

## DATA MANAGEMENT AND DATABASE DESIGN HOMEWORK: WEEK- 9

FIGURE 7-15 Adult literacy program (for Problems and Exercises 6–14)

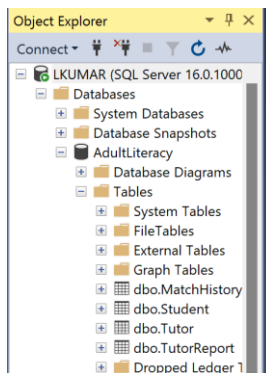
TUTOR (TutorID, CertDate, Status)		
TutorID	CertDate	Status
100	1/05/2008	Active
101	1/05/2008	Temp Stop
102	1/05/2008	Dropped
103	5/22/2008	Active
104	5/22/2008	Active
105	5/22/2008	Temp Stop
106	5/22/2008	Active

STUDENT (StudentID, Read)		
StudentID	Read	
3000	2.3	
3001	5.6	
3002	1.3	
3003	3.3	
3004	2.7	
3005	4.8	
3006	7.8	
3007	1.5	

MATCH HISTORY (MatchID, TutorID, StudentID, StartDate, EndDate)				
MatchID	TutorID	StudentID	StartDate	EndDate
1	100	3000	1/10/2008	
2	101	3001	1/15/2008	5/15/2008
3	102	3002	2/10/2008	3/01/2008
4	106	3003	5/28/2008	
5	103	3004	6/01/2008	6/15/2008
6	104	3005	6/01/2008	6/28/2008
7	104	3006	6/01/2008	

TUTOR REPORT (MatchID, Month, Hours, Lessons)			
MatchID	Month	Hours	Lessons
1	6/08	8	4
4	6/08	8	6
5	6/08	4	4
4	7/08	10	5
1	7/08	4	2

1. Create a database named “AdultLiteracy” on your RDBMS environment. Using Figure 7-5 above, write DDL commands to create table structures for each entity above. Name your tables the following names: Tutor, Student, MatchHistory, TutorReport



```
--vs2178.sql - LKU...LKUMAR\laksh (53)*
-- Query to create a database with the name "AdultLiteracy"
CREATE DATABASE AdultLiteracy;
go

--Query to use the database "AdultLiteracy"
USE AdultLiteracy
go

--Query to create a new table "Tutor" and add rows
CREATE TABLE Tutor
(
    TutorID int not null,
    CertDate date,
    Status varchar(30),
    CONSTRAINT Tutor_PK primary key (TutorID )
);

--Query to create a new table "MatchHistory" and add rows
CREATE TABLE MatchHistory
(
    MatchID int not null,
    TutorID int not null,
    StudentID int not null,
    StartDate varchar(25),
    EndDate varchar(25),
    CONSTRAINT MatchHistory_PK primary key (MatchID )
);

--Query to create a new table "Student" and add rows
CREATE TABLE Student
(
    StudentID int not null,
    "Read" float,
    CONSTRAINT Student_PK primary key (StudentID )
);

CREATE TABLE TutorReport
(
    MatchID int not null,
    Month date,
    Hours int,
    Lessons int,
    CONSTRAINT TutorReport_PK primary key (MatchID,Month)
);
```

**QUERY:**

-- Query to create a database with the name "AdultLiteracy"

```
CREATE DATABASE AdultLiteracy;  
go
```

--Query to use the database "AdultLiteracy"

```
USE AdultLiteracy  
go
```

--Query to create a new table "Tutor" and add rows

```
CREATE TABLE Tutor  
(  
TutorID int not null,  
CertDate date,  
Status varchar (30),  
CONSTRAINT Tutor_PK primary key (TutorID)  
);
```

--Query to create a new table "MatchHistory" and add rows

```
CREATE TABLE MatchHistory  
(  
MatchID int not null,  
TutorID int not null,  
StudentID int not null,  
StartDate varchar (25),  
EndDate varchar (25),  
CONSTRAINT MatchHistory_PK primary key (MatchID)  
);
```

--Query to create a new table "Student" and add rows

```
CREATE TABLE Student  
(  
StudentID int not null,  
"Read" float,  
CONSTRAINT Student_PK primary key (StudentID)  
);
```

--Query to create a new table "TutorReport" and add rows

```
CREATE TABLE TutorReport  
(  
MatchID int not null,  
Month date,  
Hours int,  
Lessons int,  
CONSTRAINT TutorReport_PK primary key (MatchID, Month)  
);
```

2. Write SQL scripts to insert sample data from Fig 7-5 into the database.

**Dbo.Tutor**

**QUERY:**

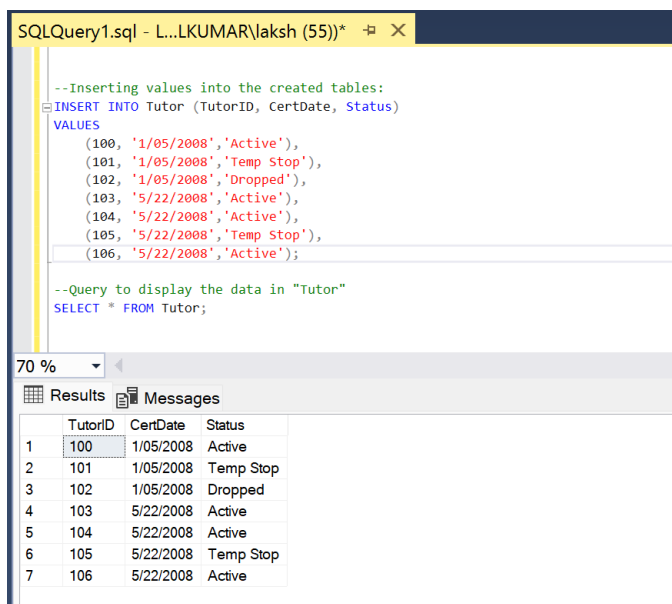
--Inserting values into the created tables:

**INSERT INTO** Tutor (TutorID, CertDate, Status)  
**VALUES**

(100, '1/05/2008', 'Active'),  
(101, '1/05/2008', 'Temp Stop'),  
(102, '1/05/2008', 'Dropped'),  
(103, '5/22/2008', 'Active'),  
(104, '5/22/2008', 'Active'),  
(105, '5/22/2008', 'Temp Stop'),  
(106, '5/22/2008', 'Active');

--Query to display the data in "Tutor"

**SELECT \*** **FROM** Tutor;



SQLQuery1.sql - L...LKUMAR\laksh (55)\*

```
--Inserting values into the created tables:  
INSERT INTO Tutor (TutorID, CertDate, Status)  
VALUES  
(100, '1/05/2008', 'Active'),  
(101, '1/05/2008', 'Temp Stop'),  
(102, '1/05/2008', 'Dropped'),  
(103, '5/22/2008', 'Active'),  
(104, '5/22/2008', 'Active'),  
(105, '5/22/2008', 'Temp Stop'),  
(106, '5/22/2008', 'Active');  
  
--Query to display the data in "Tutor"  
SELECT * FROM Tutor;
```

70 %

Results Messages

	TutorID	CertDate	Status
1	100	1/05/2008	Active
2	101	1/05/2008	Temp Stop
3	102	1/05/2008	Dropped
4	103	5/22/2008	Active
5	104	5/22/2008	Active
6	105	5/22/2008	Temp Stop
7	106	5/22/2008	Active

## Dbo. MatchHistory

### QUERY:

--Inserting values into the created tables:

```
INSERT INTO MatchHistory (MatchID, TutorID, StudentID, StartDate, EndDate)
VALUES
```

```
(1, 100, 3000, '1/10/2008', ''),
(2, 101, 3001, '1/15/2008', '5/15/2008'),
(3, 102, 3002, '2/10/2008', '3/01/2008'),
(4, 103, 3003, '5/28/2008', ''),
(5, 104, 3004, '6/01/2008', '6/15/2008'),
(6, 105, 3005, '6/01/2008', '6/28/2008'),
(7, 106, 3006, '6/10/2008', '');
```

--Query to display the data in "MatchHistory"

```
SELECT * FROM MatchHistory;
```

SQLQuery1.sql - L...LKUMAR\laksh (55)\*

```
--Inserting values into the created tables:
INSERT INTO MatchHistory(MatchID, TutorID, StudentID, StartDate, EndDate)
VALUES
    (1, 100, 3000, '1/10/2008', ''),
    (2, 101, 3001, '1/15/2008', '5/15/2008'),
    (3, 102, 3002, '2/10/2008', '3/01/2008'),
    (4, 103, 3003, '5/28/2008', ''),
    (5, 104, 3004, '6/01/2008', '6/15/2008'),
    (6, 105, 3005, '6/01/2008', '6/28/2008'),
    (7, 106, 3006, '6/10/2008', '');

--Query to display the data in "MatchHistory"
SELECT * FROM MatchHistory;
```

70 %

Results Messages

	MatchID	TutorID	StudentID	StartDate	EndDate
1	1	100	3000	1/10/2008	
2	2	101	3001	1/15/2008	5/15/2008
3	3	102	3002	2/10/2008	3/01/2008
4	4	103	3003	5/28/2008	
5	5	104	3004	6/01/2008	6/15/2008
6	6	105	3005	6/01/2008	6/28/2008
7	7	106	3006	6/10/2008	

Db0. Student

QUERY:

--Inserting values into the created tables:

INSERT INTO Student (StudentID, "Read")  
VALUES

(3000,2.3),  
(3001,5.6),  
(3002,1.3),  
(3003,3.3),  
(3004,2.7),  
(3005,4.8),  
(3006,7.8),  
(3007,1.5);

--Query to display the data in "Student"

SELECT \* FROM Student;

The screenshot shows a SQL Server Enterprise Manager interface. At the top, a tab is labeled 'SQLQuery1.sql - L...LKUMAR\laksh (55)\*'. The main area contains the following SQL script:

```
--Inserting values into the created tables:  
INSERT INTO Student(StudentID, "Read")  
VALUES  
    (3000,2.3),  
    (3001,5.6),  
    (3002,1.3),  
    (3003,3.3),  
    (3004,2.7),  
    (3005,4.8),  
    (3006,7.8),  
    (3007,1.5);  
  
--Query to display the data in "Student"  
SELECT * FROM Student;
```

Below the script, the 'Results' pane shows the output of the query. It contains a table with two columns: 'StudentID' and 'Read'. The table has 8 rows of data, with the first row highlighted.

	StudentID	Read
1	3000	2.3
2	3001	5.6
3	3002	1.3
4	3003	3.3
5	3004	2.7
6	3005	4.8
7	3006	7.8
8	3007	1.5

## Db0. TutorReport

### QUERY:

--Inserting values into the created tables:

```
INSERT INTO TutorReport (MatchID, Month, Hours, Lessons)  
VALUES
```

```
(1, '6/08/2008', 8, 4),  
(4, '6/08/2008', 8, 6),  
(5, '6/08/2008', 4, 4),  
(4, '7/08/2008', 10, 5),  
(1, '7/08/2008', 4, 2);
```

--Query to display the data in "Student"

```
SELECT * FROM TutorReport;
```

The screenshot shows a SQL IDE window titled '~vs217B.sql - LKU...LKUMAR\laksh (53))'. The editor contains the following SQL code:

```
--Inserting values into the created tables:  
INSERT INTO TutorReport(MatchID,Month,Hours,Lessons)  
VALUES  
    (1, '6/08/2008', 8, 4),  
    (4, '6/08/2008', 8, 6),  
    (5, '6/08/2008', 4, 4),  
    (4, '7/08/2008', 10, 5),  
    (1, '7/08/2008', 4, 2);  
  
--Query to display the data in "TutorReport"  
SELECT * FROM TutorReport;
```

Below the editor, the 'Results' tab is active, displaying a table with 5 rows and 5 columns: MatchID, Month, Hours, and Lessons. The data is as follows:

	MatchID	Month	Hours	Lessons
1	1	2008-06-08	8	4
2	1	2008-07-08	4	2
3	4	2008-06-08	8	6
4	4	2008-07-08	10	5
5	5	2008-06-08	4	4

3. Write the SQL command to add MATH SCORE to the STUDENT table.

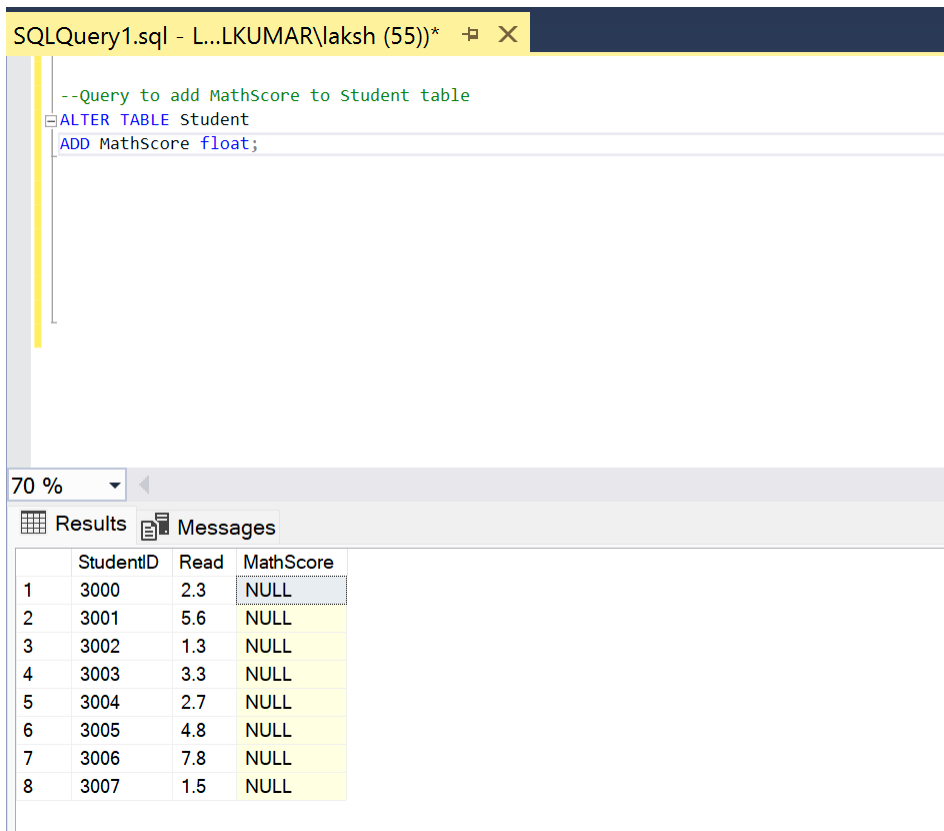
**QUERY:**

--Query to add MathScore to Student table

```
ALTER TABLE Student  
ADD MathScore float;
```

--Query to display the data in "Student"

```
SELECT * FROM Student;
```



The screenshot shows a SQL Server Enterprise Manager interface. At the top, a query window titled 'SQLQuery1.sql - L...LKUMAR\laksh (55))' contains the following SQL code:

```
--Query to add MathScore to Student table  
ALTER TABLE Student  
ADD MathScore float;
```

Below the query window, the 'Results' tab is selected, displaying the output of the query. The results are shown in a table with 4 columns: 'StudentID', 'Read', 'MathScore', and an unnamed column. The data is as follows:

	StudentID	Read	MathScore	
1	3000	2.3	NULL	
2	3001	5.6	NULL	
3	3002	1.3	NULL	
4	3003	3.3	NULL	
5	3004	2.7	NULL	
6	3005	4.8	NULL	
7	3006	7.8	NULL	
8	3007	1.5	NULL	

4. Write the SQL command to add SUBJECT to TUTOR. The only values allowed for SUBJECT will be Reading, Math, and ESL.

**QUERY:**

--Query to Add column "Subject" to Tutor table

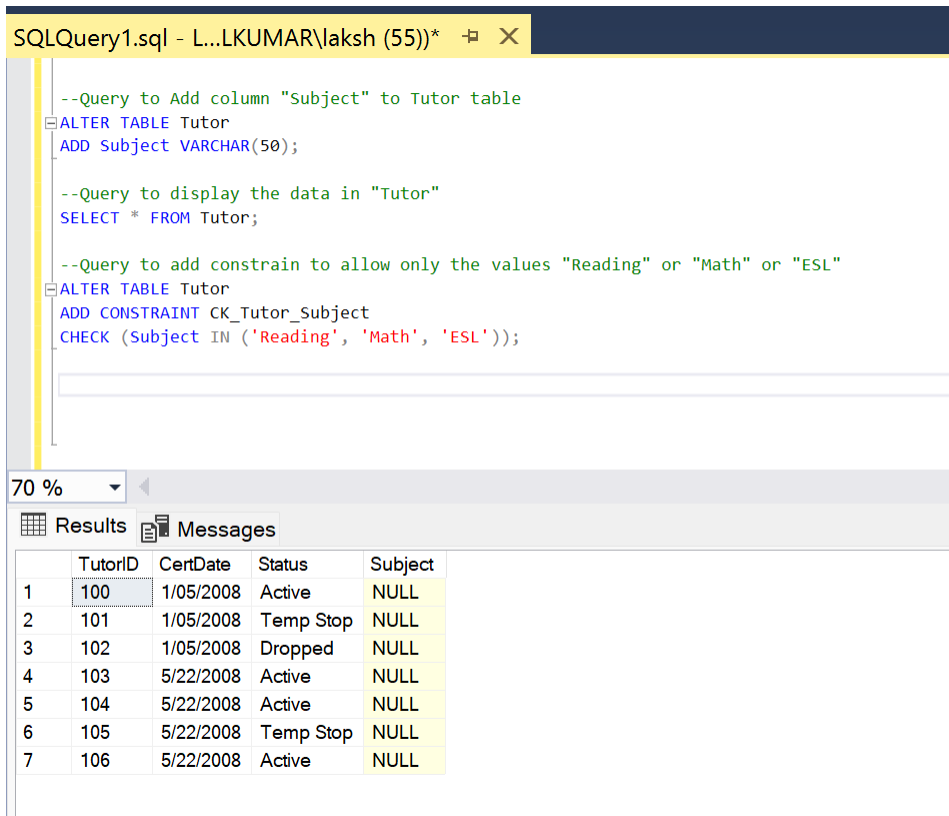
```
ALTER TABLE Tutor  
ADD Subject VARCHAR (50);
```

--Query to display the data in "Tutor"

```
SELECT * FROM Tutor;
```

--Query to add constrain to allow only the values "Reading" or "Math" or "ESL"

```
ALTER TABLE Tutor  
ADD CONSTRAINT CK_Tutor_Subject  
CHECK (Subject IN ('Reading', 'Math', 'ESL'));
```



The screenshot shows a SQL Query Editor window titled "SQLQuery1.sql - L...LKUMAR\laksh (55))\*" with the following SQL commands:

```
--Query to Add column "Subject" to Tutor table  
ALTER TABLE Tutor  
ADD Subject VARCHAR(50);  
  
--Query to display the data in "Tutor"  
SELECT * FROM Tutor;  
  
--Query to add constrain to allow only the values "Reading" or "Math" or "ESL"  
ALTER TABLE Tutor  
ADD CONSTRAINT CK_Tutor_Subject  
CHECK (Subject IN ('Reading', 'Math', 'ESL'));
```

Below the editor is a Results window showing a table with 7 rows and 5 columns: TutorID, CertDate, Status, and Subject. The Subject column contains NULL values for all rows.

	TutorID	CertDate	Status	Subject
1	100	1/05/2008	Active	NULL
2	101	1/05/2008	Temp Stop	NULL
3	102	1/05/2008	Dropped	NULL
4	103	5/22/2008	Active	NULL
5	104	5/22/2008	Active	NULL
6	105	5/22/2008	Temp Stop	NULL
7	106	5/22/2008	Active	NULL



5. What do you need to do if a tutor signs up and wants to tutor in both reading and math? (Don't need to write SQL).

**Sol:**

Create two separate records for the same professor in Tutor Table and add Reading in "Subject" for one record and add Math in "Subject" for another record for the same Tutor.

**Eg:** If the tutor with ID 101, teaches both Math and Reading, populate following values in the Tutor table.

TutorID	CertDate	Status	Subject
101	1/05/2008	Active	Reading
101	1/05/2008	Active	Math

6. Write the SQL command to find any tutors who have not submitted a report for July.

**QUERY:**

--Query to fetch all the tutors who have not filed a report for the month of July

```
SELECT T.TutorID
FROM Tutor T
WHERE NOT EXISTS (
    SELECT 1
    FROM MatchHistory M
    LEFT JOIN TutorReport TR ON M.MatchID = TR.MatchID
    WHERE T.TutorID = M.TutorID
    AND TR.Month BETWEEN '2008-07-01' AND '2008-07-31');
```

The screenshot shows a SQL IDE window titled '~vs217B.sql - LKU...LKUMAR\laksh (53))'. The query is pasted into the editor. Below the editor, the 'Results' tab is active, displaying a table with 5 rows and 1 column, 'TutorID'. The rows contain the values 101, 102, 104, 105, and 106.

```
--Query to fetch all the tutors who have not filed a report for the month of July

SELECT T.TutorID
FROM Tutor T
WHERE NOT EXISTS (
    SELECT 1
    FROM MatchHistory M
    LEFT JOIN TutorReport TR ON M.MatchID = TR.MatchID
    WHERE T.TutorID = M.TutorID
    AND TR.Month BETWEEN '2008-07-01' AND '2008-07-31');
```

TutorID
101
102
104
105
106