CSE 462 Homework #3 (Optional): Relational Algebra and SQL

Name: _____ Date: March 05, 2011

***** Due on March 25, 2011 at the beginning of class. *****

Instructions. This problem set is optional and worth 200 points. Your answers **must be typed**. All problems reference the NBA player statistics schema below:

`Team`(`tId` : `int`, `name` : `string`, `homeCity` : `string`)

`Game`(`gId` : `int`, `homeId` : `int`, `awayId` : `int`, `isPlayoff` : `boolean`, `year` : `int`)

`Player`(`pId` : `int`, `lastName` : `string`, `firstName` : `string`, `height` : `int`, `tId` : `int`)

`Stats`(`gId` : `int`, `pId` : `int`, `points` : `int`, `rebounds` : `int`, `assists` : `int`, `seconds` : `int`)

Observations:

- Keys are underlined.
- No attributes allow `NULL`s.
- Table `Team` stores all NBA teams.
- Table `Player` stores every player of every team.
- Table `Game` stores every regular season and playoff game.
- Table `Stats` records individual player statistics for every season and playoff game.
- Attribute `homeId` in `Game` has a foreign key referencing `tId` in `Team`.
- Attribute `awayId` in `Game` has a foreign key referencing `tId` in `Team`.
- Attribute `tId` in `Player` has a foreign key to `tId` in `Team`.
- Attribute `gId` in `Stats` has a foreign key to `gId` in `Game`.
- Attribute `pId` in `Stats` has a foreign key to `pId` in `Player`.
- A player achieves a *double-double* in a game when he obtains a number total of 10 or more in exactly two out of the three statistics (points, rebounds, assists). He achieves a *triple-double* when he obtains a number total of 10 or more in all three statistics.
- **Important:** if a player does not play a game, there is no entry for *that* player in the `Stats` table for *that* game.

Problem 1. [80 points] Write relational algebra expressions to answer the problems below.

a) [10 points] Find the tallest player(s) of each team. List the respective `tId` and `pId` pairs.

b) [20 points] Find player(s) who, for some year, did not play all regular season games neither all away games that his team played. List their respective `pId`s.

c) [20 points] For every year, find the player(s) who scored more points than all other players in his team in every playoff game that he played. List the respective `year` and `pId` pairs.

d) [30 points] Find the player(s) who achieved at least two triple-doubles in playoff games in at least two consecutive years. List the respective `pId`, `lastName`, and `firstName`.

Problem 2. [120 points] Create SQL views to answer the problems below.

a) [10 points] View `RegularSeasonAverages(pId, avgPoints)` that computes, for every player of every team, the average points scored by that player for the games he played in the regular season. If a player played no regular season game, his average should appear as zero.

b) [10 points] Using the view in (a), create a view `MinMaxDelta(minAvg, maxAvg, delta)` that returns a single tuple, where `minAvg` and `maxAvg` are the minimum and maximum average points computed by the view in (a), and `delta = maxAvg - minAvg` is the difference between the largest and smallest average points.

c) [10 points] You are given the task of "grading" all players according to their average scored points during the regular season. Using the views in (a) and (b), create a view `RegularSeasonGrades(pId, grade)` that classifies each player, based on his average score `s`, as follows:

- if ($s \geq avgMin + 4 * delta$) then his grade is "A"
- if ($avgMin + 3 * delta \leq s < avgMin + 4 * delta$) then his grade is "B"
- if ($avgMin + 2 * delta \leq s < avgMin + 3 * delta$) then his grade is "C"
- if ($avgMin + 1 * delta \leq s < avgMin + 2 * delta$) then his grade is "D"
- if ($s < avgMin + 1 * delta$) then his grade is "F"

d) [20 points] Using the view in (c), create a view `Frequencies(grade, absFreq, relFreq)` that computes the absolute and relative frequencies of the grades of each player. The absolute frequency is the number of players having a particular grade and the relative frequency is the absolute frequency divided by the total number of players. Note that the sum of absolute frequencies must be equal to the total number of players and the sum of all relative frequencies must be equal to ONE.

e) [20 points] View `GameScores(gId, homePoints, awayPoints)` that returns the final score of every game. *Hint:* create helper views `HomeScores(gId, homePoints)` and `AwayScores(gId, AwayPoints)` to compute the total number of points scored by the home and away teams each game.

f) [10 points] Using the view in (e), create a view `RepeatedScores(gId1, gId2)` that returns pairs of distinct `gId`s such that the respective games occurred in different years, had the same home and away teams, and also the same final score. If your query returns the pair (A,B), it must not return the pair (B,A).

g) [20 points] View `AwayYear(year)` that returns the years in which away teams won more often than home teams. *Hint:* create a helper view `AwayWins(year, wins)` that computes, for each year, the total number of games won by away teams.

h) [20 points] View `ImprovingTeams(tId)` that returns the teams that, for each year, won more games in that year than in any preceding year. The first year that a team played satisfies the condition only if the team won at least one game in that year.