Marmara University – Faculty of Engineering – Department of Computer Engineering

Fall 2021 – CSE3055 Database Systems
Homework #4

Due: 19.01.2022.Wed 23:59

Consider the *Turkish Super League* database accompanied with this homework.

Player *(PlayerID: int, FirstName: nvarchar(25), LastName: nvarchar(25), Nationality: varchar(25), Birthdate: smalldatetime, Age: smallint, Position: varchar(25))*

Team *(TeamID: int, Name: nvarchar(50), City: nvarchar(25))*

PlayerTeam *(PlayerID: int, TeamID: int, Season: varchar(5))*

Match *(MatchID: int, HomeTeamID: int, VisitingTeamID: int, DateOfMatch: smalldatetime, Week: tinyint)*

Goals *(MatchID: int, PlayerID: int, IsOwnGoal: bit, Minute: tinyint)*

Notes:

- Table *Match* stores data only for season 2013-2014.
- Table *Goals* stores data only for season 2013-2014.

1) Table creation and data insertion.

a) Run the following queries to create the tables *Standings* and *TransactionLog* in your database.

```
Create Table Standings (
   Pos tinyint,
   [Team Name] nvarchar(30),
   GP tinyint,
   W tinyint,
   T tinyint,
   L tinyint,
   GF smallint,
   GA smallint,
   GD smallint,
   Pts tinyint
)
Create Table TransactionLog (
   LogID int identity(1,1) primary key,
   LogTime datetime,
   LogType char(1),
   BeforeState nvarchar(500),
   AfterState nvarchar(500),
)
```

b) In only one "insert into" statement; write a query to insert the output data of your stored procedure *sp_GetStandingsUpToDate('20140715')* that you have in homework #3 into table *Standings*.

2) Implement a trigger *Trg_RearrangeStandings* with the followings:

- When a record is inserted into, deleted from or updated on the table *Goals* (any change for *MatchID*, *PlayerID* and/or *IsOwnGoal*); then rearrange the table *Standings*, and insert a relevant record into the table *TransactionLog*.
- In all type of operations (insert, delete, update); *PlayerID* in table *Goals* must be a player of either the home team or the visiting team for that match in season 13-14. In any wrong match-team-player assignments, the transaction will be rolled back and any further executions will be stopped.
- A value less than 1 or greater than 90 cannot be entered in the field *Goals.Minute*.
- *TransactionLog.LogTime* is the time of operation.
- *TransactionLog.LogType* is "I" for insertion, "D" for deletion and "U" for update operation/transaction.
- *TransactionLog.BeforeState* is null for insertion and *TransactionLog.AfterState* is null for deletion. For update operation, *BeforeState* is the one before the operation and *AfterState* is the one after the operation.
- For the fields *BeforeState* and *AfterState* in table *TransactionLog*, concatenate all the related fields (*MatchID*, *PlayerID*, *IsOwnGoal*, *Minute*) in table *Goals* and separate them by a semicolon (e.g. '306;324;0;58') and enter this data in the fields *BeforeState* and *AfterState*, accordingly.

3) Consider the unnormalized relation R with six attributes ABCDEF and the following functional dependencies:

AB → CDE

B → F

C → D

a) What is the key(s) for the relation?

b) What is the normal form of this relation? Explain it.

c) Decompose R into 3NF relations step by step if it is not in 3NF.

4) Consider the following normalized relations from a database in a large retail chain:

STORE (StoreID, Region, ManagerID, SquareFeet)

EMPLOYEE (EmployeeID, WhereWork, EmployeeName, EmployeeAddress)

DEPARTMENT (DepartmentID, ManagerID, SalesGoal)

SCHEDULE (DepartmentID, EmployeeID, Date)

What opportunities might exist for denormalizing these relations when defining the physical records for this database? Under what circumstances would you consider creating such denormalized records?

5) Consider the following two relations for Millennium College:

STUDENT (StudentID, StudentName, CampusAddress, GPA)

REGISTRATION (StudentID, CourseID, Grade)

Following is a typical query against these relations:

```
SELECT Student.StudentID, StudentName, CourseID, Grade
FROM Student, Registration
WHERE Student.StudentID = Registration.StudentID AND GPA > 3.0
ORDER BY StudentName;
```

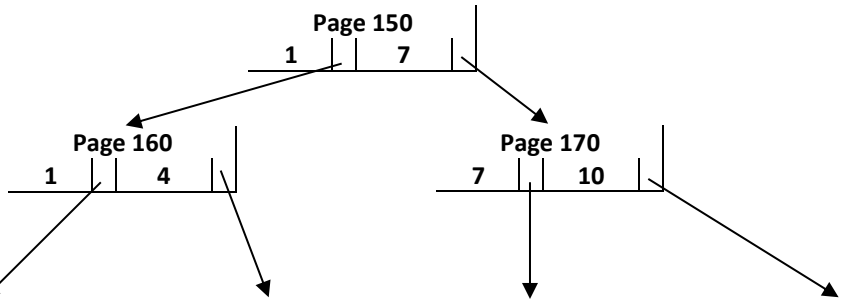
a) On what attributes should indexes be defined to speed up this query? Give the reasons for each attribute selected.

b) Write SQL commands to create indexes for each attribute you identified in part a.

6) You have a STUDENT table that has SID, Name, and Age columns. Which data pages are accessed to execute the queries below, under situations given at (a) and (b)? (Assume that index seek is used whenever possible)

STUDENT		
SID	**Name**	**Age**

a) The table has a *clustered index* on **SID** column, and no other indexes. The index structure and data is stored on data pages as the following:



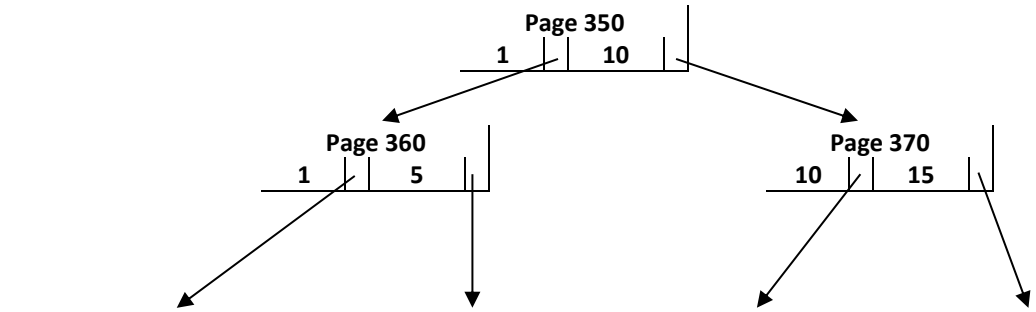
Page 110				Page 120				Page 130				Page 140			
01	1	Ayşe	2	**01**	4	Adem	4	**01**	7	Mert	8	**01**	10	Ceren	13
02	2	Fatma	7	**02**	5	Mehtap	4	**02**	8	Mehmet	10	**02**	11	Ali	16
03	3	Can	11	**03**	6	Ahmet	6	**03**	9	Zeynep	17	**03**	12	Yavuz	16

i) **Query 1:** select Name from STUDENT where SID < 11

ii) **Query 2:** select * from STUDENT where Age = 16

iii) **Query 3:** select * from STUDENT where SID = 7

b) The table has a *non-clustered index* on **Age** column, and no other indexes. The index structure and data is stored on data pages as the following:



Page 220		Page 230		Page 240		Page 250	
2	100:01	**6**	110:03	**10**	120:02	**16**	130:02
4	110:01	**7**	100:02	**11**	100:03	**16**	130:03
4	110:02	**8**	120:01	**13**	130:01	**17**	120:03

Page 100				Page 110				Page 120				Page 130			
01	1	Ayşe	2	**01**	4	Adem	4	**01**	7	Mert	8	**01**	10	Ceren	13
02	2	Fatma	7	**02**	5	Mehtap	4	**02**	8	Mehmet	10	**02**	11	Ali	16
03	3	Can	11	**03**	6	Ahmet	6	**03**	9	Zeynep	17	**03**	12	Yavuz	16

i) **Query 1:** select Age, Name from STUDENT where SID < 9

ii) **Query 2:** select Age from STUDENT where Age < 8

iii) **Query 3:** select * from STUDENT where Age = 8

IMPORTANT NOTES

1) Write the following sentence in a text file: "We hereby swear that the work done on this project is totally our own; and on our honor, we have neither given nor received any unauthorized and/or inappropriate assistance for this homework. We understand that by the school code, violation of these principles will lead to a zero grade and is subject to harsh discipline issues." Rename it as "we_swear.txt" and include this file in the zip submission file.

2) In case of any form of copying and cheating on solutions, all parts will get ZERO points. You should submit your own work. In case of any forms of cheating or copying, both giver and receiver are equally culpable and suffer equal penalties. All types of plagiarism will result in zero points from the homework.

3) Compress the following files in a zip file and name it as hw4_studentID.zip

- **a)** hw4_studentID_we_swear.txt: honor code.
- **b)** hw4_studentID_sql.txt: all your sql statements merged in a single text file in order.
- **c)** hw4_studentID_scr.pdf: all screenshots of both your SQL queries and output of the queries in order.

4) Submit your single zip file to the site http://ues.marmara.edu.tr before deadline.

5) Do not send homework submissions through e-mail. E-mail attachments will not be accepted as valid submissions.

6) You are responsible for making sure you are turning in the right file, and that it is not corrupted in anyway. We will not allow resubmissions if you turn in the wrong file, even if you can prove that you have not modified the file after the deadline.

7) No groups are allowed.

8) Grade evaluation may be done on selected parts of the homework, so try to complete all parts of your homework successfully.

9) No late submissions will be accepted.