a red boat.



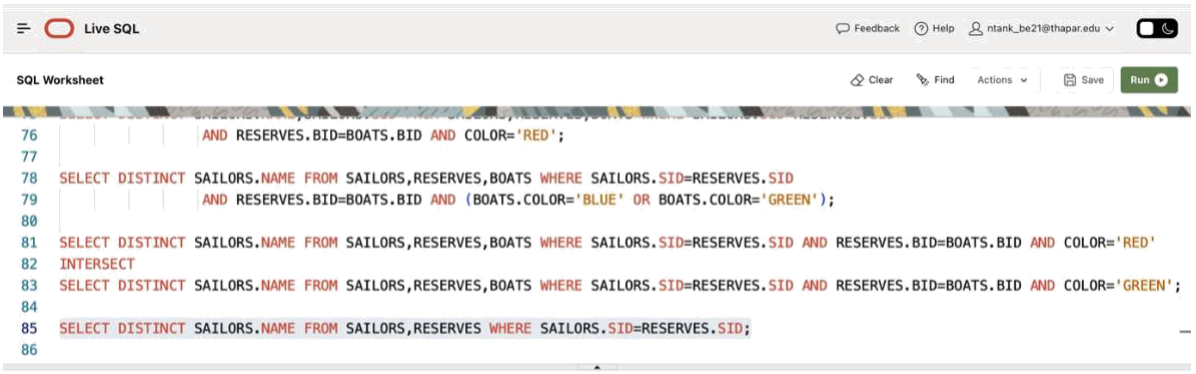
xiii) Find the names of the sailors who have reserved a green or a blue boat.



xiv) Find the names of sailors who have reserved both a red and a green boat.



xv) Find the names of sailors who have reserved at least one boat.



NAME
HORATIO
LUBBER
DUSTIN

Download CSV

3 rows selected.

ASSIGNMENT_6

1. **Display the system date**

```
mysql> SELECT CURRENT_DATE();
+----------------+
| CURRENT_DATE() |
+----------------+
| 2023-03-26     |
+----------------+
1 row in set (0.00 sec)
```

2. **Display current day**

```
mysql> SELECT DAYNAME(CURRENT_DATE());
+------------------------+
| DAYNAME(CURRENT_DATE()) |
+------------------------+
| Sunday                 |
+------------------------+
1 row in set (0.00 sec)
```

3. **Display current month and spell out year**

```
mysql> SELECT CONCAT(MONTHNAME(CURRENT_DATE()), ' ', YEAR(CURRENT_DATE()));
+-------------------------------------------------------------+
| CONCAT(MONTHNAME(CURRENT_DATE()), ' ', YEAR(CURRENT_DATE())) |
+-------------------------------------------------------------+
| March 2023                                                  |
+-------------------------------------------------------------+
1 row in set (0.00 sec)
```

4. **Display spell out current date**

```
mysql> SELECT DATE_FORMAT(CURRENT_DATE(), '%W, %M %e, %Y');
+-----------------------------------------------+
| DATE_FORMAT(CURRENT_DATE(), '%W, %M %e, %Y') |
+-----------------------------------------------+
| Sunday, March 26, 2023                        |
+-----------------------------------------------+
1 row in set (0.00 sec)
```

5. **Check whether it is AM or PM right now**

```
mysql> SELECT IF(TIME_FORMAT(CURRENT_TIME(), '%p') = 'AM', 'It is currently AM', 'It is currently PM');
+-------------------------------------------------------------------------------------------+
| IF(TIME_FORMAT(CURRENT_TIME(), '%p') = 'AM', 'It is currently AM', 'It is currently PM') |
+-------------------------------------------------------------------------------------------+
| It is currently PM                                                                        |
+-------------------------------------------------------------------------------------------+
1 row in set (0.00 sec)
```

6. **Display the date of next Friday**

```
mysql> SELECT DATE_ADD(CURRENT_DATE(), INTERVAL (5 - DAYOFWEEK(CURRENT_DATE())) % 7 + 1 DAY
);
+-----------------------------------------------------------------------------+
| DATE_ADD(CURRENT_DATE(), INTERVAL (5 - DAYOFWEEK(CURRENT_DATE())) % 7 + 1 DAY) |
+-----------------------------------------------------------------------------+
| 2023-03-31
    |
+-----------------------------------------------------------------------------+
1 row in set (0.00 sec)
```

7. **Round the system date on month**

```
mysql> SELECT DATE_FORMAT(CURRENT_DATE() + INTERVAL 15 DAY, '%Y-%m-01');
+-------------------------------------------------------------+
| DATE_FORMAT(CURRENT_DATE() + INTERVAL 15 DAY, '%Y-%m-01') |
+-------------------------------------------------------------+
| 2023-04-01                                                  |
+-------------------------------------------------------------+
```

8. **Truncate the system date on month**

```
mysql> SELECT LAST_DAY(CURRENT_DATE()) + INTERVAL 1 DAY - INTERVAL DAY(LAST_DAY(CURRENT_DAT
E())) DAY;
+-----------------------------------------------------------------------------------------+
| LAST_DAY(CURRENT_DATE()) + INTERVAL 1 DAY - INTERVAL DAY(LAST_DAY(CURRENT_DATE())) DAY |
+-----------------------------------------------------------------------------------------+
| 2023-03-01                                                                              |
+-----------------------------------------------------------------------------------------+
1 row in set (0.00 sec)
```

9. **Round the system date on year**

```
mysql> SELECT DATE_FORMAT(CURRENT_DATE() + INTERVAL 6 MONTH, '%Y-01-01');
+-------------------------------------------------------------+
| DATE_FORMAT(CURRENT_DATE() + INTERVAL 6 MONTH, '%Y-01-01') |
+-------------------------------------------------------------+
| 2023-01-01                                                  |
+-------------------------------------------------------------+
1 row in set (0.00 sec)
```

10. **Truncate the system date on year**

```
mysql> SELECT DATE_FORMAT(CURRENT_DATE(), '%Y-01-01');
+-----------------------------------------+
| DATE_FORMAT(CURRENT_DATE(), '%Y-01-01') |
+-----------------------------------------+
| 2023-01-01                              |
+-----------------------------------------+
1 row in set (0.00 sec)
```

11. **Find the day after three days**

```
mysql> SELECT DATE_ADD(CURRENT_DATE(), INTERVAL 3 DAY);
+-----------------------------------------+
| DATE_ADD(CURRENT_DATE(), INTERVAL 3 DAY) |
+-----------------------------------------+
| 2023-03-29                              |
+-----------------------------------------+
1 row in set (0.00 sec)
```

Create a table train having three four columns

12. **Train Number, date of Departure, time of departure, time of arrival**

```
mysql> use lab6;
Database changed
mysql> CREATE TABLE train (
    ->     train_number INT,
    ->     departure_date DATE,
    ->     departure_time TIME,
    ->     arrival_time TIME
    -> );
Query OK, 0 rows affected (0.02 sec)
```

13. **Insert five columns in train table**

```
mysql> INSERT INTO train (train_number, departure_date, departure_time, arrival_time)
    -> VALUES
    -> (1001, '2023-03-27', '08:00:00', '12:30:00'),
    -> (1002, '2023-03-28', '10:15:00', '14:45:00'),
    -> (1003, '2023-03-29', '14:30:00', '19:00:00'),
    -> (1004, '2023-03-30', '18:45:00', '23:15:00'),
    -> (1005, '2023-03-31', '22:00:00', '02:30:00');
Query OK, 5 rows affected (0.01 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

14. **Display all the records**

```
mysql> SELECT * FROM train;
+--------------+----------------+----------------+--------------+
| train_number | departure_date | departure_time | arrival_time |
+--------------+----------------+----------------+--------------+
|         1001 | 2023-03-27     | 08:00:00       | 12:30:00     |
|         1002 | 2023-03-28     | 10:15:00       | 14:45:00     |
|         1003 | 2023-03-29     | 14:30:00       | 19:00:00     |
|         1004 | 2023-03-30     | 18:45:00       | 23:15:00     |
|         1005 | 2023-03-31     | 22:00:00       | 02:30:00     |
+--------------+----------------+----------------+--------------+
5 rows in set (0.00 sec)
```

15. **Display the time values inserted in the columns**

```
mysql> SELECT departure_time, arrival_time FROM train;
+----------------+--------------+
| departure_time | arrival_time |
+----------------+--------------+
| 08:00:00       | 12:30:00     |
| 10:15:00       | 14:45:00     |
| 14:30:00       | 19:00:00     |
| 18:45:00       | 23:15:00     |
| 22:00:00       | 02:30:00     |
+----------------+--------------+
5 rows in set (0.00 sec)
```

16. **Display those trains which arrived on PM**

```
mysql> SELECT * FROM train WHERE arrival_time >= '12:00:00' AND arrival_tim
e < '24:00:00';
+--------------+----------------+----------------+--------------+
| train_number | departure_date | departure_time | arrival_time |
+--------------+----------------+----------------+--------------+
|         1001 | 2023-03-27     | 08:00:00       | 12:30:00     |
|         1002 | 2023-03-28     | 10:15:00       | 14:45:00     |
|         1003 | 2023-03-29     | 14:30:00       | 19:00:00     |
|         1004 | 2023-03-30     | 18:45:00       | 23:15:00     |
+--------------+----------------+----------------+--------------+
4 rows in set (0.00 sec)
```

17. **Display train number who are going to depart in next on hour.**

```
mysql> SELECT train_number FROM train
    -> WHERE departure_time >= NOW() AND departure_time < DATE_ADD(NOW(), I
NTERVAL 1 HOUR);
Empty set (0.00 sec)
```

ASSIGNMENT_7

1. Check the structure of tables.

```
mysql> show tables;
+-----------------+
| Tables_in_lab4 |
+-----------------+
| book            |
| dept            |
| emp             |
| emp2            |
| p               |
| s               |
| sp              |
| st              |
+-----------------+
8 rows in set (0.00 sec)

mysql> desc p;
+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra |
+--------+-------------+------+-----+---------+-------+
| pno    | int         | NO   | PRI | NULL    |       |
| pname  | varchar(50) | YES  |     | NULL    |       |
| color  | varchar(50) | YES  |     | NULL    |       |
+--------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)

mysql> desc s;
+-------+-------------+------+-----+---------+-------+
| Field | Type        | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| sno   | int         | NO   | PRI | NULL    |       |
| sname | varchar(50) | YES  |     | NULL    |       |
| city  | varchar(50) | YES  |     | NULL    |       |
+-------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)

mysql> desc sp;
+-------+------+------+-----+---------+-------+
| Field | Type | Null | Key | Default | Extra |
+-------+------+------+-----+---------+-------+
| sno   | int  | NO   | PRI | NULL    |       |
| pno   | int  | NO   | PRI | NULL    |       |
| qty   | int  | YES  |     | NULL    |       |
+-------+------+------+-----+---------+-------+
3 rows in set (0.00 sec)

mysql> desc st;
+-------+---------+------+-----+---------+-------+
| Field | Type    | Null | Key | Default | Extra |
+-------+---------+------+-----+---------+-------+
| Rno   | int     | NO   | PRI | NULL    |       |
| Class | char(1) | NO   | PRI | NULL    |       |
| Marks | int     | YES  |     | NULL    |       |
+-------+---------+------+-----+---------+-------+
3 rows in set (0.00 sec)

mysql> desc book;
+-------+------+------+-----+---------+-------+
| Field | Type | Null | Key | Default | Extra |
+-------+------+------+-----+---------+-------+
| Rno   | int  | NO   | PRI | NULL    |       |
| DOI   | date | YES  |     | NULL    |       |
| DOR   | date | YES  |     | NULL    |       |
+-------+------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

```
mysql> desc dept;
+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra |
+--------+-------------+------+-----+---------+-------+
| deptno | int         | NO   | PRI | NULL    |       |
| dname  | varchar(50) | YES  |     | NULL    |       |
+--------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)

mysql> desc emp;
+--------+---------------+------+-----+---------+-------+
| Field  | Type          | Null | Key | Default | Extra |
+--------+---------------+------+-----+---------+-------+
| empno  | int           | NO   | PRI | NULL    |       |
| ename  | varchar(50)   | YES  | UNI | NULL    |       |
| job    | varchar(10)   | YES  |     | NULL    |       |
| sal    | decimal(10,2) | NO   |     | NULL    |       |
| deptno | int           | YES  |     | 10      |       |
+--------+---------------+------+-----+---------+-------+
5 rows in set (0.00 sec)

mysql> desc emp2;
+--------+---------------+------+-----+---------+-------+
| Field  | Type          | Null | Key | Default | Extra |
+--------+---------------+------+-----+---------+-------+
| empno  | int           | NO   | PRI | NULL    |       |
| ename  | varchar(50)   | YES  | UNI | NULL    |       |
| job    | varchar(50)   | YES  |     | NULL    |       |
| sal    | decimal(10,2) | NO   |     | NULL    |       |
| deptno | int           | YES  | MUL | NULL    |       |
+--------+---------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

2. Check the constraint name for applied constraints?

```
mysql> SHOW CREATE TABLE emp;
+-------+--------------------------------------------------------------------
| Table | Create Table
                                                                            |
+-------+--------------------------------------------------------------------
| emp   | CREATE TABLE `emp` (
  `empno` int NOT NULL,
  `ename` varchar(50) DEFAULT NULL,
  `job` varchar(10) DEFAULT NULL,
  `sal` decimal(10,2) NOT NULL,
  `deptno` int DEFAULT '10',
  PRIMARY KEY (`empno`),
  UNIQUE KEY `ename` (`ename`),
  CONSTRAINT `emp_chk_1` CHECK ((`job` in (_utf8mb4'Prof',_utf8mb4'AP',_utf8mb4'Lect')))
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci |
+-------+--------------------------------------------------------------------
1 row in set (0.00 sec)
```

3. Drop the unique constraint on ENAME

```
mysql> ALTER TABLE emp DROP INDEX ENAME ;
Query OK, 0 rows affected (0.03 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> |
```

4. Drop the Foreign Key constraint on DEPTNO
   alter table emp drop constraint SYS_C00117255289;

5. Add Foreign Key constraint on DEPTNO
   alter table emp add foreign key (deptno) references dept(deptno);

6. Change Data type of ENAME

```
mysql> ALTER TABLE emp MODIFY column ename VArCHAR(50);
Query OK, 0 rows affected (0.03 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

7. Change width of DNAME

```
mysql> ALTER TABLE dept MODIFY column dname VARCHAR(100)
Query OK, 3 rows affected (0.04 sec)
Records: 3  Duplicates: 0  Warnings: 0
```

8. Add COMM column in EMP table

```
mysql> ALTER TABLE emp ADD COLUMN COMM DECIMAL(10,2);
Query OK, 0 rows affected (0.01 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

9. Drop CITY column from J
   table alter table J drop
   column city;

10. Create duplicate copy of EMP table

```
mysql> CREATE TABLE emp_copy AS SELECT * FROM emp;
Query OK, 4 rows affected (0.02 sec)
Records: 4  Duplicates: 0  Warnings: 0
```

11. Copy structure of DEPT table in another table with different column names

```
mysql> CREATE TABLE NEW_TABLE (
    ->     NEW_DEPTNO INT NOT NULL,
    ->     NEW_DNAME VARCHAR(14),
    ->     PRIMARY KEY (NEW_DEPTNO)
    -> ) AS
    -> SELECT DEPTNO AS NEW_DEPTNO, DNAME AS NEW_DNAME FROM DEPT WHERE 1=0;
Query OK, 0 rows affected (0.01 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

12. Change the name and job of the employee whose EMPNO =100

```
mysql> UPDATE EMP
    -> SET ENAME = 'PULKIT', JOB = 'MANAG
    -> WHERE EMPNO = 100;
Query OK, 0 rows affected (0.00 sec)
Rows matched: 0  Changed: 0  Warnings: 0
```

13. Delete the record of employee who belongs to computer department

```
mysql> DELETE FROM EMP
    -> WHERE Deptno = (SELECT deptno FROM dept WHERE dname= 'comp ');
Query OK, 0 rows affected (0.00 sec)
```

14. Drop DEPT Table

```
mysql> DROP TABLE DEPT;
ERROR 3730 (HY000): Cannot drop table 'dept' referenced by a foreign key constraint 'emp2_ibfk_1' on table 'emp2'.
```

15. Drop duplicate table of EMP table

```
mysql> DROP TABLE emp_copy;
Query OK, 0 rows affected (0.01 sec)
```

ASSIGNMENT_8

1. Create the following schema and insert some tuples in these tables shown below.Author (ID, Name, Birth_Year, Death_Year (NULL in case of Author is alive)) Book (ID, Author_ID, Title, Publish_Year, Publishing_House, Rating)) Adaptation (Book_ID, Type, Title, Release_Year, Rating)

```
mysql> SELECT * FROM adaptation
    -> ;
+---------+-----------+-----------------------------------------+--------------+--------+
| Book_ID | Type      | Title                                   | Release_Year | Rating |
+---------+-----------+-----------------------------------------+--------------+--------+
|       3 | Movie     | Harry Potter and the Philosopher's Stone |         2001 |    4.0 |
|       4 | Movie     | Harry Potter and the Chamber of Secrets  |         2002 |    4.2 |
|       5 | Movie     | Twilight                                |         2008 |    3.1 |
|       6 | Movie     | New Moon                                |         2009 |    3.3 |
|       7 | Movie     | The Hunger Games                        |         2012 |    4.1 |
|       8 | Movie     | Catching Fire                           |         2013 |    4.3 |
|       9 | TV Series | Game of Thrones                         |         2011 |    4.8 |
|      10 | TV Series | Game of Thrones                         |         2012 |    4.7 |
+---------+-----------+-----------------------------------------+--------------+--------+
8 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM BOOK;
+----+-----------+-----------------------------------------+--------------+--------------------------+--------+
| ID | Author_ID | Title                                   | Publish_Year | Publishing_House         | Rating |
+----+-----------+-----------------------------------------+--------------+--------------------------+--------+
|  1 |         1 | Five Point Someone                      |         2004 | Rupa Publications        |    3.8 |
|  2 |         1 | 2 States                                |         2009 | Rupa Publications        |    4.1 |
|  3 |         2 | Harry Potter and the Philosopher's Stone |         1997 | Bloomsbury Publishing    |    4.5 |
|  4 |         2 | Harry Potter and the Chamber of Secrets  |         1998 | Bloomsbury Publishing    |    4.3 |
|  5 |         3 | Twilight                                |         2005 | Little, Brown and Company |    3.2 |
|  6 |         3 | New Moon                                |         2006 | Little, Brown and Company |    3.5 |
|  7 |         4 | The Hunger Games                        |         2008 | Scholastic Corporation   |    4.0 |
|  8 |         4 | Catching Fire                           |         2009 | Scholastic Corporation   |    4.2 |
|  9 |         5 | A Game of Thrones                       |         1996 | Bantam Spectra           |    4.7 |
| 10 |         5 | A Clash of Kings                        |         1998 | Bantam Spectra           |    4.6 |
+----+-----------+-----------------------------------------+--------------+--------------------------+--------+
10 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM AUTHOR;
+----+-----------------+------------+------------+
| ID | Name            | Birth_Year | Death_Year |
+----+-----------------+------------+------------+
|  1 | Chetan Bhagat   |       1974 |       NULL |
|  2 | J.K. Rowling    |       1965 |       NULL |
|  3 | Stephenie Meyer |       1973 |       NULL |
|  4 | Suzanne Collins |       1962 |       NULL |
|  5 | George R.R. Martin |       1948 |       NULL |
+----+-----------------+------------+------------+
5 rows in set (0.00 sec)
```

2. Write SQL command for the following using Join:

a. Display the title of each book and the name of its author.

```
mysql> SELECT Book.Title, Author.Name
    -> FROM Book
    -> JOIN Author ON Book.Author_ID = Author.ID;
+----------------------------------------+---------------------+
| Title                                  | Name                |
+----------------------------------------+---------------------+
| Five Point Someone                     | Chetan Bhagat       |
| 2 States                               | Chetan Bhagat       |
| Harry Potter and the Philosopher's Stone | J.K. Rowling      |
| Harry Potter and the Chamber of Secrets  | J.K. Rowling      |
| Twilight                               | Stephenie Meyer     |
| New Moon                               | Stephenie Meyer     |
| The Hunger Games                       | Suzanne Collins     |
| Catching Fire                          | Suzanne Collins     |
| A Game of Thrones                      | George R.R. Martin  |
| A Clash of Kings                       | George R.R. Martin  |
+----------------------------------------+---------------------+
10 rows in set (0.00 sec)
```

b. Display the name of each author together with the title of the book they wrote and the year in which that book was published (Show only books published after 2005).

```
mysql> SELECT Author.Name, Book.Title, Book.Publish_Year
    -> FROM Author
    -> JOIN Book ON Author.ID = Book.Author_ID
    -> WHERE Book.Publish_Year > 2005;
+-----------------+-----------------+--------------+
| Name            | Title           | Publish_Year |
+-----------------+-----------------+--------------+
| Chetan Bhagat   | 2 States        |         2009 |
| Stephenie Meyer | New Moon        |         2006 |
| Suzanne Collins | The Hunger Games |         2008 |
| Suzanne Collins | Catching Fire   |         2009 |
+-----------------+-----------------+--------------+
4 rows in set (0.00 sec)
```

c. For each book, show its title, adaptation title, adaptation year, and publication year. Consider only books with a rating lower than the rating of their corresponding adaptation.

```
mysql> SELECT Book.Title, Adaptation.Title, Adaptation.Release_Year, Book.Publish_Year
    -> FROM Book
    -> JOIN Adaptation ON Book.ID = Adaptation.Book_ID
    -> WHERE Book.Rating < Adaptation.Rating;
+-----------------+-----------------+--------------+--------------+
| Title           | Title           | Release_Year | Publish_Year |
+-----------------+-----------------+--------------+--------------+
| The Hunger Games | The Hunger Games |         2012 |         2008 |
| Catching Fire   | Catching Fire   |         2013 |         2009 |
| A Game of Thrones | Game of Thrones |         2011 |         1996 |
| A Clash of Kings | Game of Thrones |         2012 |         1998 |
+-----------------+-----------------+--------------+--------------+
4 rows in set (0.00 sec)
```

d. Display the title of each book together with its rating. Consider only those books that were published by authors who are still alive. (Use Inner Join).

```
mysql> SELECT Book.Title, Book.Rating
    -> FROM Book
    -> JOIN Author ON Book.Author_ID = Author.ID
    -> WHERE Author.Death_Year IS NULL;
+----------------------------------------+--------+
| Title                                  | Rating |
+----------------------------------------+--------+
| Five Point Someone                     |    3.8 |
| 2 States                               |    4.1 |
| Harry Potter and the Philosopher's Stone |    4.5 |
| Harry Potter and the Chamber of Secrets  |    4.3 |
| Twilight                               |    3.2 |
| New Moon                               |    3.5 |
| The Hunger Games                       |    4.0 |
| Catching Fire                          |    4.2 |
| A Game of Thrones                      |    4.7 |
| A Clash of Kings                       |    4.6 |
+----------------------------------------+--------+
10 rows in set (0.00 sec)
```

e. Display the title of each book along with the name of its author. Show all books, even those without an author. Show all authors, even those who haven't published a book yet.(Use Full JOIN).

```
SELECT Book.Title,
Author.Name FROM Book
FULL JOIN Author ON Book.Author_ID = Author.ID;
```

f. Generate all possible pairs of book titles and author names. Consider only books whose author's name is 'Chetan Bhagat' (Use Cross JOIN).

```
mysql> SELECT Book.Title, Author.Name
    -> FROM Book
    -> CROSS JOIN Author
    -> WHERE Author.Name = 'Chetan Bhagat';
+-----------------------------------------+----------------+
| Title                                   | Name           |
+-----------------------------------------+----------------+
| Five Point Someone                      | Chetan Bhagat  |
| 2 States                                | Chetan Bhagat  |
| Harry Potter and the Philosopher's Stone | Chetan Bhagat  |
| Harry Potter and the Chamber of Secrets  | Chetan Bhagat  |
| Twilight                                | Chetan Bhagat  |
| New Moon                                | Chetan Bhagat  |
| The Hunger Games                        | Chetan Bhagat  |
| Catching Fire                           | Chetan Bhagat  |
| A Game of Thrones                       | Chetan Bhagat  |
| A Clash of Kings                        | Chetan Bhagat  |
+-----------------------------------------+----------------+
10 rows in set (0.00 sec)
```

g. Select each book's title, the name of its publishing house and the title of its adaptation on the type of the adaptation ('Movie'). (Use Right JOIN).

```
mysql> SELECT Book.Title, Book.Publishing_House, Adaptation.Title
    -> FROM Adaptation
    -> RIGHT JOIN Book ON Adaptation.Book_ID = Book.ID
    -> WHERE Adaptation.Type = 'Movie';
+-----------------------------------------+-------------------------+-----------------------------------------+
| Title                                   | Publishing_House        | Title                                   |
+-----------------------------------------+-------------------------+-----------------------------------------+
| Harry Potter and the Philosopher's Stone | Bloomsbury Publishing   | Harry Potter and the Philosopher's Stone |
| Harry Potter and the Chamber of Secrets  | Bloomsbury Publishing   | Harry Potter and the Chamber of Secrets  |
| Twilight                                | Little, Brown and Company | Twilight                                |
| New Moon                                | Little, Brown and Company | New Moon                                |
| The Hunger Games                        | Scholastic Corporation  | The Hunger Games                        |
| Catching Fire                           | Scholastic Corporation  | Catching Fire                           |
+-----------------------------------------+-------------------------+-----------------------------------------+
6 rows in set (0.00 sec)
```

h. Show the title of each book and the name of its author — but only if the author was born in the 20th century. Otherwise, the author's name field should be NULL (Use Left JOIN).

```
mysql> SELECT Book.Title, CASE WHEN Author.Birth_Year BETWEEN 1900 AND 1999 THEN Author.Name ELSE NULL END AS Author_Name
    -> FROM Book
    -> LEFT JOIN Author ON Book.Author_ID = Author.ID;
+-----------------------------------------+--------------------+
| Title                                   | Author_Name        |
+-----------------------------------------+--------------------+
| Five Point Someone                      | Chetan Bhagat      |
| 2 States                                | Chetan Bhagat      |
| Harry Potter and the Philosopher's Stone | J.K. Rowling       |
| Harry Potter and the Chamber of Secrets  | J.K. Rowling       |
| Twilight                                | Stephenie Meyer    |
| New Moon                                | Stephenie Meyer    |
| The Hunger Games                        | Suzanne Collins    |
| Catching Fire                           | Suzanne Collins    |
| A Game of Thrones                       | George R.R. Martin |
| A Clash of Kings                        | George R.R. Martin |
+-----------------------------------------+--------------------+
10 rows in set (0.00 sec)
```

3. Consider the following relation and execute the given queries(Aggregate/Group By/Having):

i) Show the total number of prizes awarded.

SELECT COUNT(winner) FROM nobel

ii) List each subject - just once

SELECT DISTINCT subject FROM nobel

iii) Show the total number of prizes awarded for Physics

SELECT COUNT(*) FROM nobel WHERE subject='Physics'

iv) For each subject show the subject and the number of prizes.

SELECT subject,

COUNT(*) FROM nobel

GROUP BY subject

v) For each subject show the first year that the prize was awarded.

SELECT subject,MIN(yr) FROM

nobel GROUP BY subject

vi) For each subject show the number of prizes awarded in the year 2000.

SELECT subject,

COUNT(winner) FROM nobel

WHERE yr=2000

GROUP BY subject

vii) Show the number of different winners for each

subject. SELECT DISTINCT subject,

COUNT(DISTINCT winner) FROM nobel

GROUP BY subject

viii) For each subject show how many years have had prizes awarded.

SELECT subject, COUNT(DISTINCT yr)

FROM nobel

GROUP BY subject

ix) Show the years in which three prizes were given for

Physics. SELECT yr FROM nobel

WHERE

subject='Physics' GROUP

BY yr

HAVING COUNT(yr)=3

x) Show winners who have won more than once.

```
SELECT winner FROM nobel
GROUP BY winner
HAVING COUNT(winner)>1
```

xi) Show winners who have won more than one subject.

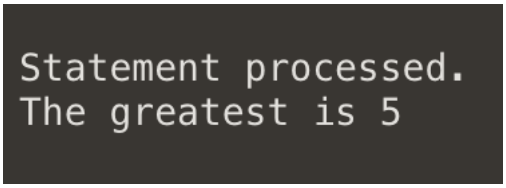
```
SELECT winner FROM nobel
GROUP BY winner
HAVING COUNT(DISTINCT subject) > 1
```

xii) Show the year and subject where 3 prizes were given. Show only years 2000 onwards.

```
SELECT yr, subject FROM nobel
WHERE yr >= 2000
GROUP BY yr, subject
HAVING COUNT(DISTINCT winner)=3
```

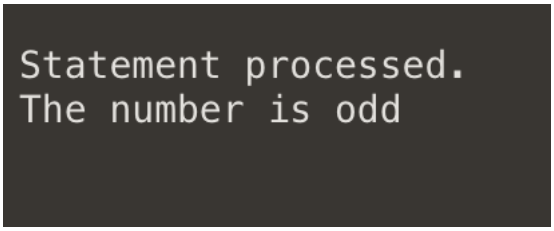
1) WAP to find the greatest of three numbers.

```
DECLARE
  num1 NUMBER := 2;
  num2 NUMBER := 5;
  num3 NUMBER := 7;
  greatest NUMBER;
BEGIN
  IF(num1 > num2 AND num1 > num3) THEN
    greatest := num1;
  ELSIF(num2 > num1 AND num2 > num3) THEN
    greatest := num2;
  ELSE
    greatest := num3;
  END IF;
  DBMS_OUTPUT.PUT_LINE('The greatest number is: ' || greatest);
END;
```



WAP to check whether number is odd or even.

```
DECLARE
  num NUMBER := 5;
BEGIN
  IF(num MOD 2 = 0) THEN
    DBMS_OUTPUT.PUT_LINE('The number is even');
  ELSE
    DBMS_OUTPUT.PUT_LINE('The number is odd');
  END IF;
END;
```



WAP to find the grade.

```
DECLARE
  marks NUMBER := 75;
  grade CHAR(1);
BEGIN
  IF(marks > 80) THEN
    grade := 'A';
  ELSIF(marks > 70) THEN
    grade := 'B';
  ELSIF(marks > 50) THEN
    grade := 'C';
  ELSIF(marks > 40) THEN
    grade := 'D';
  ELSE
    grade := 'E';
  END IF;
  DBMS_OUTPUT.PUT_LINE('The grade is: ' || grade);
END;
```

```
Statement processed.
The grade is: B
```

WAP to print the table of a given number.(use for loop)

```
DECLARE
  num NUMBER := 6;
BEGIN
  FOR i IN 1..10 LOOP
    DBMS_OUTPUT.PUT_LINE(num || ' x ' || i || ' = ' || num*i);
  END LOOP;
END;
```

```
Statement processed.
6 x 1 = 6
6 x 2 = 12
6 x 3 = 18
6 x 4 = 24
6 x 5 = 30
6 x 6 = 36
6 x 7 = 42
6 x 8 = 48
6 x 9 = 54
6 x 10 = 60
```

WAP to find out the factorial of a given number.(use while loop)

```
DECLARE
  num NUMBER := &num;
  factorial NUMBER := 1;
BEGIN
  WHILE(num > 0) LOOP
    factorial := factorial*num;
    num := num-1;
  END LOOP;
  DBMS_OUTPUT.PUT_LINE('The factorial is: ' || factorial);
END;
```

WAP to find out the Fibonacci series.

```
DECLARE
  num NUMBER := 5;
  fib1 NUMBER := 0;
  fib2 NUMBER := 1;
  fib3 NUMBER;
BEGIN
  DBMS_OUTPUT.PUT_LINE(fib1);
  DBMS_OUTPUT.PUT_LINE(fib2);
  FOR i IN 3..num LOOP
    fib3 := fib1 + fib2;
    DBMS_OUTPUT.PUT_LINE(fib3);
    fib1 := fib2;
    fib2 := fib3;
  END LOOP;
```

```
END;
```

```
Statement processed.
0
1
1
2
3
```

WAP to find the reverse of a number

```
DECLARE
  num NUMBER := 12;
  reverse NUMBER := 0;
  rem NUMBER;
BEGIN
  WHILE(num > 0) LOOP
    rem := num MOD 10;
    reverse := reverse*10 + rem;
    num := FLOOR(num/10); -- Use integer division using the FLOOR function
  END LOOP;
  DBMS_OUTPUT.PUT_LINE('The reverse number is: ' || reverse);
END;
```

```
Statement processed.
The reverse number is: 21
```

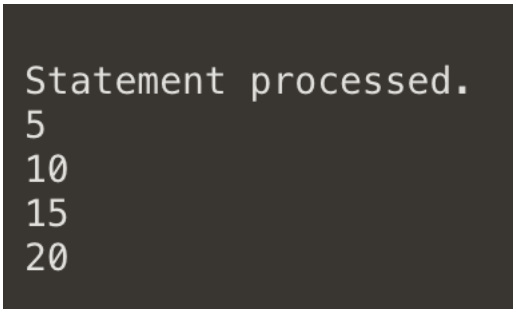
Write PL/SQL block that performs addition (+), subtraction (-), multiplication (*) and division (/) of two numbers as choice by the user.

```
DECLARE
  -- Declare variables for two numbers and the operation choice
  num1 NUMBER;
  num2 NUMBER;
  choice CHAR(1);
  result NUMBER;

BEGIN
  -- Prompt the user to enter two numbers
  num1 := 3;
  num2 := 7;

  -- Prompt the user to choose an operation
  choice := '+';

  -- Perform the operation based on user choice
  CASE choice
    WHEN '+' THEN
      result := num1 + num2;
      dbms_output.put_line(num1 || ' + ' || num2 || ' = ' || result);
    WHEN '-' THEN
      result := num1 - num2;
      dbms_output.put_line(num1 || ' - ' || num2 || ' = ' || result);
    WHEN '*' THEN
      result := num1 * num2;
      dbms_output.put_line(num1 || ' * ' || num2 || ' = ' || result);
    WHEN '/' THEN
      IF num2 = 0 THEN
        dbms_output.put_line('Cannot divide by zero');
      ELSE
```

```
        result := num1 / num2;
        dbms_output.put_line(num1 || ' / ' || num2 || ' = ' || result);
      END IF;
    ELSE
      dbms_output.put_line('Invalid choice');
  END CASE;
END;
```

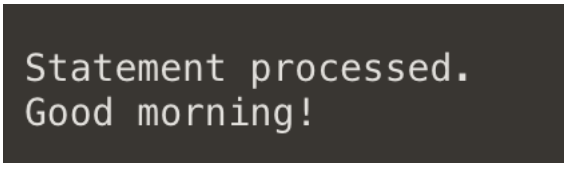
Write PL/SQL block to print 5, 10, 15,20 by using For Loop.

```
BEGIN
  FOR i IN 1..4 LOOP
    DBMS_OUTPUT.PUT_LINE(i * 5);
  END LOOP;
END;
```



Write Pl/SQL block to display welcome message like good morning, good afternoon, good

```
DECLARE
  current_hour NUMBER := TO_NUMBER(TO_CHAR(SYSDATE, 'HH24'));
BEGIN
  IF current_hour >= 0 AND current_hour < 12 THEN
    DBMS_OUTPUT.PUT_LINE('Good morning!');
  ELSIF current_hour >= 12 AND current_hour < 18 THEN
    DBMS_OUTPUT.PUT_LINE('Good afternoon!');
  ELSE
    DBMS_OUTPUT.PUT_LINE('Good night!');
  END IF;
END;
```



Double-click (or enter) to edit

```
DECLARE
  emp_salary NUMBER;
  emp_count NUMBER;
BEGIN
  -- Too Many Rows
  SELECT COUNT(*) INTO emp_count FROM hr.employees;
  IF(emp_count > 1) THEN
    RAISE_APPLICATION_ERROR(-20100, 'Too Many Rows');
  END IF;

  -- No Data Found
  SELECT salary INTO emp_salary FROM hr.employees WHERE employee_id = 9999;
  IF(emp_salary IS NULL) THEN
    RAISE_APPLICATION_ERROR(-20101, 'No Data Found');
  END IF;
END;
```

```
ORA-20100: Too Many Rows ORA-06512: at line 8
ORA-06512: at "SYS.DBMS_SQL", line 1721
```

Write a PL/SQL code to display a message to check whether the record is deleted or not.

```
DECLARE
  emp_id employees.employee_id%TYPE:= 234;
BEGIN
  DELETE FROM employees WHERE employee_id = emp_id;
  IF(SQL%ROWCOUNT = 1) THEN
    DBMS_OUTPUT.PUT_LINE('Record Deleted Successfully');
  ELSE
    DBMS_OUTPUT.PUT_LINE('Record Not Found');
  END IF;
END;
```

Double-click (or enter) to edit

Write a PL/SQL code to display a message to provide the information about the number of records deleted by the delete statement issued in a PL/SQL block. Cursor EMP (empno, ename, job, sal, deptno)

```
DECLARE
  emp_count NUMBER;
BEGIN
  DELETE FROM employees WHERE department_id = 10;
  emp_count := SQL%ROWCOUNT;
  DBMS_OUTPUT.PUT_LINE('Total ' || emp_count || ' records deleted');
END;
```

Double-click (or enter) to edit

Write a PL/SQL code to demonstrate %TYPE and %ROWTYPE to display details of employees in EMP table.

```
DECLARE
  emp employees%ROWTYPE;
BEGIN
  SELECT * INTO emp FROM employees WHERE employee_id = 100;
  DBMS_OUTPUT.PUT_LINE('Employee Details:');
  DBMS_OUTPUT.PUT_LINE('Employee ID: ' || emp.employee_id);
  DBMS_OUTPUT.PUT_LINE('Employee Name: ' || emp.first_name || ' ' || emp.last_name);
  DBMS_OUTPUT.PUT_LINE('Job: ' || emp.job_id);
  DBMS_OUTPUT.PUT_LINE('Salary: ' || emp.salary);
  DBMS_OUTPUT.PUT_LINE('Department ID: ' || emp.department_id);
END;
```

Write a PL/SQL code to display the empno, ename, job of employees of department number 10 for EMP table (using Cursor).

```
DECLARE
  CURSOR emp_cursor IS
    SELECT employee_id, first_name, last_name, job_id
    FROM employees
    WHERE department_id = 10;
BEGIN
  DBMS_OUTPUT.PUT_LINE('Employee Details:');
  FOR emp IN emp_cursor LOOP
    DBMS_OUTPUT.PUT_LINE('Employee ID: ' || emp.employee_id);
    DBMS_OUTPUT.PUT_LINE('Employee Name: ' || emp.first_name || ' ' || emp.last_name);
    DBMS_OUTPUT.PUT_LINE('Job: ' || emp.job_id);
    DBMS_OUTPUT.PUT_LINE('-----------------------');
  END LOOP;
END;
```

Write a PL/SQL code to display the employee number and name of top 5 highest paid Employees (using Cursor).

```
DECLARE
  CURSOR emp_cursor IS
    SELECT employee_id, first_name || ' ' || last_name AS emp_name, salary
    FROM employees
    ORDER BY salary DESC
    FETCH FIRST 5 ROWS ONLY;
BEGIN
  DBMS_OUTPUT.PUT_LINE('Top 5 highest paid Employees:');
  FOR emp IN emp_cursor LOOP
    DBMS_OUTPUT.PUT_LINE('Employee ID: ' || emp.employee_id);
    DBMS_OUTPUT.PUT_LINE
```

Write a PL/SQL code to calculate the total salary of first n records of emp table. The value of n is passed to cursor as parameter

```
DECLARE
  n NUMBER := 5;
  total_salary NUMBER := 0;
  CURSOR emp_cursor IS
    SELECT salary FROM emp WHERE ROWNUM <= n;
BEGIN
  FOR emp_record IN emp_cursor LOOP
    total_salary := total_salary + emp_record.salary;
  END LOOP;
  DBMS_OUTPUT.PUT_LINE('Total salary of first ' || n || ' employees is: ' || total_salary);
END;
```

PL/SQL code for local procedure to return multiple values through parameters:

```
CREATE OR REPLACE PROCEDURE arithmetic_operations(
    num1 IN NUMBER,
    num2 IN NUMBER,
    sum OUT NUMBER,
    diff OUT NUMBER,
    prod OUT NUMBER,
    quotient OUT NUMBER) IS
BEGIN
    sum := num1 + num2;
    diff := num1 - num2;
    prod := num1 * num2;
    quotient := num1 / num2;
END;
```

PL/SQL code for local procedure to raise salary and update it to the database:

```
CREATE OR REPLACE PROCEDURE raise_salary(
    empid IN NUMBER,
    bonus IN NUMBER) IS
    current_salary NUMBER;
BEGIN
    SELECT salary INTO current_salary FROM emp WHERE eno = empid;
    UPDATE emp SET salary = current_salary + bonus WHERE eno = empid;
    COMMIT;
    DBMS_OUTPUT.PUT_LINE('Salary of employee ' || empid || ' has been raised by ' || bonus);
END;
```

PL/SQL code for stored procedure to delete employee by employee number:

```
CREATE OR REPLACE PROCEDURE fire_employee(
    empid IN NUMBER) IS
BEGIN
    DELETE FROM emp WHERE eno = empid;
    COMMIT;
    DBMS_OUTPUT.PUT_LINE('Employee ' || empid || ' has been fired');
END;
```

PL/SQL code for local function to return total number of employees in EMP table:

```
CREATE OR REPLACE FUNCTION get_employee_count RETURN NUMBER IS
    employee_count NUMBER;
BEGIN
    SELECT COUNT(*) INTO employee_count FROM emp;
    RETURN employee_count;
END;
```

PL/SQL code for stored function to add two numbers:

```
CREATE OR REPLACE FUNCTION add_numbers(
    num1 IN NUMBER,
    num2 IN NUMBER) RETURN NUMBER IS
    result NUMBER;
BEGIN
    result := num1 + num2;
    RETURN result;
END;
```

PL/SQL code for stored function to return the number of records updated by UPDATE statement:

```
CREATE OR REPLACE FUNCTION get_update_count RETURN NUMBER IS
    update_count NUMBER;
BEGIN
    UPDATE emp SET salary = salary + 100 WHERE eno = 100;
    update_count := SQL%ROWCOUNT;
    RETURN update_count;
```

```
END;
```

PL/SQL code for trigger to convert ENAME column values to uppercase:

```
CREATE OR REPLACE TRIGGER convert_ename_to_uppercase
BEFORE INSERT OR UPDATE ON emp
FOR EACH ROW
BEGIN
    :NEW.ename := UPPER(:NEW.ename);
END;
```

PL/SQL code for trigger to show old and new values of ENAME after every update:

```
CREATE OR REPLACE TRIGGER show_old_new_ename
AFTER UPDATE ON emp
FOR EACH ROW
BEGIN
    DBMS_OUTPUT.PUT_LINE('Old value of ENAME: ' || :OLD.ename || ', New value of ENAME: ' || :NEW.ename);
END;
```

PL/SQL code for trigger to prevent operations on EMP table on Sunday:

```
CREATE OR REPLACE TRIGGER prevent_sunday_operations
BEFORE INSERT OR UPDATE OR DELETE ON emp
BEGIN
    IF TO_CHAR(SYSDATE, 'DAY') = 'SUNDAY' THEN
        RAISE_APPLICATION_ERROR(-20001, 'No operations allowed on Sundays');
    END IF;
END;
```

PL/SQL code for trigger to enforce commission to be less than or equal to salary:

```
CREATE OR REPLACE TRIGGER commission_check
BEFORE INSERT OR UPDATE ON EMP
FOR EACH ROW
BEGIN
    IF :new.COMMISSION > :new.SALARY THEN
        RAISE_APPLICATION_ERROR(-20001, 'Commission cannot be greater than salary.');
    END IF;
END;
/
```

Trigger to implement the primary key constraint on column ENO of table EMP:

```
CREATE OR REPLACE TRIGGER eno_pk
BEFORE INSERT ON EMP
FOR EACH ROW
DECLARE
  l_cnt NUMBER;
BEGIN
  SELECT COUNT(*)
  INTO l_cnt
  FROM EMP
  WHERE ENO = :NEW.ENO;
  IF l_cnt > 0 THEN
    RAISE_APPLICATION_ERROR(-20001, 'Duplicate ENO not allowed');
  END IF;
END;
/
```

Trigger to implement foreign key constraint on DEPTNO column of EMP table:

```
CREATE OR REPLACE TRIGGER deptno_fk
BEFORE INSERT OR UPDATE ON EMP
FOR EACH ROW
DECLARE
  l_cnt NUMBER;
```

```
BEGIN
  SELECT COUNT(*)
  INTO l_cnt
  FROM DEPT
  WHERE DEPTNO = :NEW.DEPTNO;
  IF l_cnt = 0 THEN
    RAISE_APPLICATION_ERROR(-20001, 'Invalid DEPTNO');
  END IF;
END;
/
```