Lab 10

Trigger, Transaction Management, Date

Version 2.0

Trigger

[illegible]

- [illegible]
- [illegible]

[illegible]

```
CREATE OR REPLACE TRIGGER check_salary BEFORE INSERT OR UPDATE ON
employees
FOR EACH ROW
DECLARE
   c_min constant number(8,2) := 1000.0;
   c_max constant number(8,2) := 500000.0;
BEGIN
  IF :new.salary > c_max OR :new.salary < c_min THEN
  RAISE_APPLICATION_ERROR(-20000,'New salary is too small or large');
END IF;
END;
/
```

[illegible] **new** [illegible]

[illegible] **student** [illegible]

```
roll_no number(4) PRIMARY KEY
name varchar2(50) NOT NULL
marks number(3)
grade char(1)
course_id number(4) FOREIGN KEY
```

[illegible]

```
marks>80 grade =A
marks 80-70 grade = B
marks 70-60 grade =C
marks 60- 50 grade = D
marks 50-40 grade = E
marks<=40 grade = F
```

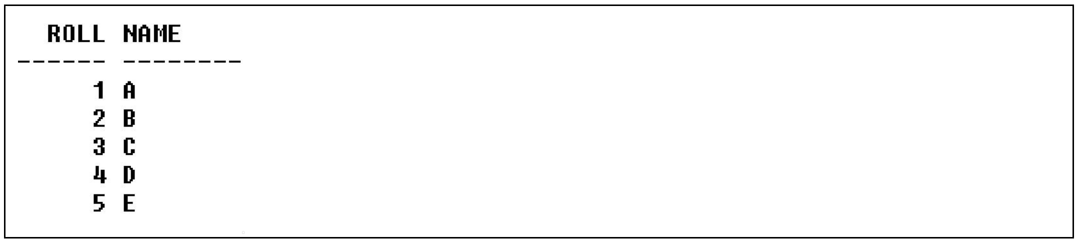
```
CREATE TABLE STUDENT
(
roll_no number(4) PRIMARY KEY,
name varchar2(50) NOT NULL,
marks number(3),
grade char(1),
course_id number(4)
);
```

```
CREATE TRIGGER TR_GRADE
BEFORE UPDATE OR INSERT ON STUDENT
FOR EACH ROW
BEGIN
IF :NEW.MARKS>80 THEN
:NEW.GRADE:='A';
ELSIF :NEW.MARKS>70 AND :NEW.MARKS<80 THEN
:NEW.GRADE:='B';
ELSIF :NEW.MARKS>60 AND :NEW.MARKS<70 THEN
:NEW.GRADE:='C';
ELSIF :NEW.MARKS>50 AND :NEW.MARKS<60 THEN
:NEW.GRADE:='D';
ELSIF :NEW.MARKS>40 AND :NEW.MARKS<50 THEN
:NEW.GRADE:='E';
ELSIF :NEW.MARKS<=40 THEN
:NEW.GRADE:='F';
END IF;
END TR_GRADE;
/
```

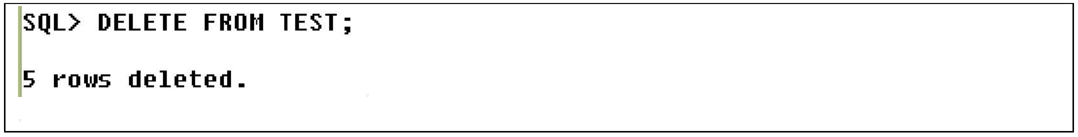
```
insert into student VALUES (1,'SATHISH',3,NULL,123);
SELECT * FROM STUDENT;
```

Transaction Management

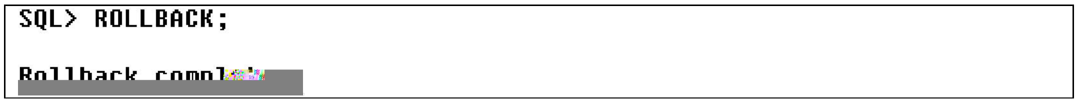
```
DROP TABLE test;
CREATE TABLE test (
      Roll  int,
      Name  Varchar (10)
);

INSERT INTO test VALUES (1, 'A');
INSERT INTO test VALUES (2, 'B');
INSERT INTO test VALUES (3, 'C');
INSERT INTO test VALUES (4, 'D');
INSERT INTO test VALUES (5, 'E');

COMMIT;
```



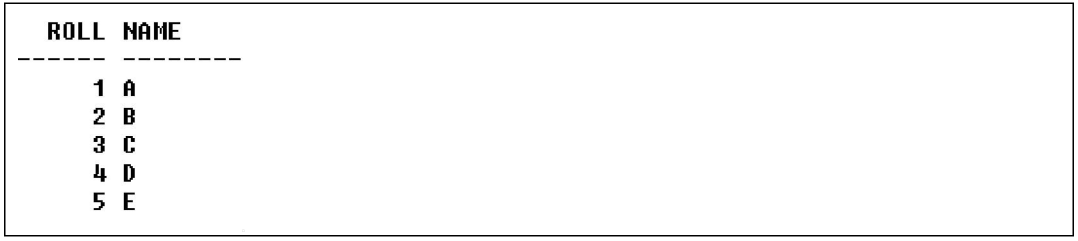
```
  ROLL NAME
------ --------
     1 A
     2 B
     3 C
     4 D
     5 E
```



```
SQL> DELETE FROM TEST;

5 rows deleted.
```

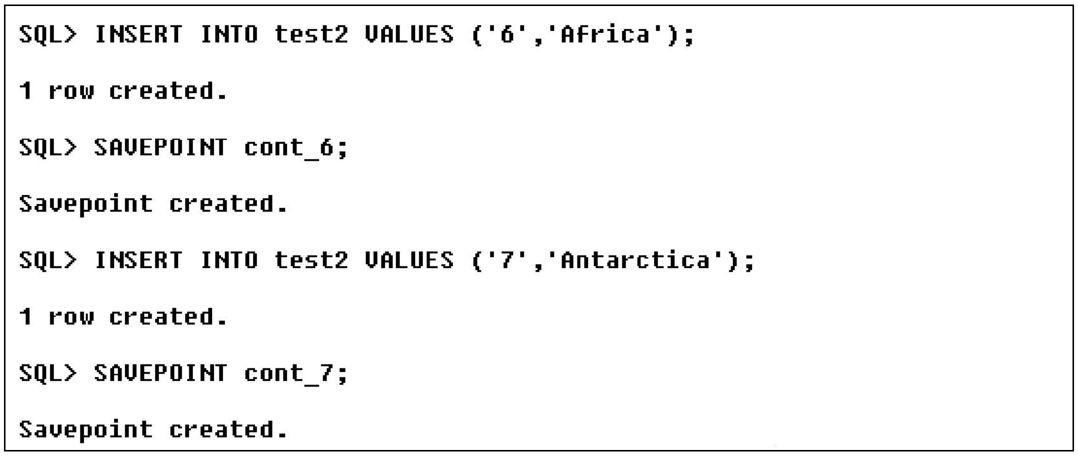


```
SQL> ROLLBACK;

Rollback compl
```

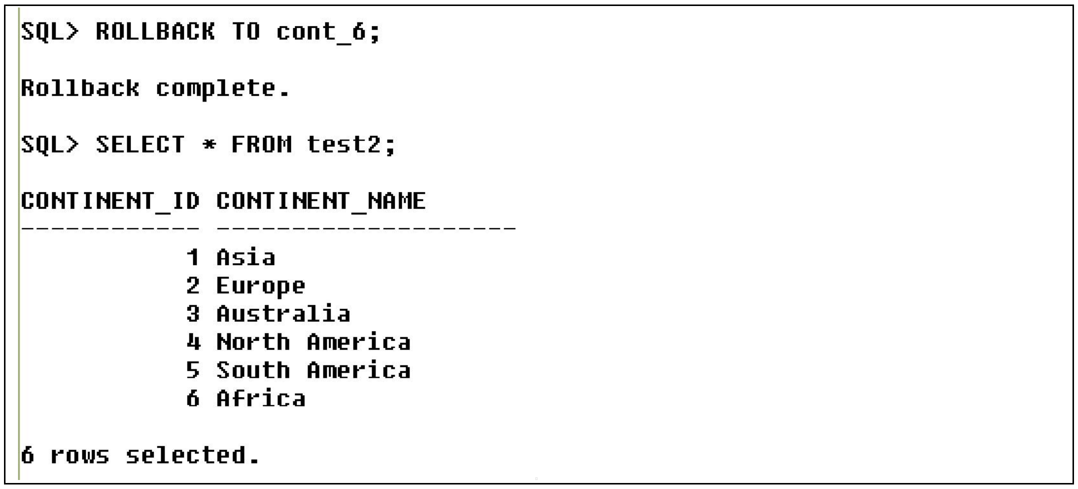


```
  ROLL NAME
------ --------
     1 A
     2 B
     3 C
     4 D
     5 E
```

```
DROP TABLE test2;
CREATE TABLE test2 (
      Continent_ID     int,
      Continent_Name   varchar(20)
);

INSERT INTO test2 VALUES (1, 'Asia');
INSERT INTO test2 VALUES (2, 'Europe');
INSERT INTO test2 VALUES (3, 'Australia');
INSERT INTO test2 VALUES (4, 'North America');
INSERT INTO test2 VALUES (5, 'South America');
commit;
```

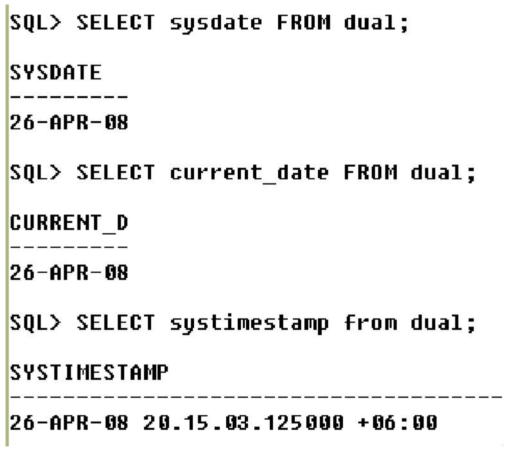


```
SQL> INSERT INTO test2 VALUES ('6','Africa');

1 row created.

SQL> SAVEPOINT cont_6;

Savepoint created.

SQL> INSERT INTO test2 VALUES ('7','Antarctica');

1 row created.

SQL> SAVEPOINT cont_7;

Savepoint created.
```



```
SQL> ROLLBACK TO cont_6;

Rollback complete.

SQL> SELECT * FROM test2;

CONTINENT_ID CONTINENT_NAME
------------ --------------------
           1 Asia
           2 Europe
           3 Australia
           4 North America
           5 South America
           6 Africa

6 rows selected.
```

Date



```
SQL> SELECT sysdate FROM dual;

SYSDATE
---------
26-APR-08

SQL> SELECT current_date FROM dual;

CURRENT_D
---------
26-APR-08

SQL> SELECT systimestamp from dual;

SYSTIMESTAMP
-------------------------------------
26-APR-08 20.15.03.125000 +06:00
```

```
SELECT Employee, Viva_date, Joining_date
FROM joining
WHERE Viva_date - Joining_date !=0;
```

☺

```
SELECT ADD_MONTHS (Joining_date,6) AS Six_months_Extension
FROM joining
WHERE Employee = 'A';
```

```
SELECT ADD_MONTHS (Joining_date, -6) AS Six_months_Extension
FROM joining
WHERE Employee = 'A';
```

```
SELECT LEAST (TO_DATE ('22-DEC-2007'),TO_DATE ('12-DEC-2008'))
FROM dual;
SELECT  GREATEST  (TO_DATE  ('22-DEC-2007'),TO_DATE  ('12-DEC-
2008'))
FROM dual;
```

[illegible]

[illegible]

```
SELECT LAST_DAY (Viva_date)
FROM joining;
```

[illegible]

```
SELECT employee, EXTRACT(Year FROM Viva_date) AS Year
FROM joining;
```

[illegible]

[illegible]

```
SELECT employee, EXTRACT(Month FROM Viva_date) AS Month
FROM joining;
```