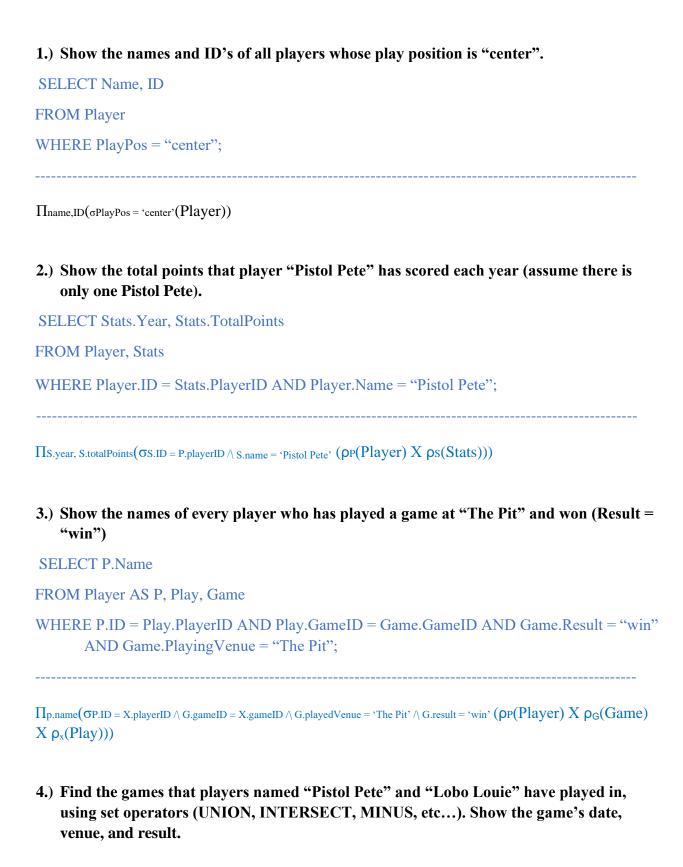
## **Assignment-2**

## CS482/502 Database Management Systems I Assignment: Relational Algebra + SQL

Spring 2020

Assume that you are given the following relational schemas for the basketball team at NMSU.

- Player (ID: integer, Name: varchar(64), Birthday: date, Address: varchar(128), Email: varchar(32), PhoneNumber: char(10), PlayPos: varchar(16))
- Manager (ID: integer, LoginID: varchar(16), Name: varchar(64), Password: varchar(8), Birthday: date, Address: varchar(128), Email: varchar(32), PhoneNumber: char(10))
- ManagerCertificate (ManagerID: integer, CertificateId: integer, Certificate: blob) Foreign key: ManagerID references Manager(ID)
- Doctor (Email: varchar(32), Name: varchar(64), PhoneNumber: char(10))
- TakeExam (PlayerID: integer, DocEmail: varchar(32), TestDate: date, TestResult: varchar(256))
  - Foreign key: PlayerID references Player(ID)
  - Foreign key: DocEmail references Doctor(Email)
- Stats (PlayerID: integer, Year: char(4), TotalPoints: integer, ASPG: integer) Foreign key: PlayerID references Player(ID)
- Training (TrainingName: varchar(256), Instruction: varchar(256), TimePeriodInHour: integer)
- AssignTraining (PlayerID: integer, ManagerID: integer, TrainingName: varchar(256))
  - Foreign key: PlayerID references Player(ID)
  - Foreign key: ManagerID references Manager(ID)
  - Foreign key: TrainingName references Training(TrainingName)
- Game (GameID: integer, Date: date, Result: varchar(16), PlayingVenue: varchar(256), OpponentTeam: varchar(32))
- Play (PlayerID: integer, GameID: integer)
  - Foreign key: PlayerID references Player(ID)
  - Foreign key: GameID references Game(GameID)



```
(SELECT G.Date, G.Playing Venue, G.Result FROM Player as P, Game as G, Plays WHERE P.ID = Plays.PlayerID AND Plays.GameID = G.GameID AND P.name = "Pistol Pete") INTERSECT (SELECT G.Date, G.Playing Venue, G.Result FROM Player as P, Game as G, Plays WHERE P.ID = Plays.PlayerID AND Plays.GameID = G.GameID AND P.name = "Lobo Louie");  \frac{1}{1} \frac{1}{1}
```

5.) Find the Names and IDs of players who have scored more points than the average player.

```
SELECT P.Name, P.ID
```

FROM Stats S, Player P

WHERE P.ID = S.PlayerID and S.TotalPoints > (SELECT AVG(TotalPoints) FROM Stats);

\_\_\_\_\_\_

Temp  $\leftarrow \mathcal{G}$  AVG(TotalPoints)(Stats)

 $\Pi_{P.name, P.ID}$  ( $\sigma_{P.ID} = S.playerID \land S.totalPoints > Temp(<math>\rho_{P}(Player) \times \rho_{S}(Stats)$ ))

Assume that you are given the following relational schemas.

- members (memb\_no int(3), name varchar(64))
- books (isbn int(6), title varchar(64), authors varchar(128), publisher varchar(128))
- borrowed (memb\_no int(3), isbn int(6))

Write an SQL Query for each of the following.

1.) Show the names of members who borrowed books with title "Math".

SELECT m.name

FROM members m, books b, borrowed br

WHERE m.memb\_no = br.memb\_no and br.isbn = b.isbn and b.title = "Math";

2.) Show the details of members whose name does not start with 'J'.

**SELECT** \*

FROM members

WHERE name not like "j%";

3.) Find the number of books borrowed by each member and show them in descending order.

SELECT m.memb\_no, count(m.memb\_no)

FROM members m, borrowed br

WHERE m.memb\_no = br.memb\_no

GROUP BY m.memb\_no

ORDER BY count(m.memb\_no) DESC;

4.) Show the details of members whose name contains 'A'.

**SELECT** \*

FROM members

WHERE name like "%a%";

5.) Find the distinct publisher name of the book which has been borrowed by "Sam".

SELECT DISTINCT(b.publisher)

FROM members m, books b, borrowed br

WHERE m.memb\_no = br.memb\_no and br.isbn = b.isbn and m.name = "Sam";