CSE 4308: Database Management Systems Lab

Lab-02

Group-B

Mohammod Mahbub Ur Rahman
220042148
BSc.in Software Engineering

Task-1

1.

```sql
-- Table creation
CREATE TABLE Lab1.STUDENT(
    ID INT NOT NULL,
    NAME VARCHAR(50),
    DEPT_NAME VARCHAR(30),
    TOT_CRED INT,
    PRIMARY KEY (ID),
    CONSTRAINT VALIDITY_CHECK CHECK (TOT_CRED <= 180)
);
```

creating the table using the ‘CREATE TABLE’. Adding column names and their data type. Also declaring ID as not null. Setting ID as primary key and checking if an entry has credit greater than or = 180.

2.inserting data in the following format.

```sql
-- Data insert
INSERT INTO STUDENT (ID, NAME, DEPT_NAME, TOT_CRED) VALUES
(00128, 'Zhang', 'Comp. Sci.', 102),
(12345, 'Shankar', 'Comp. Sci.', 32),
(19991, 'Brandt', 'History', 80),
(23121, 'Chavez', 'Finance', 110),
(44553, 'Peltier', 'Physics', 56),
(45678, 'Levy', 'Physics', 46),
(54321, 'Williams', 'Comp. Sci.', 5),
(55739, 'Sanchez', 'Music', 38),
(70557, 'Snow', 'Physics', 0),
(76543, 'Brown', 'Comp. Sci.', 58),
(76653, 'Aoi', 'Elec. Eng.', 60),
(98765, 'Bourikas', 'Elec. Eng.', 9),
(98988, 'Tanaka', 'Biology', 120);
```

3.
a> “SELECT * FROM STUDENT;”
it selects all the entry of the students in the student table and returns us all the entries with data.

b> “SELECT ID, NAME FROM STUDENT;”
it shows ID and name of the entries of the student table.

c> “SELECT NAME, DEPT_NAME FROM STUDENT
WHERE (TOT_CRED >=80 AND TOT_CRED <= 120);”

It shows us name and department name from the student table where the entry’s credit is between 80 and 120.

d> “SELECT ID, NAME FROM STUDENT WHERE DEPT_NAME = 'Comp. Sci.'”;

It shows us the ID and name from the student table which meet the condition where its department name is 'Comp. Sci.

e> “SELECT NAME, TOT_CRED FROM STUDENT WHERE DEPT_NAME = 'Physics';”

Same as the previous

f> “SELECT ID, NAME FROM STUDENT WHERE DEPT_NAME = 'Comp. Sci.' OR TOT_CRED < 10;”

Same as before but added an extra constraint where total credit is less than 10. Now it will show us the id and the name from the table where he is in 'Comp. Sci. or his credit is less than 10.

g> “SELECT DISTINCT DEPT_NAME FROM STUDENT;”

it gives us the distinct department name from the table. It will ignore duplicate data.

h> “DROP TABLE STUDENT CASCADE;”

it deletes the table with all its data. Cascade keyword deletes the data with constrains.

Task-2

1

```sql
-- ACCOUNT TABLE
CREATE TABLE ACCOUNT(
    ACCOUNT_NO CHAR(5) PRIMARY KEY,
    BALANCE INT NOT NULL
);

-- CUSTOMER TABLE
CREATE TABLE CUSTOMER(
    CUSTOMER_NO CHAR(5) PRIMARY KEY,
    CUSTOMER_NAME VARCHAR(20) NOT NULL,
    CUSTOMER_CITY VARCHAR(10)

);

-- DEPOSITOR TABLE
CREATE TABLE DEPOSITOR(
    ACCOUNT_NO CHAR(5),
    CUSTOMER_NO CHAR(5),
    PRIMARY KEY(ACCOUNT_NO, CUSTOMER_NO)
);

```

Created 3 tables and selected primary key.

2.
a> “ALTER TABLE CUSTOMER ADD DATE_OF_BIRTH DATE;”

added date of birth column in the customer table.

b> “ALTER TABLE ACCOUNT MODIFY BALANCE DECIMAL (12, 2);”

modified the balance data type from int to decimal where it will be 12 digits and 2 digits after decimal point.

c> “ALTER TABLE DEPOSITOR RENAME COLUMN ACCOUNT_NO TO A_NO;

ALTER TABLE DEPOSITOR RENAME COLUMN CUSTOMER_NO TO C_NO;”

Renamed the columns to A_NO and C_NO

d> “ALTER TABLE DEPOSITOR RENAME TO DEPOSITOR_INFO;”

Changed the table name to DEPOSITOR_INFO.

e> “ALTER TABLE DEPOSITOR_INFO ADD CONSTRAINT FK_DEPOSITOR_ACCOUNT FOREIGN KEY (A_NO)

```
REFERENCES ACCOUNT(ACCOUNT_NO) ON DELETE
CASCADE ON UPDATE CASCADE;

ALTER TABLE DEPOSITOR_INFO ADD CONSTRAINT
FK_DEPOSITOR_CUSTOMER FOREIGN KEY (C_NO)
REFERENCES CUSTOMER(CUSTOMER_NO) ON DELETE
CASCADE ON UPDATE CASCADE;”
```

Adding foreign key constrains and naming them. Here C_NO is foreign key and it is connected to CUSTOMER_NO. “ON DELETE CASCADE ON UPDATE CASCADE” it is used so that if any data containing foreign key is changed it will automatically update or delete the connected data.

3

```sql
INSERT INTO ACCOUNT (ACCOUNT_NO, BALANCE) VALUES
('SWE12', 4900),
('SWE13', 5900),
('SWE14', 6400),
('SWE15', 5600),
('SWE16', 3400),
('SWE17', 7800);

INSERT INTO CUSTOMER (CUSTOMER_NO, CUSTOMER_NAME, CUSTOMER_CITY, DATE_OF_BIRT
('CUS12', 'Mahbub', "KHL", '2002-09-10'),
('CUS13', 'Rahim', "RJH", '2004-07-21'),
('CUS14', 'Karim', "CTG", '2002-05-26'),
('CUS15', 'Rahman', "CHK", '2006-06-24'),
('CUS16', 'Abul', "KHL", '2004-07-10'),
('CUS17', 'Abdul', "DHK", '2001-04-16');

INSERT INTO DEPOSITOR_INFO (A_NO, C_NO) VALUES
('SWE12', 'CUS12'),
('SWE13', 'CUS13'),
('SWE14', 'CUS14'),
('SWE15', 'CUS15'),
('SWE16', 'CUS16'),
('SWE17', 'CUS17');
```

Inserted values into the tables.

4

a> “SELECT ACCOUNT_NO FROM ACCOUNT WHERE BALANCE < 100000;”

Shows us the account number from the account table where the balance is below 100000.

b> “SELECT CUSTOMER_NAME FROM CUSTOMER WHERE CUSTOMER_CITY = 'KHL';”

Shows us the names of the customers whose city is KHL

c> “SELECT * FROM ACCOUNT, DEPOSITOR_INFO;”

Shows us all info in the account table and the depositor table.

d> “SELECT * FROM CUSTOMER NATURAL JOIN DEPOSITOR_INFO;”

this NATURAL JOIN keyword joins the tables data. Every single row of depositor_info is added to the every customer rows.

e> “SELECT CUSTOMER_NAME, CUSTOMER_CITY FROM CUSTOMER, DEPOSITOR_INFO WHERE CUSTOMER.CUSTOMER_NO = DEPOSITOR_INFO.C_NO ;”

Here we are selecting name and city info from 2 tables where customer table’s customer_no matches depositor_info table’s c_no.

f> “SELECT * FROM CUSTOMER WHERE CUSTOMER_NO IN (SELECT C_NO FROM DEPOSITOR_INFO WHERE A_NO IN (SELECT ACCOUNT_NO FROM ACCOUNT WHERE BALANCE > 1000));”

it is a nested condition.

we are selecting everything from customer table where customer_no matches c_no from DEPOSITOR_INFO table where A_NO matches from ACCOUNT table where balance is greater than 1000

5

a> “UPDATE CUSTOMER SET CUSTOMER_CITY = 'KLN' WHERE CUSTOMER_CITY = 'KHL';”

we are updating the table data of CUSTOMER_CITY column where it is ’KHL’ we are updating it to ‘KLN’

b> “UPDATE CUSTOMER SET CUSTOMER_NO = 'CUS32' WHERE CUSTOMER_NO = 'CUS17';”

it will update the customer table data where the customer no is cus17 to cus32.

c> “DELETE FROM CUSTOMER WHERE CUSTOMER_NO = 'CUS31';”

deletes the entry where customer no is cus31

d>

```
80    -- 5.d
81    DELETE FROM ACCOUNT;
82    DELETE FROM CUSTOMER;
83    DELETE FROM DEPOSITOR_INFO;
```

deletes the table data without deleting the table itself.