Readings:

- chapter 22: “Business Intelligence: An Introduction” (Petkovic)
- chapter 9: “Data Warehousing” (Hoffer, Ramesh, & Topi) from page 375 - 394

Homework:

- chapter 9: “Data Warehousing” (Hoffer, Ramesh, & Topi)
  - Problems and Exercises 1, 2, 3

1. Examine the three tables with student data shown in Figure 9-1. Design a single-table format that will hold all of the data (nonredundantly) that are contained in these three tables. Choose column names that you believe are most appropriate for these data.

   StudentNo, LastName, MI, FirstName, Telephone, Status, Address, Dept, Hours, Insurance

2. Your assignment involves two parts:

   a. Construct tables for 06/21/2010 and 06/22/2010, reflecting these transactions; assume that the data are transient (refer to Figure 9-7).

06/20/2010:

Key	**Name**	**Major**
001	**Amy**	**Music**
002	**Tom**	**Business**
003	**Sue**	**Art**
004	**Joe**	**Math**
005	**Ann**	**Engineering**

06/21/2010:

Key	**Name**	**Major**
001	**Amy**	**Music**
002	**Tom**	**Business**
003	**Sue**	**Art**

004	Joe	Business
006	Jim	Phys Ed

06/22/2010:

Key	Name	Major
001	Amy	Music
002	Tom	Business
003	Sue	History
004	Joe	Business
006	Jim	Basket Weaving

b. Construct tables for 06/21/2010 and 06/22/2010, reflecting these transactions; assume that the data are periodic (refer to Figure 9-8).

06/20/2010:

Key	Date	Name	Major	Action
001	06/20	Amy	Music	C
002	06/20	Tom	Business	C
003	06/20	Sue	Art	C
004	06/20	Joe	Math	C
005	06/20	Ann	Engineering	C

06/21/2010:

Key	Date	Name	Major	Action
001	06/20	Amy	Music	C
002	06/20	Tom	Business	C
003	06/20	Sue	Art	C
004	06/20	Joe	Math	C

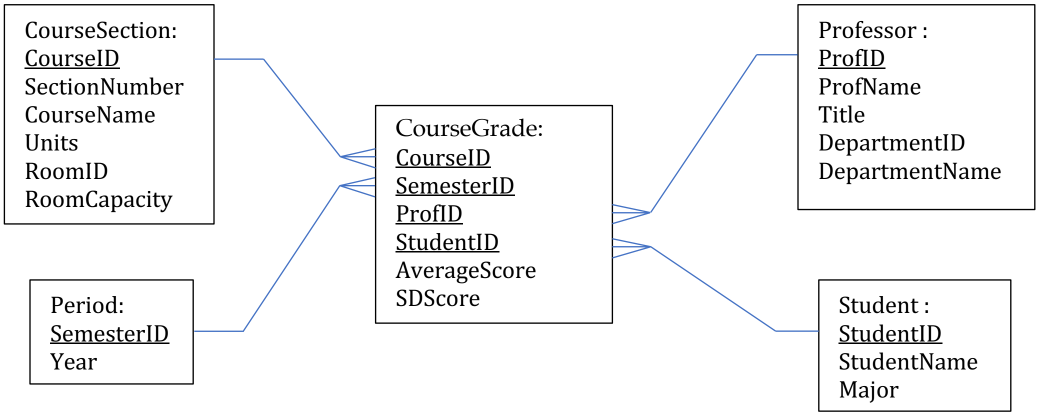
004	**06/21**	**Joe**	**Business**	**U**
005	**06/20**	**Ann**	**Engineering**	**C**
005	**06/21**	**Ann**	**Engineering**	**D**
006	**06/21**	**Jim**	**Phys Ed**	**C**

06/22/2010:

Key	**Date**	**Name**	**Major**	**Action**
001	**06/20**	**Amy**	**Music**	**C**
002	**06/20**	**Tom**	**Business**	**C**
003	**06/20**	**Sue**	**Art**	**C**
003	**06/22**	**Sue**	**History**	**U**
004	**06/20**	**Joe**	**Math**	**C**
004	**06/21**	**Joe**	**Business**	**U**
005	**06/20**	**Ann**	**Engineering**	**C**
005	**06/21**	**Ann**	**Engineering**	**D**
006	**06/21**	**Jim**	**Phys Ed**	**C**
006	**06/22**	**Jim**	**Basket Weaving**	**U**

3. The only fact that is to be recorded in the fact table is CourseGrade.

a. Design a star schema for this problem. See Figure 9-10 for the format you should follow.



b. Estimate the number of rows in the fact table, using the assumptions stated previously.

   930.

c. Estimate the total size of the fact table (in bytes), assuming that each field has an average of 5 bytes.

   4650B.

d. If you didn't want to or didn't have to stick with a strict star schema for this data mart, how would you change the design? Why?

   Combine the tables as much as possible to avoid redundant data and save space.

e. Various characteristics of sections, professors, and students change over time. How do you propose designing the star schema to allow for these changes? Why?

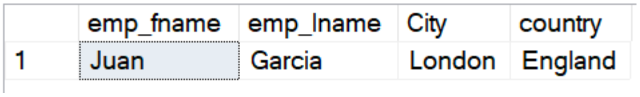
   Record the average, range, and standard deviation instead of the actual data to avoid the great change.

- Using the sample database from prior lecture, write the following persistent stored modules, and write commands to execute each:

1. Create a stored procedure that takes Country as an input parameter, and outputs a list of employees in that country.
   a. Procedure Name: GetEmployees @Country.
   b. Columns: First Name, Last Name, City, Country

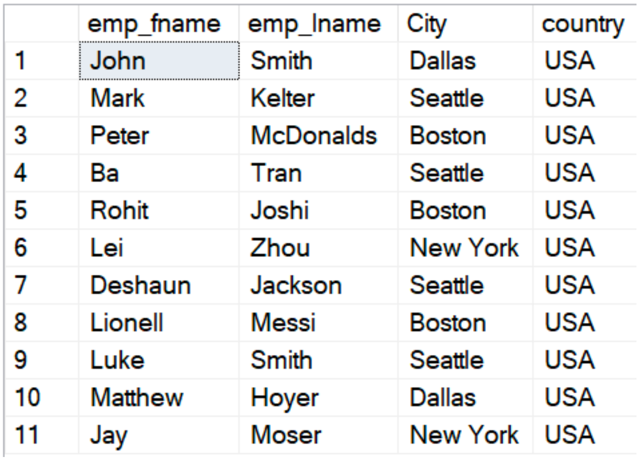
```
create procedure GetEmployees @Country varchar(10)
as
begin
       if @Country = 'England'
       begin
       select e.[emp_fname], e.[emp_lname], d.[Location] As City, @Country as
country
       from employee e join department d on e.dept_no = d.dept_no
       where d.location = 'London'
       end
       if @Country = 'USA'
       begin
       select e.[emp_fname], e.[emp_lname], d.[Location] As City, @Country as
country
```

```
        from employee e join department d on e.dept_no = d.dept_no
        where d.location <> 'London'
        end
end

GetEmployees 'England'
```



	emp_fname	emp_lname	City	country
1	Juan	Garcia	London	England

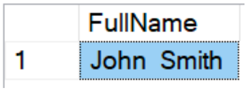
```
GetEmployees 'USA'
```



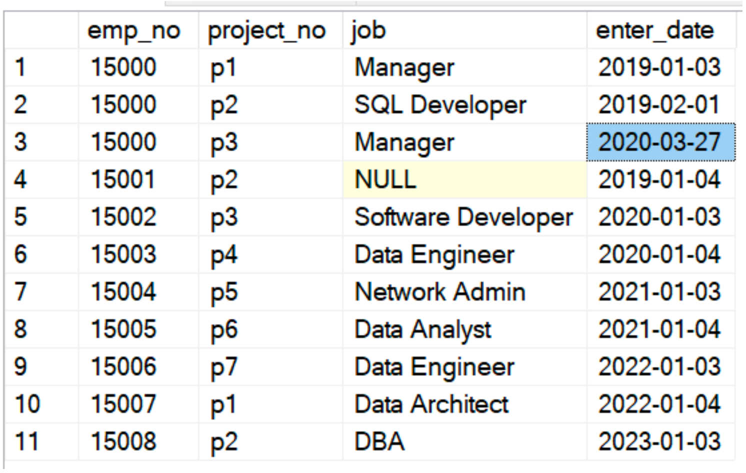
	emp_fname	emp_lname	City	country
1	John	Smith	Dallas	USA
2	Mark	Kelter	Seattle	USA
3	Peter	McDonalds	Boston	USA
4	Ba	Tran	Seattle	USA
5	Rohit	Joshi	Boston	USA
6	Lei	Zhou	New York	USA
7	Deshaun	Jackson	Seattle	USA
8	Lionell	Messi	Boston	USA
9	Luke	Smith	Seattle	USA
10	Matthew	Hoyer	Dallas	USA
11	Jay	Moser	New York	USA

2. Write a stored procedure AssignWork which takes emp_no, project_no, job as input parameters. It must add a record to the works_on table with enterd_date = system date, and returns the employee full name as an output parameter

```
create procedure AssignWork @emp_no int, @project_no char(4), @job varchar(50)
as
declare @FullName varchar(50)
begin
        insert into works_on values(@emp_no,@project_no,@job,GETDATE())

        select @FullName = [emp_fname] + ' ' + [emp_lname]
        from employee
        where emp_no = @emp_no

        select @FullName as FullName
end


exec AssignWork 15000,'p3','Manager'
```



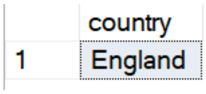
	FullName
1	John Smith



	emp_no	project_no	job	enter_date
1	15000	p1	Manager	2019-01-03
2	15000	p2	SQL Developer	2019-02-01
3	15000	p3	Manager	2020-03-27
4	15001	p2	NULL	2019-01-04
5	15002	p3	Software Developer	2020-01-03
6	15003	p4	Data Engineer	2020-01-04
7	15004	p5	Network Admin	2021-01-03
8	15005	p6	Data Analyst	2021-01-04
9	15006	p7	Data Engineer	2022-01-03
10	15007	p1	Data Architect	2022-01-04
11	15008	p2	DBA	2023-01-03

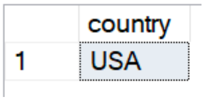
3. Create a UDF that takes @City as an input parameter and returns country. Procedure Name: GetCountry (@city)

```
create function GetCountry
(@City varchar(25))
returns varchar(10)
as
begin
       declare @Country varchar(10)

       if @City = 'London'
       begin
       set @Country = 'England'
       end
       else
       begin
       set @Country = 'USA'
       end

       return @Country
end

select dbo.GetCountry ('London') as country
```



	country
1	England

```
select dbo.GetCountry ('Boston') as country
```



	country
1	USA