Zachary Neeley
3/10/2020

1.) Show the names and ID's of all players whose play position is "center".

Π P.name, P.ID ($\sigma_{P.PlayPos = \text{'center'}}$ (P_P(Player)))

2.) Show the total points that player "Pistol Pete" has scored each year (assume there is only one Pistol Pete).

S.Year^G (Π S.Year, S.TotalPoints ($\sigma_{S.PlayerID = P.ID \wedge P.name = \text{'Pistol Pete'}}$ (P_S(Stats) X P_P(Player))))

3.) Show the names of every player who has played a game at "The Pit" and won (Result = "win")

PL.Birthday^G desc(Π Pl.name ($\sigma_{PL.ID = P.PlayerID \wedge G.GameID = P.GameID \wedge G.PlayingVenue = \text{'The Pit'} \wedge G.result = \text{'win'}}$ (P_{PL}(Player) X P_P(Play) X P_G(Game))))

4.) Find the games that players named "Pistol Pete" and "Lobo Louie" have played in, using set operators (UNION, INTERSECT, MINUS, etc…). Show the game's date, venue, and result.

g.GameID^G (Π g.GameID, g.Date, g.PlayingVenue, g.Result($\sigma_{plr.ID = p.PlayerID \wedge g.GameID = p.GameID \wedge plr.Name = \text{'Lobo Louie'}}$ (P_g(Game) X P_{plr}(Player) X P_p(Play))))

$\cap$

Π g.GameID, g.Date, g.PlayingVenue, g.Result($\sigma_{plr.ID = p.PlayerID \wedge g.GameID = p.GameID \wedge plr.Name = \text{'Pistol Pete'}}$ (P_g(Game) X P_{plr}(Player) X P_p(Play)))

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

Π P.name, P.ID ($\sigma_{P.ID = S.PlayerID \wedge S.TotalPoints > S.ASPG}$ (P_P(Player) X P_S(Stats)))

Zachary Neeley
3/10/2020

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 distinct(m.name)
FROM Members M, Books B, Borrowed Bow
WHERE M.Memb_no = Bow.Memb_no
AND B.isbn = Bow.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 distinct(Bow.memb_no), count(*)
FROM Borrowed Bow
GROUP BY Bow.Memb_no
ORDER BY Bow.Memb_no desc;
```

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

```
SELECT *
FROM Members Mem
WHERE Mem.Name LIKE '%A%';
```

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

```
SELECT distinct(B.Publisher)
FROM Members Mem, Books B, Borrowed Bow
```

Zachary Neeley
3/10/2020

```
WHERE Mem.Memb_no = Bow.Memb_no
AND B.Isbn = Bow.Isbn
AND Mem.Name = "Sam";
```