DATABASE PROJECT REPORT

BRANDON DOOLEY

#16327446

TRINITY COLLEGE DUBLIN

**Introduction**

As part of the CS3041 Information Management database design project I decided to model a database representing the top 10 teams in the Barclays Premier League as of Thursday the 1st November 2018.

The relational tables that I chose to model are as follows:

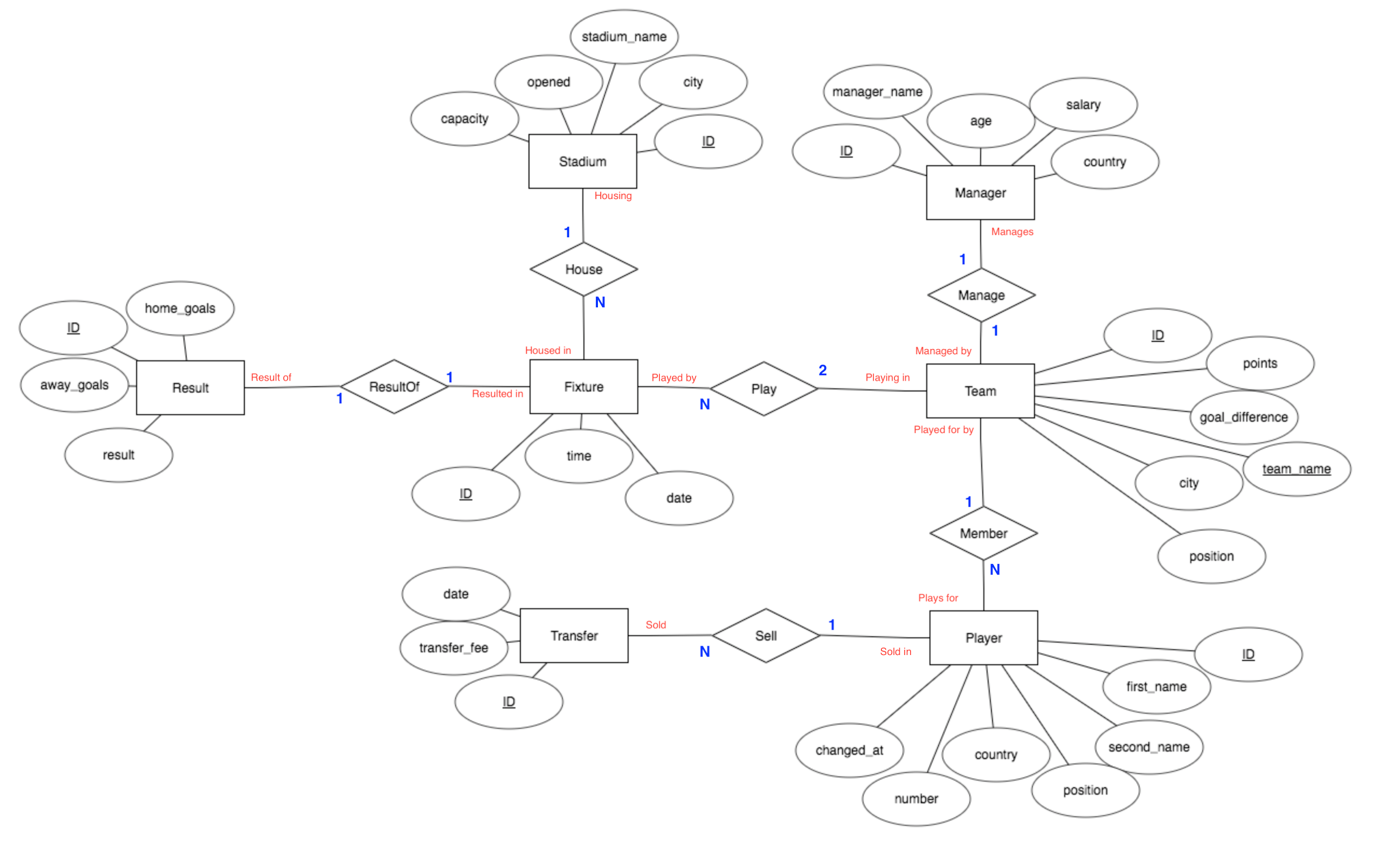
* Fixtures
* Managers
* Players
* Stadiums
* Teams
* Transfers

Within the database I modelled all fixtures from Sunday 11th up to Saturday the 8th of December. This was done to allow for me to continuously input the results of these fixtures into the database in order to demonstrate the use of my designed triggers and the effect they have on other relations within the database.

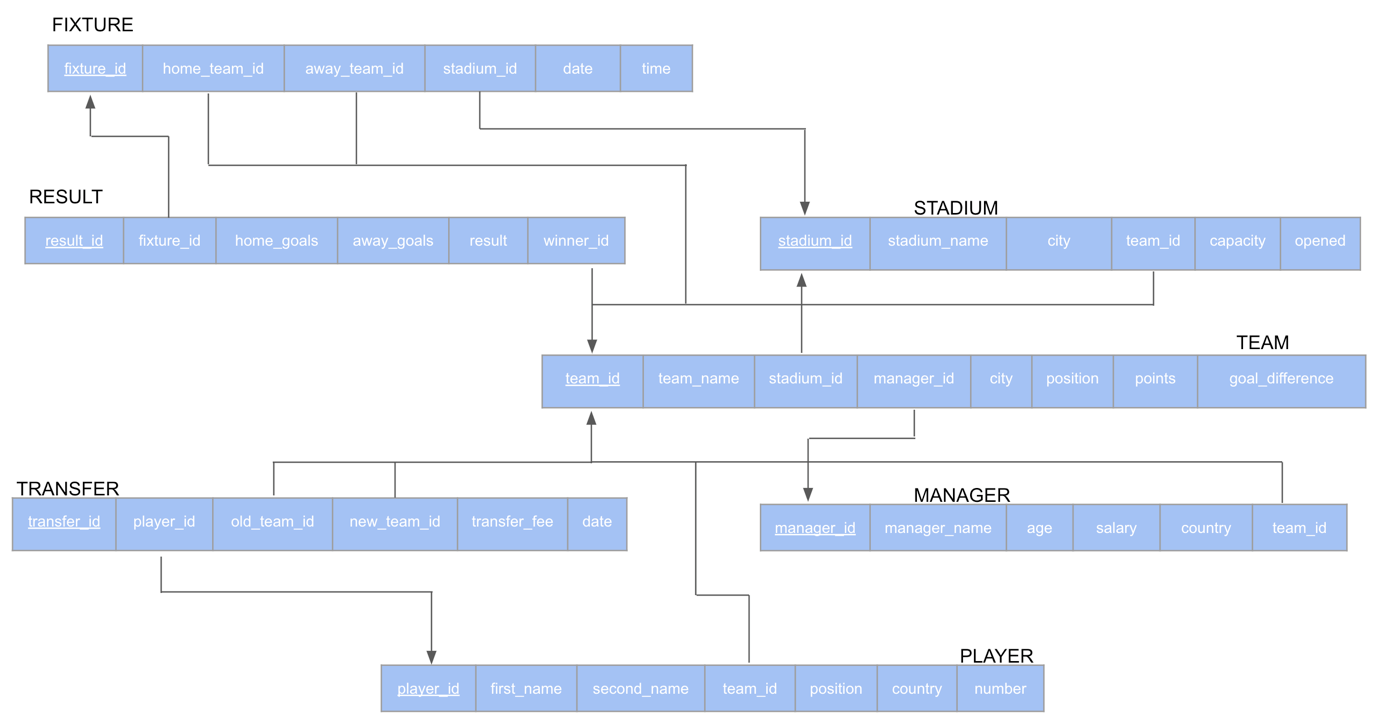
Within the players table I modelled five players from each of the teams including a goalkeeper, defender, midfielder and two forwards for each team. The data regarding stadiums and managers is valid as of Sunday 11th of November.

All data regarding transfers is fictional and does not represent any real transfer that has occurred in the premier league within this period. They also solely serve the purpose of demonstrating the use of the designed triggers and their respective effects.

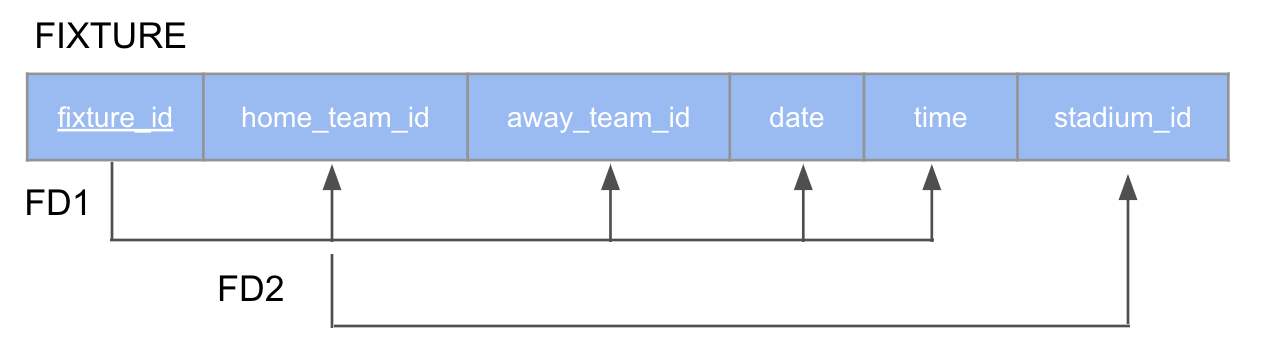
**Entity Relationship Diagram**



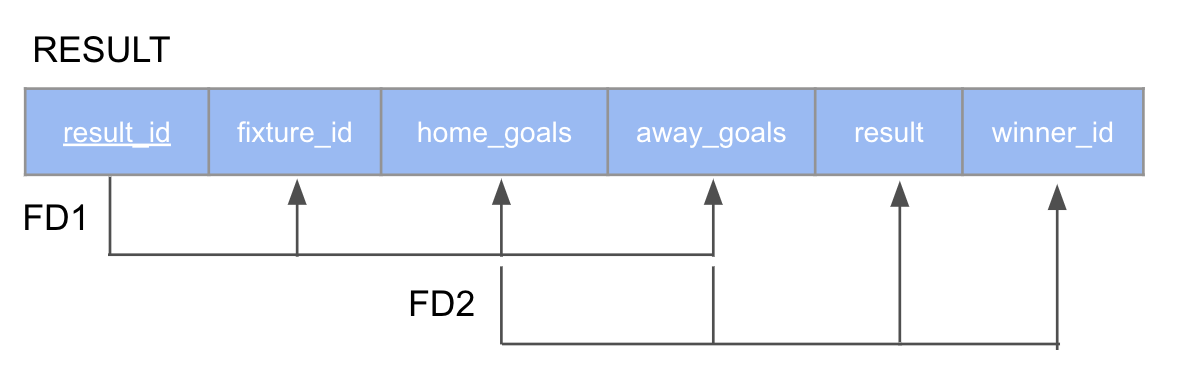
**Relational Schema Diagram**



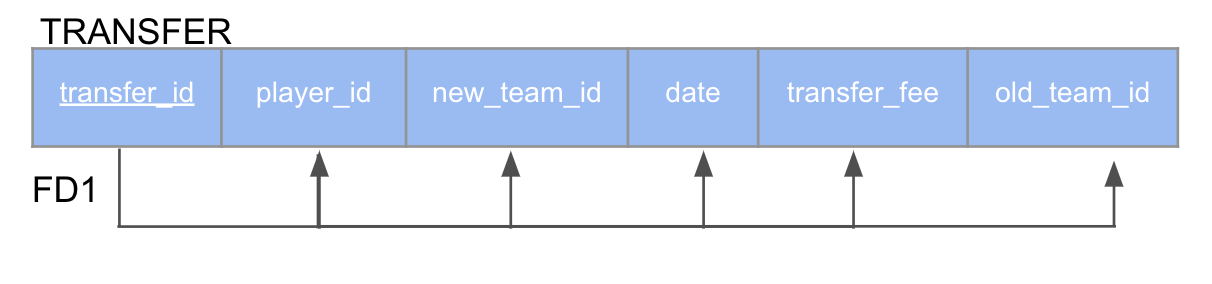
**Functional Dependency Diagrams**



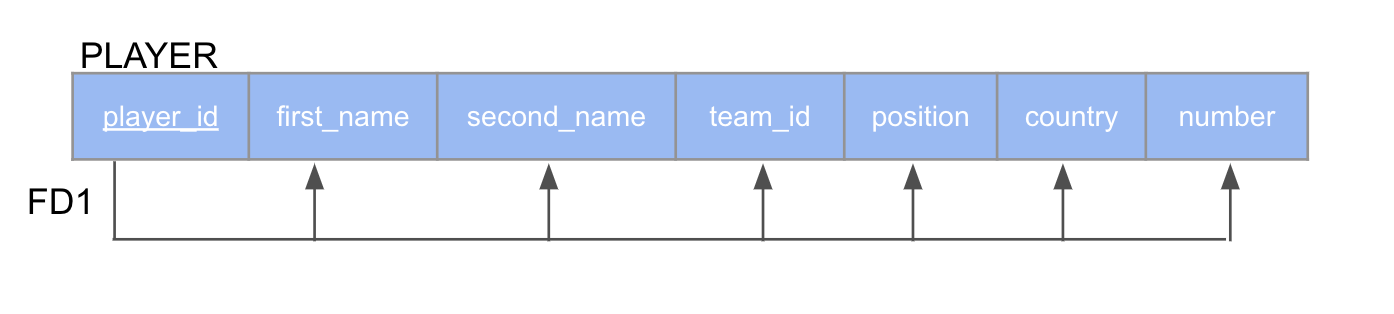
* Primary Key: fixture\_id
* Foreign Keys: {home\_team\_id, away\_team\_id, stadium\_id}



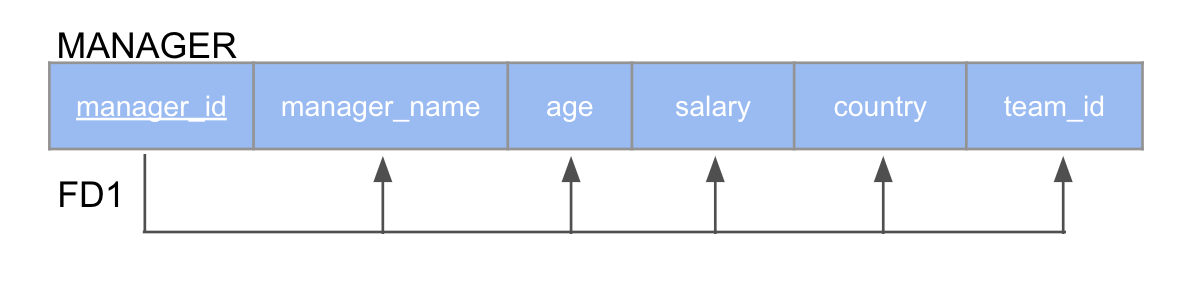
* Primary Key: result\_id
* Foreign Keys: {fixture\_id, winner\_id}



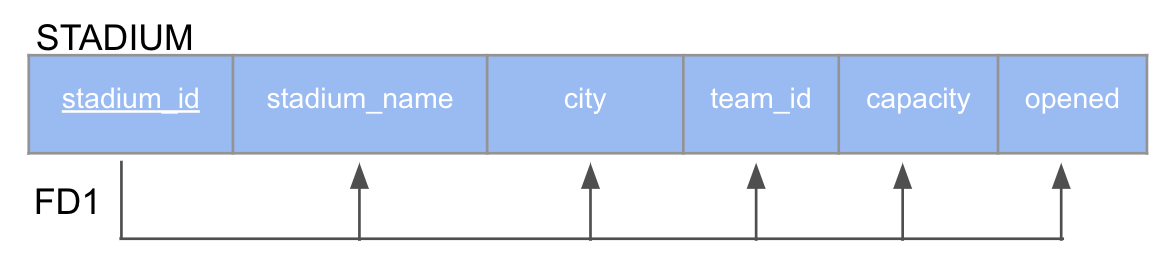
* Primary Key: transfer\_id
* Foreign Keys: {player\_id, new\_team\_id, old\_team\_id}



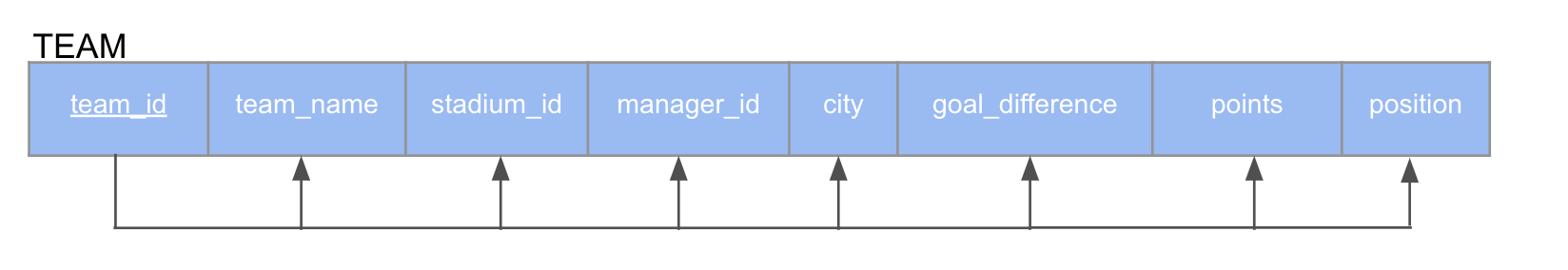
* Primary Key: player\_id
* Foreign Keys: {team\_id}



* Primary Key: manager\_id
* Foreign Keys: {team\_id}

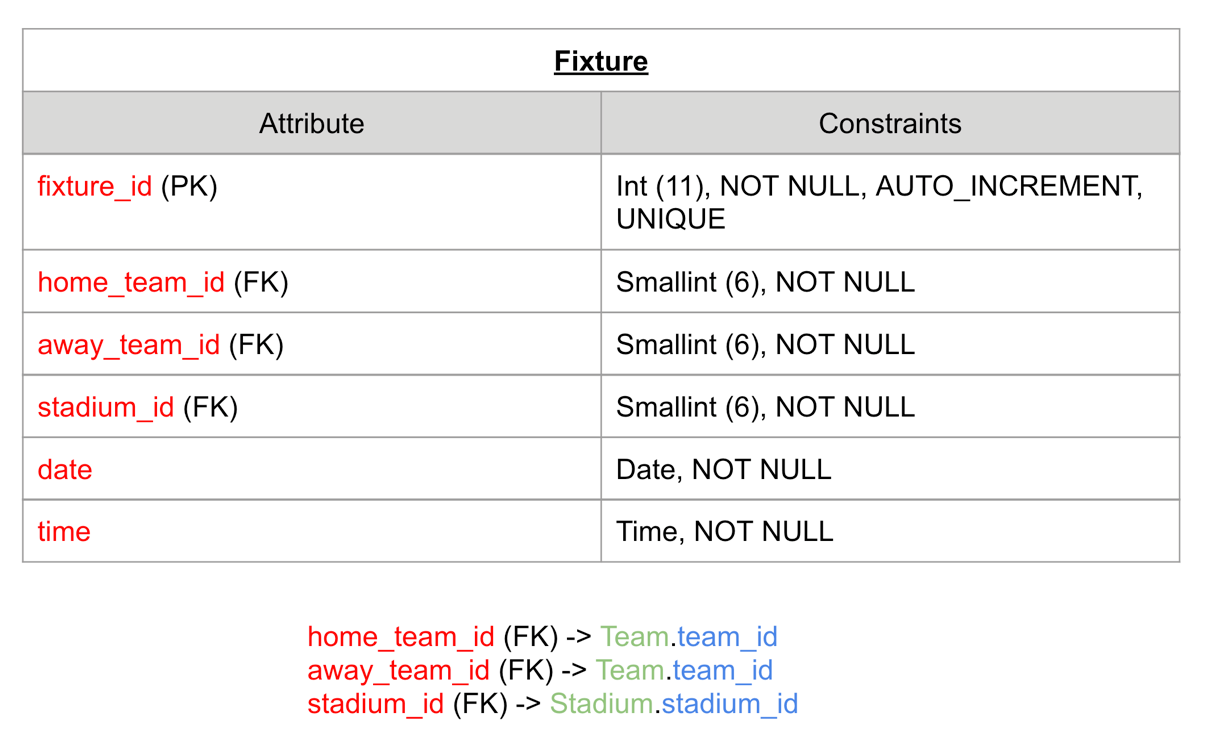


* Primary Key: stadium\_id
* Foreign Keys: {team\_id}

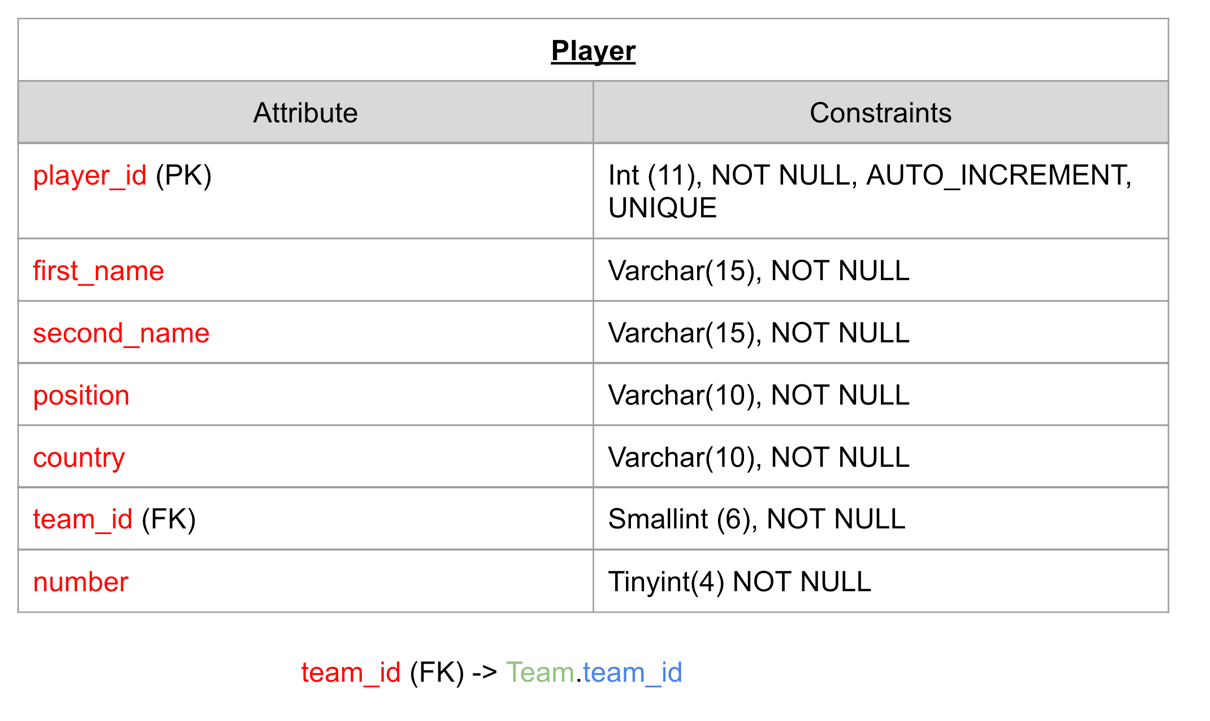


* Primary Key: team\_id
* Foreign Keys: {stadium\_id, manager\_id}

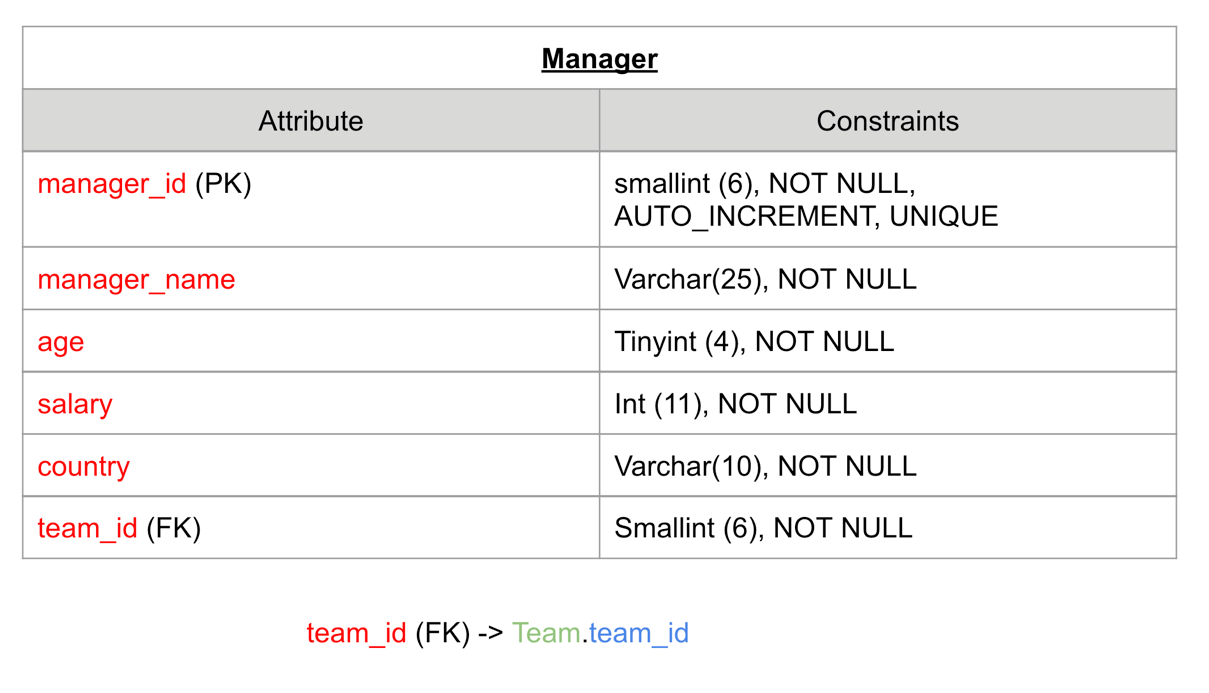
**Semantic Constraints**



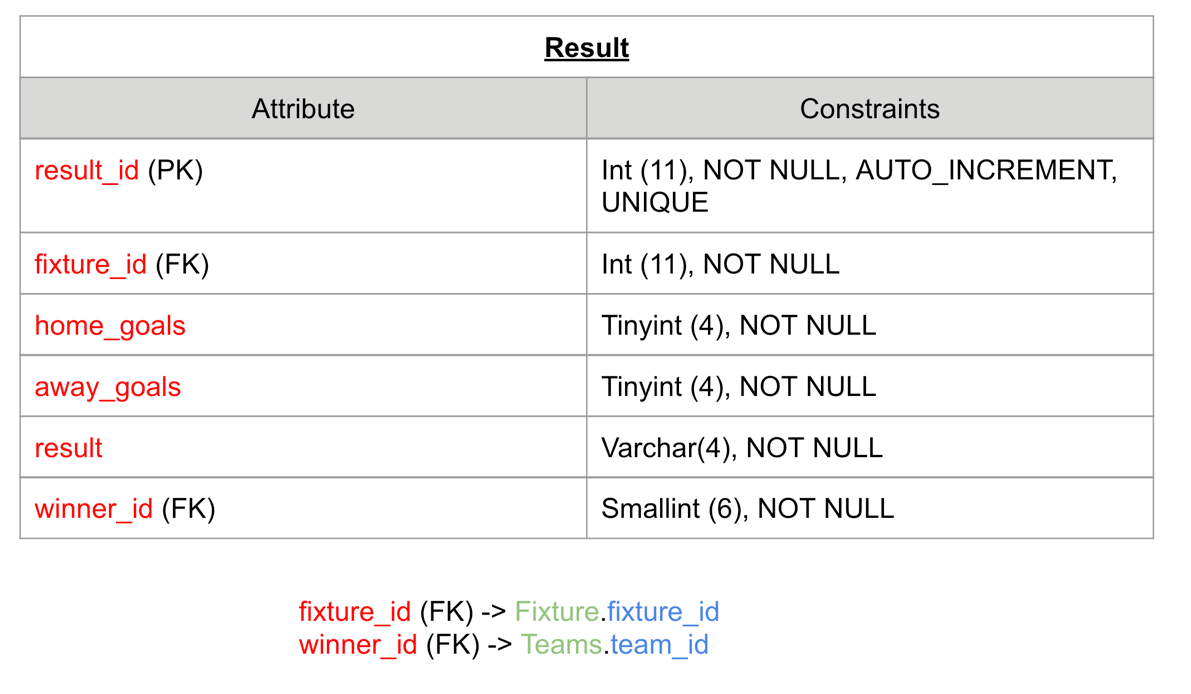
For *fixture\_id* I chose to use int() as the storage type since there could be over 32,767 entries of fixtures which is the maximum provided by smallint(). I chose to use the Date and Time data types to store the date and time of a fixture as they would ensure the validity of their formatting.



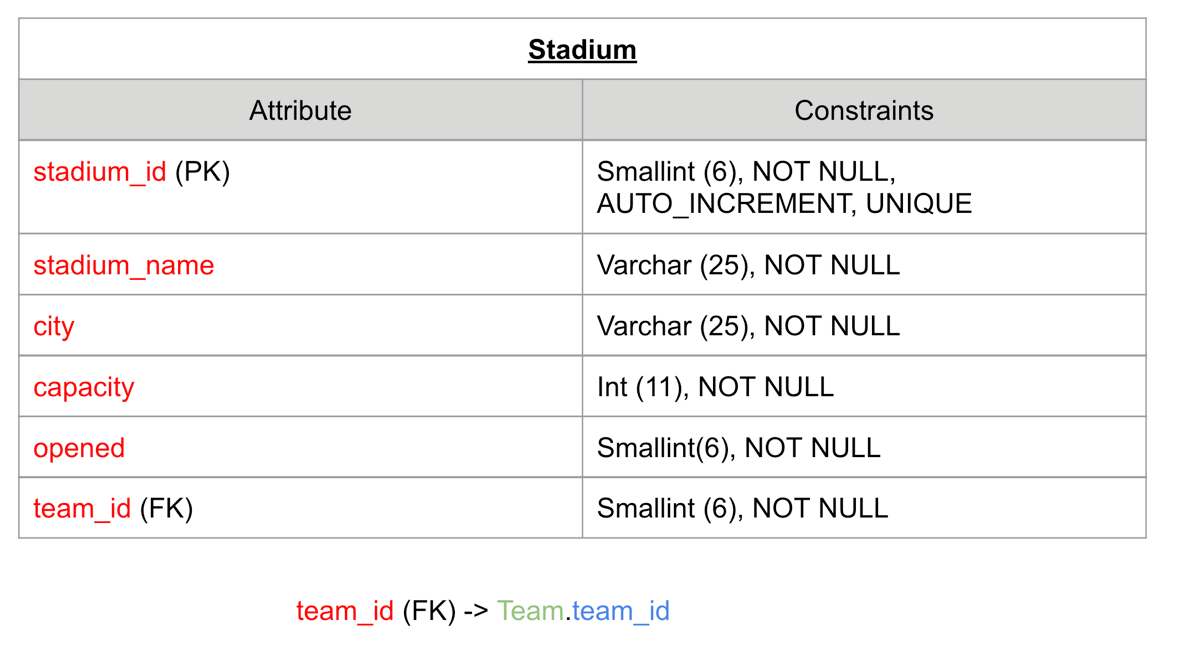
For *player\_id* I chose to use int() as the storage type since there could be over 32,767 entries of players which is the maximum provided by smallint(). I chose to use varchar of size 15 and 10 to represent player names, positions and countries and I felt this size would suffice. For *team\_id* I used smallint() as I believe there will be less than 32,767 entries of teams.



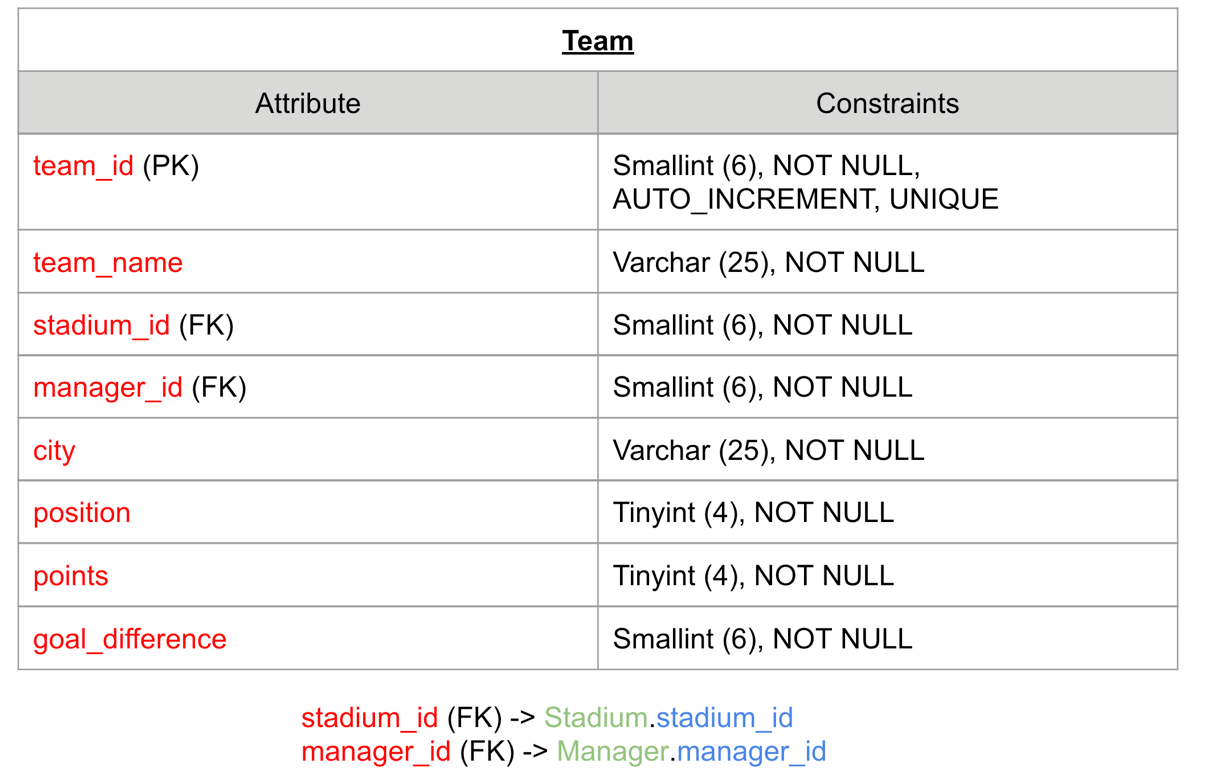
For *manager\_id* I chose to use int() as the storage type since there could be over 32,767 entries of managers which is the maximum provided by smallint(). For *age* I used tinyint() as it is extremely unlikely a manager will be older than 255. For *salary* I used an int() as a managers salary is likely to be over £32,767.



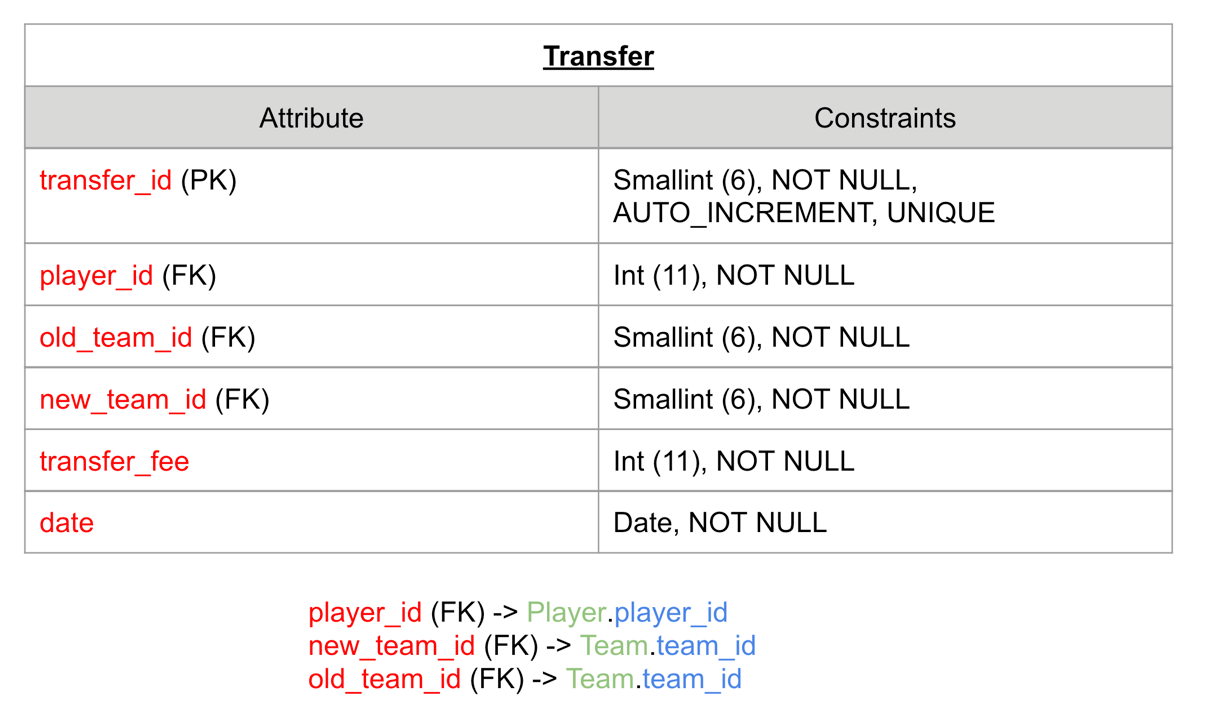
For *result\_id* I chose to use int() as the storage type since there could be over 32,767 entries of results which is the maximum provided by smallint(). For *home\_goals* and *away\_goals* I used tinyint() as it is extremely unlikely a given team will score more than 255 goals. For *result* I limited the varchar() storage type to 4 characters as a result will be either ‘win’ or ‘draw’.



For *stadium\_id* I chose to use smallint() as the storage type since it is unlikely for there to be over 32,767 entries of stadiums which is the maximum provided by smallint(). For *opened* I used smallint() instead of year since year only supports values after 1901 and some stadiums in the premier league were older than this.



For *team\_id* I chose to use smallint() as the storage type since it is unlikely for there to be over 32,767 entries of teams which is the maximum provided by smallint(). For *position* I used tinyint() as players numbers don’t tend to be more than 255. For *goal\_difference* I used smallint() as I needed the smallest data type available to support negative numbers.



For *transfer\_id* I chose to use smallint() as the storage type since it is unlikely for there to be over 32,767 entries of transfers which is the maximum provided by smallint(). For *transfer\_fee* I used int() as transfer fees tend to be in the millions and this was the smallest data type that supported a suitable range.

All of the above data types and constraints were chosen in order to:

1. Use minimal storage
2. Ensure validity of data entries
3. Ensure consistency across all relations