Answers:

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

-- DATA BASE CREATION

**CREATE** **DATABASE** AdultLiteracy

A screenshot of a cell phone

Description automatically generated

--DDL STATEMENTS

**CREATE** **TABLE** Tutor (

TutorID **INTEGER**,

CertDate **date**,

status **NVARCHAR**(25),

**CONSTRAINT** tutor\_PK **PRIMARY** **KEY**(TutorID));

**CREATE** **TABLE** STUDENT(

studentID **INTEGER**,

[read] **nvarchar**(25),

**CONSTRAINT** student\_PK **PRIMARY** **KEY**(studentID)

);

**CREATE** **TABLE** MatchHistory(

MatchID **INTEGER**,

TutorID **INTEGER**,

StudentID **INTEGER**,

StartDate **date**,

EndDate **date**,

**CONSTRAINT** MatchHistory\_PK **PRIMARY** **KEY**(MatchID),

**CONSTRAINT** Tutor\_FK **FOREIGN** **KEY** (TutorID) **REFERENCES** TUTOR (TutorID),

**CONSTRAINT** Student\_FK **FOREIGN** **KEY** (StudentID) **REFERENCES** STUDENT (StudentID)

);

**CREATE** **TABLE** TutorReport(

MatchID **int** **not** **null**,

[Month] **DATE** **not** **null**,

[Hours] **INT**,

Lessons **INT**,

**CONSTRAINT** PK\_TutorReport **PRIMARY** **KEY**(MatchID,[Month])

);

A screenshot of a social media post

Description automatically generated

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

**INSERT** **INTO** Tutor(TutorID,CertDate,[Status])**values**(100,'1/5/2008','Active');

**INSERT** **INTO** Tutor(TutorID,CertDate,[Status])**values** (101,'1/5/2008','Temp Stop');

**INSERT** **INTO** Tutor(TutorID,CertDate,[Status])**values** (102,'1/5/2008','Dropped');

**INSERT** **INTO** Tutor(TutorID,CertDate,[Status])**values** (103,'5/22/2008','Active');

**INSERT** **INTO** Tutor(TutorID,CertDate,[Status])**values** (104,'5/22/2008','Active');

**INSERT** **INTO** Tutor(TutorID,CertDate,[Status])**values** (105,'5/22/2008','Temp Stop');

**INSERT** **INTO** Tutor(TutorID,CertDate,[Status])**values**(106,'5/22/2008','Active');

**INSERT** **INTO** Student(StudentID,[Read])**Values** (3000,2.3);

**INSERT** **INTO** Student(StudentID,[Read])**Values** (3001,5.6);

**INSERT** **INTO** Student(StudentID,[Read])**Values** (3002,1.3);

**INSERT** **INTO** Student(StudentID,[Read])**Values** (3003,3.3);

**INSERT** **INTO** Student(StudentID,[Read])**Values** (3004,2.7);

**INSERT** **INTO** Student(StudentID,[Read])**Values** (3005,4.8);

**INSERT** **INTO** Student(StudentID,[Read])**Values** (3006,7.8);

**INSERT** **INTO** Student(StudentID,[Read])**Values** (3007,1.5);

**INSERT** **INTO** MatchHistory **VALUES** (1,100,3000,'1/10/2008',**NULL**);

**INSERT** **INTO** MatchHistory **VALUES** (2,101,3001,'1/15/2008','5/15/2008');

**INSERT** **INTO** MatchHistory **VALUES** (3,102,3002,'2/10/2008','3/01/2008');

**INSERT** **INTO** MatchHistory **VALUES** (4,106,3003,'5/28/2008',**NULL**);

**INSERT** **INTO** MatchHistory **VALUES** (5,103,3004,'6/01/2008','6/15/2008');

**INSERT** **INTO** MatchHistory **VALUES** (6,104,3005,'6/01/2008','6/28/2008');

**INSERT** **INTO** MatchHistory **VALUES** (7,104,3006,'6/01/2008',**NULL**);

**INSERT** **INTO** TutorReport **VALUES** (1,'06/08/2008',8,4);

**INSERT** **INTO** TutorReport **VALUES** (4,'06/08/2008',8,6);

**INSERT** **INTO** TutorReport **VALUES** (5,'06/08/2008',4,4);

**INSERT** **INTO** TutorReport **VALUES** (4,'07/08/2008',10,5);  
**INSERT** **INTO** TutorReport **VALUES** (1,'06/08/2008',4,2);

A screenshot of a social media post

Description automatically generated

1. 7. Write the SQL command to add MATH SCORE to the STUDENT table.  
   **ALTER** **TABLE** STUDENT **ADD** MathScore **varchar**(25) ;

A screenshot of a cell phone

Description automatically generated

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

A screenshot of a social media post

Description automatically generated

**ALTER** **TABLE** TUTOR **ADD** subject **varchar**(25) **CHECK** ([Subject] = 'Reading' **and** [Subject] = 'Math' **and** [Subject] = 'ESL')

1. 9. 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).

To implement such scenario, we need to add additional tables below are the details for same  
Tutor Entity with Attributes: TutorID and Status  
Subject Entity with Attributes : SubjectID, SubjectName  
TutorSubject Entity with Attributes : TutorID, CertDate, SubjectID

These all will help to add both reading and math against a tutor.

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

**select** tutorID **from** MatchHistory **where** MatchID **in** (**select** MatchID **from** TutorReport **where** [MONTH] = '2008-07-08')

A screenshot of a social media post

Description automatically generated

1. Where do you think student and tutor information such as name, address, phone, and e-mail should be kept? Write the necessary SQL commands to capture this information. Make up sample data to populate your newly created table.

Student information should be kept in student table and tutor information should be kept in tutor table.

STUDENT

**ALTER** **TABLE** STUDENT **ADD** name **varchar**(25);

**ALTER** **TABLE** STUDENT **ADD** address **varchar**(25);

**ALTER** **TABLE** STUDENT **ADD** Phone **varchar**(25);

**ALTER** **TABLE** STUDENT **ADD** email **varchar**(25);

TUTOR

**ALTER** **TABLE** TUTOR **ADD** name **varchar**(25);

**ALTER** **TABLE** TUTOR **ADD** address **varchar**(25);

**ALTER** **TABLE** TUTOR **ADD** Phone **varchar**(25);

**ALTER** **TABLE** TUTOR **ADD** email **varchar**(25);

A screenshot of a social media post

Description automatically generated

A screenshot of a cell phone

Description automatically generated

STUDENT

A screenshot of a cell phone

Description automatically generated

TUTOR

A screenshot of a cell phone

Description automatically generated

1. List all active students in June by name. (the names you made up above). Include the number of hours students received tutoring and how many lessons they completed. Write the SQL command.

**select** s.name, mh.studentID ,[hours], lessons **from** student s, MatchHistory mh **inner** **join** TutorReport tr **on** mh.matchID= tr.matchID

**where** tr.[month] = '06/08/2008'**and** s.studentID=mh.studentID

A screenshot of a social media post

Description automatically generated

1. Which tutors, by name, are available to tutor? Write the SQL command.

**SELECT** name **FROM** TUTOR **WHERE** [status] = 'Active' **group** **by** name;

A screenshot of a cell phone

Description automatically generated

1. Which tutor needs to be reminded to turn in reports? Write the SQL command.

**select** **DISTINCT** tutorID **from** MatchHistory

**where** matchID **not** **in**(**select** matchID **from** TutorReport )

A screenshot of a social media post

Description automatically generated

1. Create a stored procedure that returns a result set of all tutors that are available to tutor. No input parameters required

**CREATE** **PROCEDURE** GetActiveTutor

**AS**

**BEGIN**

**SELECT** name **FROM** TUTOR **WHERE** [status] = 'Active' **group** **by** name;

**END**

A screenshot of a social media post

Description automatically generated

A screenshot of a cell phone

Description automatically generated

1. Create a User Define Function which determine if a specific tutor is available to tutor. The function takes a TutorID as an input value and returns a scaler value of ‘Y’ or ‘N’ corresponding to the tutor availability

**CREATE** **FUNCTION** ActiveTutorDetails(@TutorID **INT**)

**RETURNS** **varchar**(25)

**AS**

**BEGIN**

**DECLARE**

@statusValue **varchar**(25)

**IF** @TutorID **IN** (100,103,104,106)

**SET** @statusValue = 'Y'

**ELSE**

**SET** @statusValue = 'N'

**RETURN** (@statusValue)

**END**;

A screenshot of a social media post

Description automatically generated

A screenshot of a social media post

Description automatically generated

**A screenshot of a social media post

Description automatically generated**