DBMS Lab 7

V Venkataraman 106118106

Q1

```
CREATE DATABASE LAB7;
USE LAB7;
DROP TABLE IF EXISTS PRODUCT;
DROP TABLE IF EXISTS SALE;
DROP TABLE IF EXISTS SALEITEM;

CREATE TABLE PRODUCT (
    BARCODE INT NOT NULL,
    PNAME VARCHAR(40),
    PRICE INT,
    QUANTITYINSTOCK INT,
    PRIMARY KEY (BARCODE)
);

CREATE TABLE SALE (
    SALEID INT NOT NULL,
    DELIVERYADDRESS VARCHAR(40),
    CREDITCARD INT,
    PRIMARY KEY(SALEID)
);

CREATE TABLE SALEITEM (
    SALEID INT NOT NULL,
    BARCODE INT NOT NULL,
    QUANTITY INT,
    PRIMARY KEY (BARCODE, SALEID),
    FOREIGN KEY (BARCODE) REFERENCES PRODUCT(BARCODE)
);

CREATE TRIGGER UPDATEAVAILABLEQUANTITY
AFTER INSERT
ON SALEITEM
FOR EACH ROW
UPDATE PRODUCT SET PRODUCT.QUANTITYINSTOCK = PRODUCT.QUANTITYINSTOCK -
NEW.QUANTITY WHERE NEW.BARCODE = PRODUCT.BARCODE;


INSERT INTO PRODUCT
VALUES (1, "SSD", 5000, 20),
    (2, "HDD", 2000, 10);
```

```
INSERT INTO SALEITEM
VALUES (1, 1, 2);
```

Q2

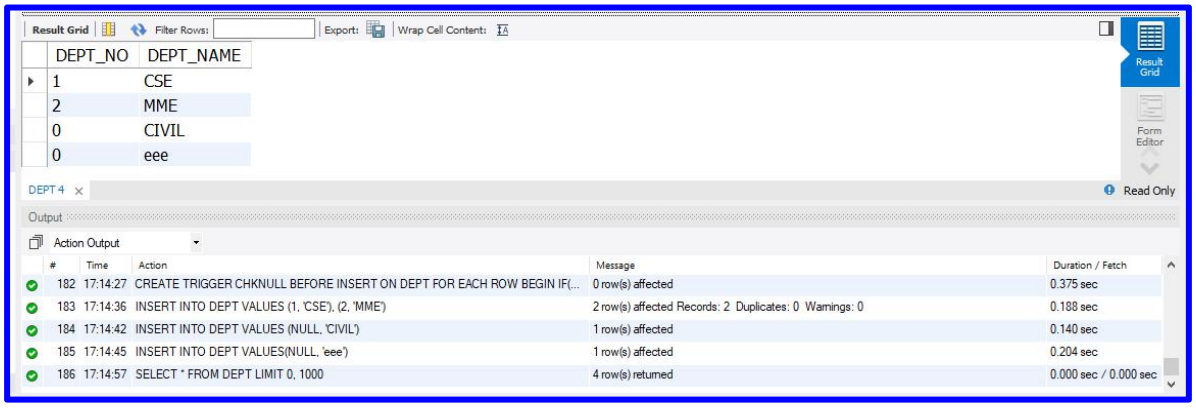
```
USE LAB7;
DROP TABLE IF EXISTS CONCERTS;
DROP TABLE IF EXISTS PERFORMERS;
DROP TABLE IF EXISTS ARENAS;
DROP TABLE IF EXISTS ACTIVITIES;

CREATE TABLE ACTIVITIES (
    ACTIVITYID INT NOT NULL,
    ACTIVITYNAME VARCHAR(40),
    PRIMARY KEY (ACTIVITYID)
);

CREATE TABLE PERFORMERS (
    PERFORMERID    INT NOT NULL,
    PERFORMERNAME VARCHAR(40),
    STREET VARCHAR(40),
    CITY VARCHAR(40),
    STATE VARCHAR(40),
    ZIP INT(6),
    ACTIVITYID INT,
    PRIMARY KEY (PERFORMERID),
    FOREIGN KEY (ACTIVITYID) REFERENCES ACTIVITiES(ACTIVITYID)
);

CREATE TABLE ARENAS (
    ARENAID INT NOT NULL,
    ARENANAME VARCHAR(40),
    CITY VARCHAR(40),
    ARENACAPACITY VARCHAR(40),
    PRIMARY KEY (ARENAID)
```

```
);

CREATE TABLE CONCERTS (
    PERFORMERID INT,
    ARENAID INT,
    CONCERTDATE DATETIME,
    TICKETPRICE INT,
    PRIMARY KEY (PERFORMERID, ARENAID),
    FOREIGN KEY (PERFORMERID) REFERENCES PERFORMERS(PERFORMERID),
    FOREIGN KEY (ARENAID) REFERENCES ARENAS(ARENAID)
);


INSERT INTO ACTIVITiES
VALUES (1, "DANCING"),
    (2, "SWIMMING"),
    (3, "ATHLETICS");

INSERT INTO PERFORMERS
VALUES (1, "ASHUTOSH", "THUVAKUDI", "TRICHY", "TAMIL NADU", 620015, 1),
    (2, "ABHSISHEK", "THUVAKUDI", "TRICHY", "TAMIL NADU", 620015, 1),
    (3, "DHRUV", "THUVAKUDI", "TRICHY", "TAMIL NADU", 620015, 2);



DELIMITER //
CREATE TRIGGER DELETEPERFORMER
BEFORE DELETE
ON PERFORMERS
FOR EACH ROW
BEGIN
      IF (1 = (SELECT COUNT(PERFORMERID)
                          FROM PERFORMERS
                          WHERE PERFORMERS.ACTIVITYID = OLD.ACTIVITYID
                        )
          ) THEN
          SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT ='DELETION IS NOT ALLOWED!';
      END IF;
END;
//

DELETE FROM PERFORMERS WHERE PERFORMERID=2;
```

Q3

```
USE LAB7;
DROP TABLE IF EXISTS DEPT;
DROP TABLE IF EXISTS EMPLOYEE;
DELETE FROM DEPT;

CREATE TABLE DEPT (
    DEPT_NO INT,
    DEPT_NAME VARCHAR(20)
);

CREATE TABLE EMPLOYEE (
    EMP_NO INT,
    DEPT_NO INT,
    JOB VARCHAR(20),
    SALARY INT
);


--  3.1
DELIMITER //
CREATE TRIGGER CHKNULL
BEFORE INSERT
ON DEPT
FOR EACH ROW
BEGIN
    IF(new.DEPT_NO IS NULL)
    THEN SET new.DEPT_NO=0;
      END IF;
END;
//

INSERT INTO DEPT
VALUES (1, 'CSE'),
    (2, 'MME');
```

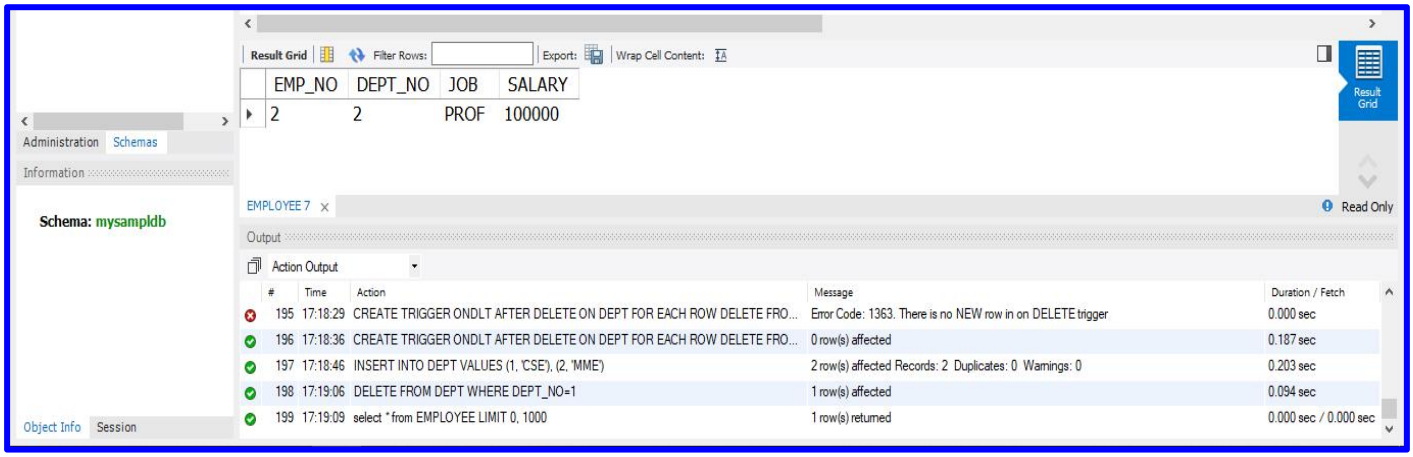
```
INSERT INTO DEPT
VALUES (NULL, 'CIVIL');

INSERT INTO DEPT VALUES(NULL, 'eee');
```



DEPT_NO	DEPT_NAME
1	CSE
2	MME
0	CIVIL
0	eee

#	Time	Action	Message	Duration / Fetch
182	17:14:27	CREATE TRIGGER CHKNULL BEFORE INSERT ON DEPT FOR EACH ROW BEGIN IF(...	0 row(s) affected	0.375 sec
183	17:14:36	INSERT INTO DEPT VALUES (1, 'CSE'), (2, 'MME')	2 row(s) affected Records: 2 Duplicates: 0 Warnings: 0	0.188 sec
184	17:14:42	INSERT INTO DEPT VALUES (NULL, 'CIVIL')	1 row(s) affected	0.140 sec
185	17:14:45	INSERT INTO DEPT VALUES(NULL, 'eee')	1 row(s) affected	0.204 sec
186	17:14:57	SELECT * FROM DEPT LIMIT 0, 1000	4 row(s) returned	0.000 sec / 0.000 sec

```
--      3.2
CREATE TRIGGER ONDLT
AFTER DELETE
ON DEPT
FOR EACH ROW
DELETE FROM EMPLOYEE WHERE EMPLOYEE.DEPT_NO = OLD.DEPT_NO;


INSERT INTO EMPLOYEE
VALUES (1, 1, 'STUDENT', 10000),
     (2, 2, 'PROF', 100000);

DELETE FROM DEPT WHERE DEPT_NO=1;

select * from dept;
```



EMP_NO	DEPT_NO	JOB	SALARY
2	2	PROF	100000

#	Time	Action	Message	Duration / Fetch
195	17:18:29	CREATE TRIGGER ONDLT AFTER DELETE ON DEPT FOR EACH ROW DELETE FRO...	Error Code: 1363. There is no NEW row in on DELETE trigger	0.000 sec
196	17:18:36	CREATE TRIGGER ONDLT AFTER DELETE ON DEPT FOR EACH ROW DELETE FRO...	0 row(s) affected	0.187 sec
197	17:18:46	INSERT INTO DEPT VALUES (1, 'CSE'), (2, 'MME')	2 row(s) affected Records: 2 Duplicates: 0 Warnings: 0	0.203 sec
198	17:19:06	DELETE FROM DEPT WHERE DEPT_NO=1	1 row(s) affected	0.094 sec
199	17:19:09	select * from EMPLOYEE LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec

```
--      3.3
CREATE TRIGGER INSUPD
BEFORE INSERT
ON DEPT
FOR EACH ROW
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT ='AUTHORIZATION ERROR';
```

#	Time	Action	Message	Duration / Fetch
197	17:18:46	INSERT INTO DEPT VALUES (1, 'CSE'), (2, 'MME')	2 row(s) affected Records: 2 Duplicates: 0 Warnings: 0	0.203 sec
198	17:19:06	DELETE FROM DEPT WHERE DEPT_NO=1	1 row(s) affected	0.094 sec
199	17:19:09	select * from EMPLOYEE LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
200	17:20:24	CREATE TRIGGER INSUPD BEFORE INSERT ON DEPT FOR EACH ROW SIGNAL SQ...	0 row(s) affected	0.250 sec
201	17:20:28	INSERT INTO DEPT VALUES(5,'mECH')	Error Code: 1644. AUTHORIZATION ERROR	0.000 sec

Object Info Session