**CS3431 A19 Wong**

**Assignment 3: Relational Algebra and More SQL**

Due Date: T 9/10 at 11:59pm

Late Policy: **No late submissions** because solutions will be posted immediately to allow you to study for exam 1. Maximum grade is 100 points. Submission: In PDF or Word format using the Assignment 3 Submission button. Be aware that Canvas will be applying late penalties automatically! As a result, the Canvas deadline is set for Wednesday at 1am to give you a little extra time if you have problems with your submission but technically the actual deadline is T 9/10 at 11:59pm.

This assignment is to be typed. The following relational algebra symbols are provided for your use in the assignment:

σ, π, ρ, γ, **δ,** ⋈, 🡨, ∩, **∪**

**Part 1: Science Fiction and Nonfiction Books**

Use the abbreviations in place of the full table names in your relational algebra or 5 points off. Use the full table names in your SQL code. Primary keys are underlined. ***Foreign keys*** are in bold and italicized.

For problem 1, use the relations given below for tennis players and their associations and tournaments:

P 🡨 Players (fullName, birthDate, handedness, currentRank, totalEarnings, ***association***)

A 🡨 Association (associationName, location, foundingYear)

T 🡨 Tournament (tournamentName, startingDate, endingDate, prizeMoney)

PT 🡨 PlayersInTournament (***tournamentName***, ***startingDate***, ***playerFullName, playerBirthDate***)

1. (10 points) Based on the given primary keys, specify below the foreign key relationships that exist between the tables that would make sense.
   1. Using the table abbreviations, write the constraints using the following format:   
      Foreign Key (TableA.ID1, TableA.ID2) References (TableB.ID1, TableB.ID2)

Foreign Key (P.association) References (A.associationName)

Foreign Key (PT.tournamentName) References (T.tournamentName)

Foreign Key (PT.startingDate) References (T.startingDate)

Foreign Key (PT.playerFullName, PT.playerBirthDate) References (P.fullName, P.birthDate)

* 1. Using the full table names, write named SQL constraints for the foreign keys. If a tournament is canceled then the foreign key in all records that refer to the tournament should be changed to null. Assume the tables with the field names already exist but without any foreign key constraints. Use ALTER commands to create the foreign key constraints.

ALTER table Players

ADD Constraint P\_A\_FK Foreign Key (association) References Association (associationName);

ALTER table PlayersInTournament

ADD Constraint PT\_T\_FK Foreign Key (tournamentName) References Tournament (tournamentName)

On Delete Null;

ALTER table PlayersInTournament

ADD Constraint PT\_T\_FK2 Foreign Key (startingDate) References Tournament (startingDate)

On Delete Null;

ALTER table PlayersInTournament

ADD Constraint PT\_P\_FK Foreign Key (playerFullName, playerBirthDate) References Players (fullName, birthDate);

For problems 2 to 4, use the relations given below:

**A 🡨 Authors** (fullname, birthDate, website, city)

-- science fiction books

**SF 🡨 SFBooks** (ISBN, title, copyrightYear, price, awardWinner, ***publisher***, ***fullname, birthDate***)

-- non-fiction books

**NF 🡨 NFBooks** (ISBN, title, copyrightYear, price, awardWinner, ***publisher***, ***fullname, birthDate***)

**P 🡨 Publishers** (publisherName, city, phone)

**W 🡨 Warehouses** (code, address, city)

**S 🡨 Stocks** (warehouseCode, ISBN, numberOfBooks)

Write efficient relational algebra and SQL code by projecting only those fields that are needed and applying selection conditions as early as possible (often results in subqueries). Note: We do not have to worry about a science fiction book having the same ISBN number as a non fiction book because the US Library of Congress assigns unique numbers to books.

1. (20 points) List the title, copyright year, and number of copies for science fiction books that have a total of more than 100 copies throughout the warehouses. Use theta joins.
2. Write the relational algebra

π title, copyrightYear, numberOfBooks (π title, copyrightYear, ISBN (SF) ⋈SF.ISBN = S.ISBN π ISBN, numberOfBooks σnumberOfBooks > 100 (S))

1. Write the SQL code

Select title, copyrightYear, numberOfBooks

From (select title, copyrightYear, ISBN from SFBooks)

JOIN (select ISBN, numberOfBooks from Stocks

Where numberOfBooks > 100)

On SF.ISBN = S.ISBN;

1. (20 points) For each author, list the author name, city, and the total number of books (science fiction and non-fiction) published before the year 2015. Use natural joins instead of theta joins.
   1. Write the relational algebra

π fullName, city, γfullName, count(SF.ISBN + NF.ISBN) σnumberOfBooks > 100 (π fullName, city, ISBN σcopyrightYear < 2015 (SF) ⋈ π fullName, city, ISBN σcopyrightYear < 2015 (NF))

* 1. Write the SQL code for the above, but sorted by city and then by author name

Select fullName, city, count(SF.ISBN + NF.ISBN)

From (select fullName, city, ISBN from SFBooks

Where copyrightYear < 2015) SF

NATURAL JOIN (select fullName, city, ISBN from NFBooks

Where copyrightYear < 2015) NF

GROUP BY (city, fullName);

1. (20 points) For authors who published non-fiction books but not science fiction books, list the author’s fullname, birthDate and the price of the most expensive book they wrote as MaxPrice. For this question, use natural joins. Do not use the null value in your code, use set theory instead.
   1. Write the relational algebra

π fullName, birthDate, ρMaxPrice (γmax(price)) [(π fullName, birthDate, price NF ⋈ π fullName, birthDate, price SF) – (π fullName, birthDate, price SF)]

* 1. Write the SQL code for the above using subqueries in the from/join statement, but sorted in reverse order by maximum price

Select fullName, birthDate, MAX(price) as MaxPrice

From (select fullName, birthDate, price from NFBooks)

NATURAL JOIN (select fullName, birthDate, price from SFBooks)

Where fullName NOT IN (select fullName, birthDate, price from SFBooks)

Group By fullName, birthDate

ORDER BY price;

**Part 2: Relational Algebra [30 Points (5 Points each subpart)]**

Given the relations M and N:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **M** | | | |  | **N** | | |
| **A** | **B** | **C** | **D** |  | **X** | **Y** | **B** |
| 14 | 4 | Henry | David |  | 3 | Matt | 5 |
| 8 | 2 | Mary | Helen |  | 2 | Tom | 8 |
| 2 | 6 | Jane | Susan |  | 8 | Mary | 61 |
| 61 | 3 | Tom | Paul |  | 8 | Lisa | 2 |
| 2 | 5 | Lisa | Mary |  | 5 | John | 8 |
| 24 | 9 | Mark | John |  | 13 | Tom | 42 |
| 8 | 1 | Paul | Amy |  | 42 | Henry | 2 |
| 5 | 2 | Dan | Matt |  |  |  |  |

For each relational algebra statement

1. calculate the following output tables and be careful in labeling the attributes
2. write the corresponding efficient (be careful!) SQL code and using subqueries as appropriate.
3. π A+B as E, C, X as G σ X>B (M ⋈N)

a.

|  |  |  |
| --- | --- | --- |
| E | C | G |
| 7 | Dan | 8 |
| 10 | Mary | 8 |
| 10 | Mary | 42 |
| 7 | Dan | 42 |

b. select A + B as E, C, X as G

from (select A, B, C from M) NATURAL JOIN (select X, B from N where X > B);

1. **γ**C, D, sum(A+M.B+X) as Total (M ⋈M.B=N.X N)

a.

|  |  |  |
| --- | --- | --- |
| C | D | Total |
| Mary | Helen | 12 |
| Tom | Paul | 67 |
| Lisa | Mary | 12 |
| Dan | Matt | 9 |

b. select C, D, SUM(A + M.B + X) as Total

from (select B, C, D from M) JOIN (select B, X from N)

on M.B = N.X

group by C, D;

1. πR.X,R.Y,N.B (N ⋈ N.X = R.X and N.Y = R.Y

ρR (σX>5 (N) -πB as X,D as Y,A as B σA>B (M)))

a.

|  |  |  |
| --- | --- | --- |
| X | Y | B |
| 8 | Mary | 61 |
| 8 | Lisa | 2 |
| 13 | Tom | 42 |
| 42 | Henry | 2 |

b. select R.X, R.Y, N.B

from N JOIN (select \* from N

where X > 5 NOT IN

(select B AS X, D AS Y, A AS B

from M

where A > B)) R

on N.X = R.X AND N.Y = R.Y;