

### ERD Paragraph:

The Team Table has attributes Team\_Name, Team\_ID, Team\_Abbreviation, Manager\_ID, and Year\_Founded, where **Team\_ID** is the **primary key**, and **Manager\_ID** is a **foreign key** pointing to the **Manager Table**. **Team\_ID** can be found as a **foreign key** in the **Player Table**.

The Manager Table has attributes First\_Name, Last\_Name, Age, and Manager\_ID, where **Manager\_ID** is a **primary key**, and can be found as a **foreign key** in the **Team Table**.

The Player Table has attributes First\_Name, Last\_Name, Age, Player\_ID, Team\_ID, and Shirt\_Number, where **Player\_ID** is a primary key, and **Team\_ID** is a foreign key, pointing to the **Team Table**.

### Table Normalisation

These tables are all in BCNF, as all non-key attributes are functionally dependent on the primary keys of each table, and there are no transitive dependencies, and no non-prime attributes are dependent on any part of the table except the candidate keys.

### DDL Statements

```
CREATE TABLE Team (  
    Team_ID INTEGER,  
    Team_Name VARCHAR(100),  
    Team_Abbreviation VARCHAR(5),  
    Year_Founded INTEGER,  
    Manager_ID INTEGER NOT NULL,  
    PRIMARY KEY (Team_ID),  
    FOREIGN KEY (Manager_ID) REFERENCES Manager(Manager_ID)  
);
```

```
CREATE TABLE Manager (  
    Manager_ID INTEGER  
    First_Name VARCHAR(50),  
    Last_Name VARCHAR(50),  
    Age INTEGER,  
    PRIMARY KEY (Manager_ID) ON DELETE CASCADE  
);
```

```

CREATE TABLE Player (
    Player_ID INTEGER,
    First_Name VARCHAR(50),
    Last_Name VARCHAR(50),
    Age INTEGER,
    Team_ID INTEGER,
    Shirt_Number INTEGER,
    PRIMARY KEY (Player_ID)
    FOREIGN KEY (Team_ID) REFERENCES Team(Team_ID)
);

```

### Relational Algebra Statements

1. Finding the average age in each team:

```

SELECT Team.Team_ID, AVG(Player.Age) AS Avg_Age, Team.Team_Name
FROM Player, Team
WHERE Player.Team_ID = Team.Team_ID
GROUP BY Player.Team_ID, Team.Team_Name;

```

2. Finding the manager for each team, sorting by age:

```

SELECT Manager.Age, Team.Manager_ID, Manager.First_Name, Manager.Last_Name
FROM Manager, Team
WHERE Team.Manager_ID = Manager.Manager_ID
GROUP BY Manager.Age, Team.Manager_ID
ORDER BY Manager.Age;

```

3. Deleting Arsenal FC

```

DELETE *
FROM Team
WHERE Team_ID = 1;

```

4. Deleting Arsenal's Manager:

```

DELETE *
FROM Manager
WHERE Manager_ID = 1;

```

## DML Statements

```
INSERT INTO Team(Team_ID, Team_Name, Team_Abbreviation, Year_Founded, Manager_ID)
VALUES (?, ?, ?, ?, ?);
```

```
INSERT INTO Manager(Manager_ID, First_Name, Last_Name, Age)
VALUES (?, ?, ?, ?);
```

```
INSERT INTO Player(Player_ID, First_Name, Last_Name, Age, Team_ID, Shirt_Number)
VALUES (?, ?, ?, ?, ?, ?);
```

?’s will be replaced with data in the CSV when it is all extracted, via a preparedStatement object.