Using SQL in MS Access

Himadri Barman

1. Creating Tables

♦Create a table PLAYERS. The fields and corresponding data types along with other requirements are:

Player_ID	*Text* of size 10, primary key
Player_Name	*Text* of size 100, mandatory field
Player_Add	*Text* of size 200
Player_DOB	Date/Time
Player_Weight	Number
Player_Height	Number
Player_Salary	Currency, Default salary is 50000
Player_Eligible	Yes/No
Player_Remarks	Memo

```
CREATE TABLE PLAYERS (Player_ID TEXT (10) PRIMARY KEY,
Player_Name TEXT (100) NOT NULL, Player_Add TEXT (200),
Player_DOB DATETIME, Player_Weight NUMBER, Player_Height NUMBER,
Player_Salary CURRENCY DEFAULT[1] 50000, Player_Eligible YESNO,
Player_Remarks MEMO)
```

2. Insert Data

♦Enter data in all the fields.

```
INSERT INTO PLAYERS VALUES ('123', 'RAM','GUWAHATI', '12/31/87',
67, 167, 4500, yes, 'captain')
```

♦Selectively enter data into some fields.

```
INSERT INTO PLAYERS (Player_ID, Player_Name) VALUES ('456',
'SHYAM')
```

3. Changing the Structure of the Table

♦Add an additional field *Player_Rating (data type is Number)* to the table PLAYERS.

```
ALTER TABLE PLAYERS ADD COLUMN Player_Rating NUMBER
```

[1] If the DEFAULT is not working, you have to change a setting in MS Access. Go to File -> Options -> Object Designers -> Query Design. Tick both the options under SQL Server Compatible Syntax (ANSI 92)

♦Delete the field *Player_Rating from* the table PLAYERS.

```
ALTER TABLE PLAYERS DROP COLUMN Player_Rating
```

♦What to do if you forget to add the primary key in the table PLAYER?

```
ALTER TABLE PLAYERS ADD PRIMARY KEY (Player_ID)
```

4. Updating Data

♦Update the address of the player whose id is 123 to DIBRUGARH.

UPDATE PLAYERS SET Player_Add = 'DIBRUGARH' WHERE Player_ID = '123'

5. Deleting Data

♦Delete player whose id is 456 from table PLAYERS.

```
DELETE FROM PLAYERS WHERE Player_ID = '456'
```

6. Creating Relationships

♦Create table TEAM and then create a 1 : M relationship between TEAM and PLAYERS. Enforce referential integrity. The table TEAM has the following fields:

Team_ID	Text of size 10, primary key
Team_Name	Text of size 100, mandatory
Team_Address	Text of size 200

```
CREATE TABLE TEAM (Team_ID TEXT (10) PRIMARY KEY, Team_Name TEXT
(100) NOT NULL, Team_Add TEXT (200))
```

```
ALTER TABLE PLAYERS ADD COLUMN Team_ID TEXT (10)
ALTER TABLE PLAYERS ADD FOREIGN KEY (Team_ID) REFERENCES TEAM
(Team_ID)
```

OR

```
ALTER TABLE PLAYERS ADD COLUMN Team_ID TEXT (10) REFERENCES TEAM
(Team_ID)
```

♦Create table SPONSOR and then create a 1 : 1 relationship between TEAM and SPONSOR (A team has only one lead sponsor and a sponsor can handle only one team as a lead sponsor. It is to be remembered that there are many sponsors whose data will be stored in the SPONSOR table). Enforce referential integrity. The table TEAM has the following fields:

Sponsor_ID	Text of size 10, primary key
Sponsor_Name	Text of size 100, mandatory
Sponsor_Address	Text of size 200
Sponsor_Amount	Currency, Default sponsorship amount is 100 thousand

```
CREATE TABLE SPONSOR (Sponsor_ID TEXT (10) PRIMARY KEY,
Sponsor_Name TEXT (100) NOT NULL, Sponsor_Add TEXT (200),
Sponsor_Amount CURRENCY DEFAULT 100000)
```

```
ALTER TABLE TEAM ADD COLUMN Sponsor_ID TEXT (10)
CREATE UNIQUE INDEX LEAD_SPONSOR ON TEAM (Sponsor_ID)
ALTER TABLE TEAM ADD FOREIGN KEY (Sponsor_ID) REFERENCES SPONSOR
(Sponsor_ID)
```

7. Retrieving Data

Before executing the following queries, see to it that sufficient data is there in both the tables or you'll be returned with no results.

♦Retrieve all data from table PLAYERS.

SELECT * FROM PLAYERS

♦Retrieve the name of all players from table PLAYERS. In the output, display field Player_Name as Name of Player.

```
SELECT Player_Name as [Name of Player], Player_Add FROM PLAYERS
ORDER BY Player_Name ASC
```

♦Retrieve the ids of the team distinctly (only once) from the table PLAYERS.

```
SELECT DISTINCT TEAM_ID FROM PLAYERS
```

♦Retrieve the total number of players per team and the salary paid to them from the table PLAYERS. In the output, display the output for players per team as Total Players and salary paid to them as Total Salary.

```
SELECT Team_ID, Count (Player_ID) as [Total Players], SUM(
Player_Salary) as [Total Salary]  FROM PLAYERS GROUP BY Team_ID
```

♦*As previous query*, but this time the team ids should be sorted in the descending order.

```
SELECT Team_ID, Count (Player_ID) as [Total Players], SUM(
Player_Salary) as [Total Salary]  FROM PLAYERS GROUP BY Team_ID
ORDER BY Team_ID DESC
```

♦Retrieve the names of all the players along with their address for team with id 001.

```
SELECT Player_Name as [Name of Player], Player_Add FROM PLAYERS
WHERE Team_ID ='001'
```

♦Retrieve the names of all the players along with their address for team with id 001 or 002.

```
SELECT Player_Name as [Name of Player], Player_Add FROM PLAYERS
WHERE Team_ID IN ('001','002')
```

♦Retrieve the names of all the players along with their address for team with id 001 or salary of player is greater than 3000.

```
SELECT Player_Name as [Name of Player], Player_Add FROM PLAYERS
WHERE Team_ID ='001' OR Player_Salary > 3000
```

♦Retrieve the names of all the players whose name starts with R.

```
SELECT Player_Name FROM PLAYERS WHERE Player_Name like 'R*'
```

♦Retrieve the names of all the players, their addresses and DOB whose DOB is between January 1, 1985 and January 1, 1988. The results should be ordered by player name in the ascending order.

```
SELECT Player_Name, Player_Add, Player_DOB FROM PLAYERS WHERE
Player_DOB BETWEEN #1/1/1985# AND #1/1/1988# ORDER BY
Player_Name ASC
```

♦Retrieve the names of all the players, their address and the team name.

```
SELECT Player_Name, Player_Add, Team_Name FROM PLAYERS INNER
JOIN TEAM ON PLAYERS.Team_ID=TEAM.Team_ID
```

(The INNER JOIN *keyword selects all rows from both tables as long as there is a match between the columns in both tables.)*

♦Retrieve the names of all the players, their address and the team name from the team with id 001.

```
SELECT Player_Name, Player_Add, Team_Name FROM PLAYERS INNER
JOIN TEAM ON PLAYERS.Team_ID=TEAM.Team_ID WHERE
PLAYERS.Team_ID='001'
```

♦A Left Outer Join Query

```
SELECT Player_Name, Player_Add, Team_Name FROM PLAYERS LEFT
OUTER JOIN TEAM ON PLAYERS.Team_ID=TEAM.Team_ID
```

(The LEFT OUTER JOIN keyword returns all rows from the left table (table1), with the matching rows in the right table (table2). The result is NULL in the right side when there is no match. A left outer join retains all of the rows of the "left" table, regardless of whether there is a row that matches on the "right" table.)

♦A Right Outer Join Query

```
SELECT Player_Name, Player_Add, Team_Name FROM PLAYERS RIGHT
OUTER JOIN TEAM ON PLAYERS.Team_ID=TEAM.Team_ID
```

The RIGHT OUTER JOIN keyword returns all rows from the right table (table2), with the matching rows in the left table (table1). The result is NULL in the left side when there is no match.

♦Retrieve the information as to how to many players are there in the PLAYERS table

```
SELECT COUNT(*) as 'Number of Players' FROM PLAYERS
```

♦Find the average salary of the players who are there in the PLAYERS table

```
SELECT AVG(Player_Salary) as 'Average Salary of Players' FROM
PLAYERS
```

8. Deleting Tables

♦Delete the table PLAYERS.

```
DROP TABLE PLAYERS
```