GoSports Final Individual Report - Sam DeFrancisco (sjdefran)

Python 3, Django, sqLite3, HTML

5/1/2022

SE/COMS 319

Successful Implemented Story Cards

1. Gather Common Player info and Reorganize Django Models

   Solution: We needed to gather common player info, stuff like height, weight, position. We also wanted to grab every season of their career for their player page. This raised questions for me on the DB design. I decided to reorganize the player model to store their common info rather than current season data. Next I created a model called PlayerSeasons, this was the new db table that held each active players career season data which was connected to the player using foreign keys (player_id).

   Code:

```
# returns most recent season stats for a player
def player_recent_season_stats(playerID):
    gathered = False
    while gathered == False:
        try:
            player_stats =
            ↪  playerprofilev2.PlayerProfileV2(player_id=playerID,
            ↪  per_mode36='PerGame', timeout=10, headers=headers)
            gathered = True # got data
            player_stats = player_stats.season_totals_regular_season
            player_stats = player_stats.get_data_frame()
            # some players don't have 2021-2022 data, return as empty list
            if player_stats.empty == True:
                return []

            player_stats = player_stats.iloc[-1]
        # chooses last row of dataframe (most recent season)
        except:
            print('gather failed for player.. Sleeping for 10sec...Trying
            ↪  again')
            time.sleep(10)
    return player_stats

# returns all seasons for player
def player_seasons(playerID):
    gathered = False
    while gathered == False:
        try:
            player_stats =
             ↪  playerprofilev2.PlayerProfileV2(player_id=playerID,
             ↪  per_mode36='PerGame', timeout=10, headers=headers)
            gathered = True # got data
            player_stats = player_stats.season_totals_regular_season
            player_stats = player_stats.get_data_frame()
            # some players don't have 2021-2022 data, return as empty list
            if player_stats.empty == True:
                return []
            exists = False
        except:
            print('gather failed for player.. Sleeping for 10sec...Trying
             ↪  again')
            time.sleep(10)
    return player_stats
```

2. Update Team Player Pages

Solution: Similar reorganization was needed for the Team model as well due to gathering each teams last 5 seasons for their page. I removed current season data from the Team Model and added a new table/model TeamSeasons. I then changed the html templates to display the teams last 5 seasons rather than just one. I also changed the player html page to display each season of their career.

Code:

```
class Teams(models.Model):
    index = models.IntegerField(blank=True, null=True)
    name = models.TextField(db_column='TEAM_NAME', blank=True, null=True)   #
     ↪  'Boston Celtics'
    team_id = models.IntegerField(primary_key=True, db_column='TEAM_ID',
     ↪  blank=True, null=False)  # '1610612737'

    class Meta:
        managed = True
        db_table = 'Teams'


# holds each teams last 5 seaons stats
# PK: index
# FK: team, which allows you to get any teams last 5 seaons the same way as
↪  players
class TeamSeasons(models.Model):
    index = models.IntegerField(blank=True, null=False, primary_key=True)
    team = models.ForeignKey(Teams, on_delete=models.CASCADE)
    year = models.TextField(db_column='YEAR', blank=True, null=True)  #
    ↪  '2017-18'
    gp = models.IntegerField(db_column='GP', blank=True, null=True)
    wins = models.IntegerField(db_column='WINS', blank=True, null=True)
    losses = models.IntegerField(db_column='LOSSES', blank=True, null=True)
    win_pct = models.FloatField(db_column='WIN_PCT', blank=True, null=True)
    conf_rank = models.IntegerField(db_column='CONF_RANK', blank=True,
    ↪  null=True)
    div_rank = models.IntegerField(db_column='DIV_RANK', blank=True,
    ↪  null=True)
    fgm = models.FloatField(db_column='FGM', blank=True, null=True)
    fga = models.FloatField(db_column='FGA', blank=True, null=True)
    fg_pct = models.FloatField(db_column='FG_PCT', blank=True, null=True)
    fg3m = models.FloatField(db_column='FG3M', blank=True, null=True)
    fg3a = models.FloatField(db_column='FG3A', blank=True, null=True)
    fg3_pct = models.FloatField(db_column='FG3_PCT', blank=True, null=True)
    ftm = models.FloatField(db_column='FTM', blank=True, null=True)
    fta = models.FloatField(db_column='FTA', blank=True, null=True)
    ft_pct = models.FloatField(db_column='FT_PCT', blank=True, null=True)
    oreb = models.FloatField(db_column='OREB', blank=True, null=True)
    dreb = models.FloatField(db_column='DREB', blank=True, null=True)
    reb = models.FloatField(db_column='REB', blank=True, null=True)
    ast = models.FloatField(db_column='AST', blank=True, null=True)
    pf = models.FloatField(db_column='PF', blank=True, null=True)
    stl = models.FloatField(db_column='STL', blank=True, null=True)
    tov = models.FloatField(db_column='TOV', blank=True, null=True)
    blk = models.FloatField(db_column='BLK', blank=True, null=True)
    pts = models.FloatField(db_column='PTS', blank=True, null=True)
```

3. Gatherin player pictures

   Solution: Every headshot on nba.com uses a similar link. "https://ak-static.cms.nba.com/wp-content/uploads/headshots/nba/latest/260x190/1628369.png". The only thing that differs is the player_id at the end of the link. Luckily we already store each players player_id and use it quite frequently. Using django's template language we can access the specific players id and append it to the end of the link within the img tag

   Code:

```
<div>
    <img
     ↪  src="https://ak-static.cms.nba.com/wp-content/uploads/headshots/nba/latest/260x19
    {{player.person_id}}.png" style="  width: 250px;    ;" alt="Player
 ↪  Picture" align="left">
    <p> Name: {{player.full_name}}</p>
    <p> Team: <a href="{% url 'team' team_name %}"> {{team_name}} </a> </p>
    <p> Height: {{player.height}}</p>
    <p> Weight: {{player.weight}}</p>
    <p style="margin-left: 17%;"> Position: {{player.position}}</p>
    <p style="margin-left: 17%;"> Jersey Number: {{player.jersey}}</p>

</div>
```

Level of Proficiency

Now that this project has finished I can look back and see how much I learned. The best way to learn about how to use coding languages is to actually build something with them. I feel much more confident with Python and sql then I did first coming into the project. Learning how Django works was extremely useful, we have developed a full stack app and seen all sides of the development (not deployment). I had very limited expierence with HTML & CSS before this project and now feel much more comfortable using it.

Contributions

Figure 1: Num Commits

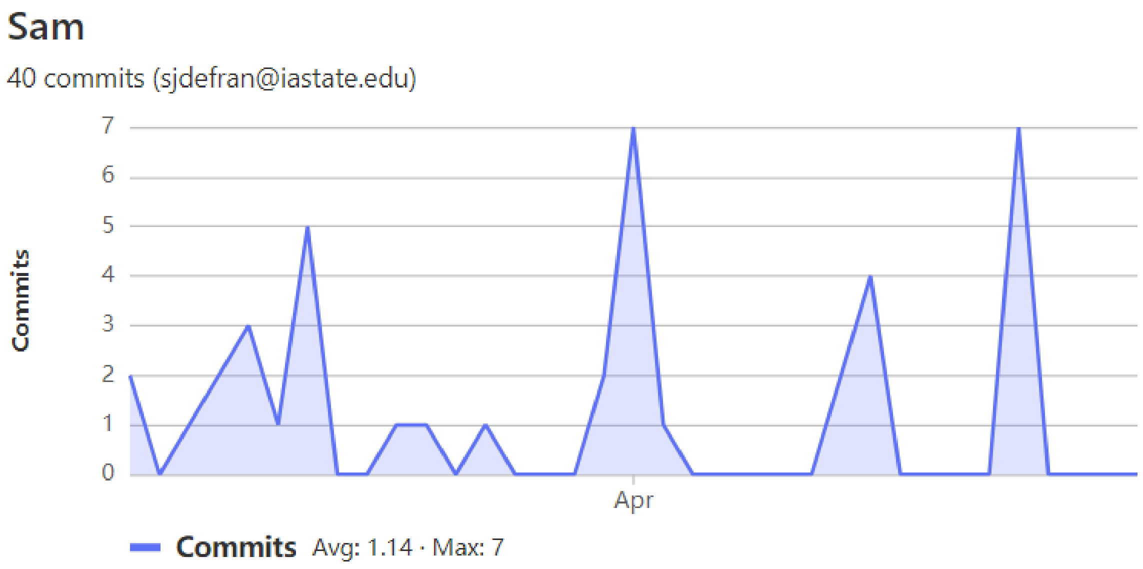
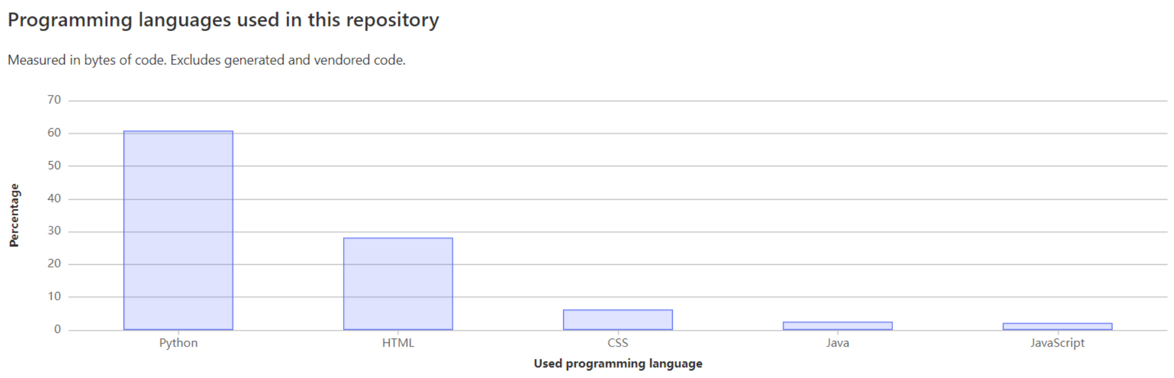


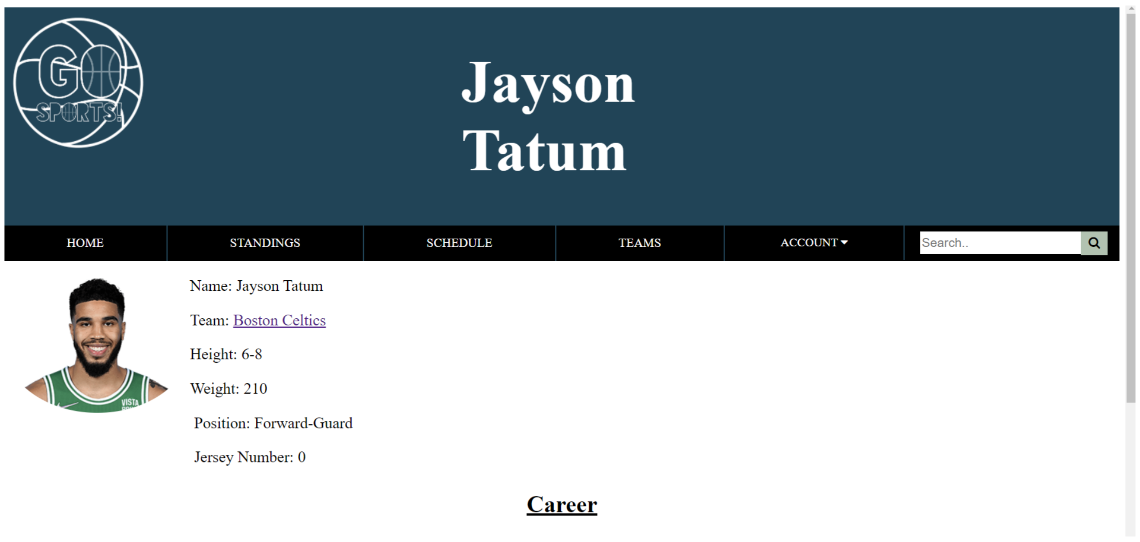
Figure 2: Languages Used



- Python - 60%
- HTML - 30%
- CSS - 6%
- Java - 2%
- JavaScipt - 2%

Screen Shots

Figure 3: New Player Page (1)



Name: Jayson Tatum

Team: Boston Celtics

Height: 6-8

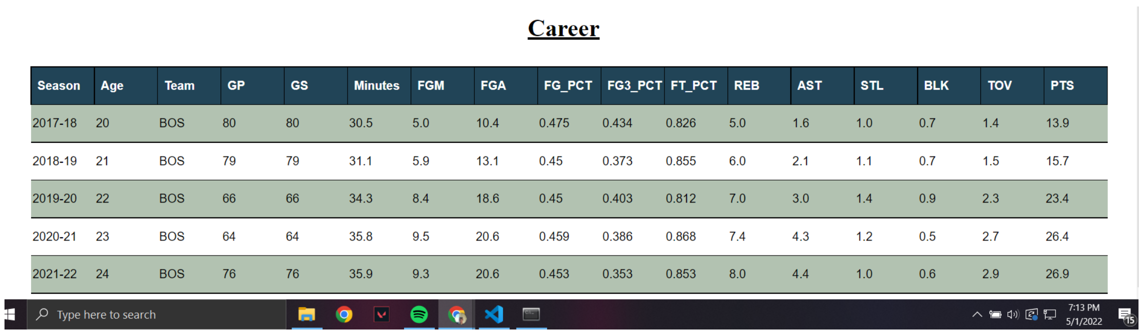
Weight: 210

Position: Forward-Guard

Jersey Number: 0

Career

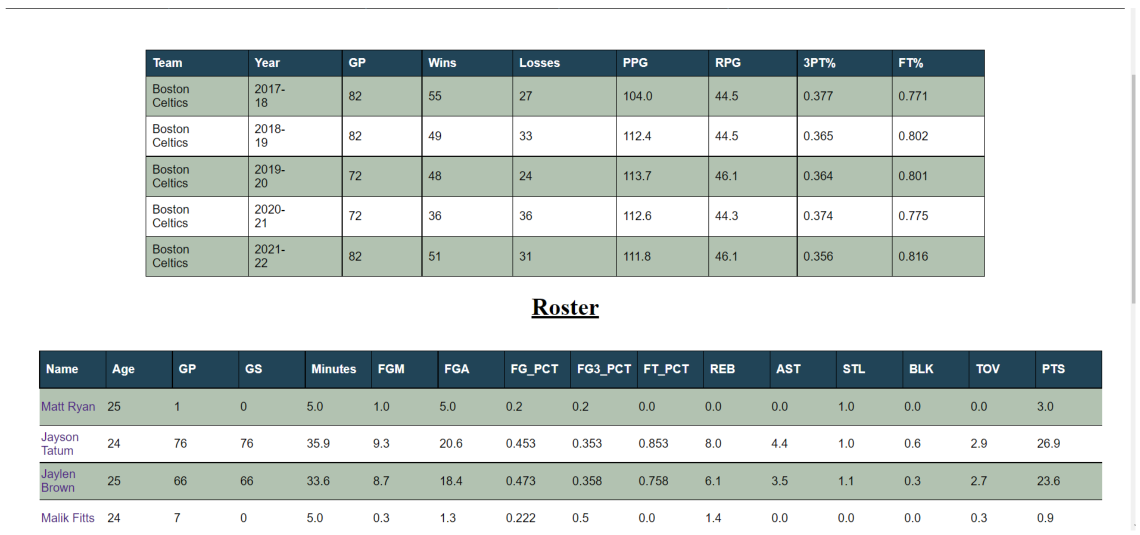
Figure 4: New Player Page (2)



Career

Season	Age	Team	GP	GS	Minutes	FGM	FGA	FG_PCT	FG3_PCT	FT_PCT	REB	AST	STL	BLK	TOV	PTS
2017-18	20	BOS	80	80	30.5	5.0	10.4	0.475	0.434	0.826	5.0	1.6	1.0	0.7	1.4	13.9
2018-19	21	BOS	79	79	31.1	5.9	13.1	0.45	0.373	0.855	6.0	2.1	1.1	0.7	1.5	15.7
2019-20	22	BOS	66	66	34.3	8.4	18.6	0.45	0.403	0.812	7.0	3.0	1.4	0.9	2.3	23.4
2020-21	23	BOS	64	64	35.8	9.5	20.6	0.459	0.386	0.868	7.4	4.3	1.2	0.5	2.7	26.4
2021-22	24	BOS	76	76	35.9	9.3	20.6	0.453	0.353	0.853	8.0	4.4	1.0	0.6	2.9	26.9

Figure 5: New Team Page

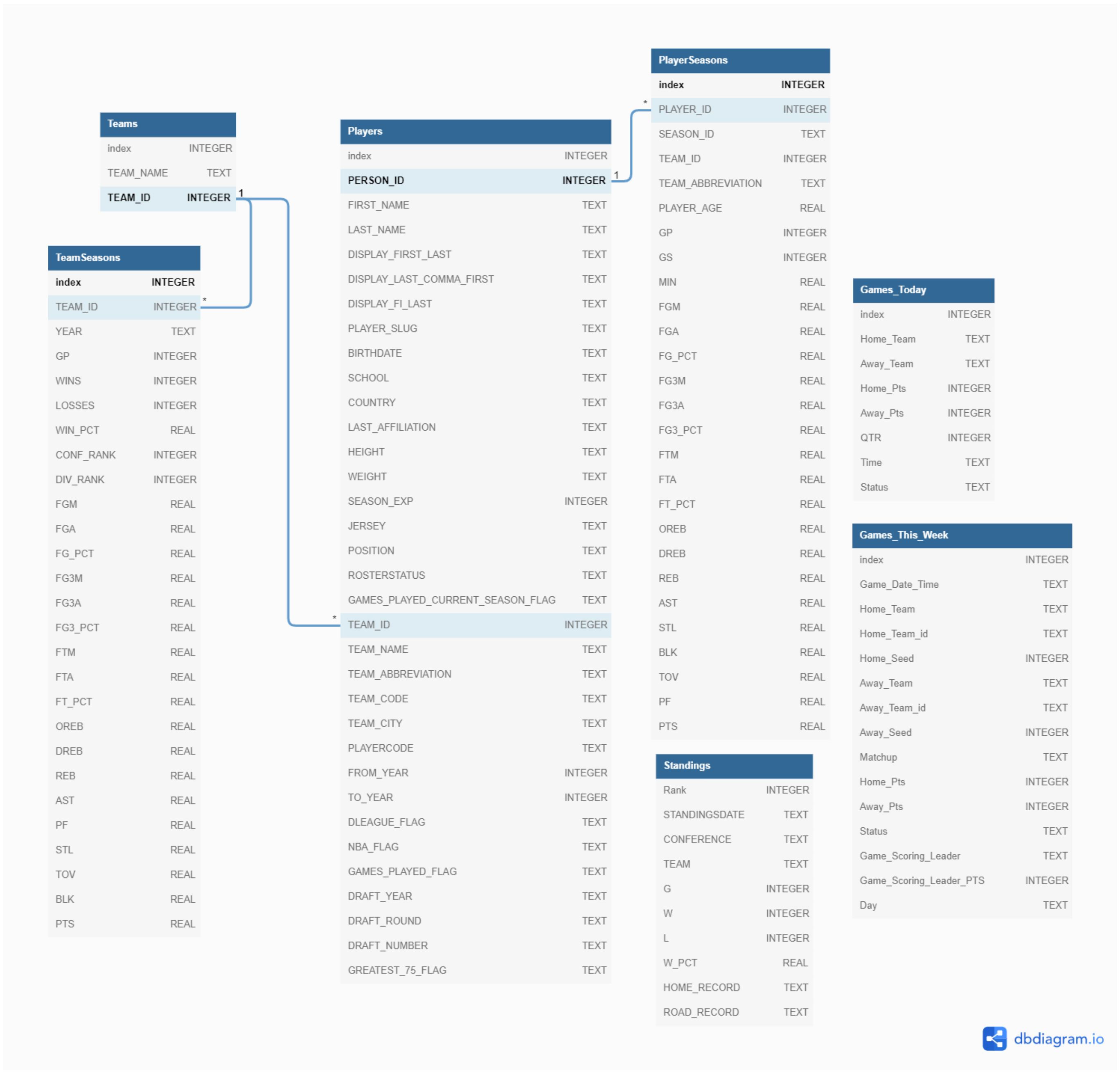


Team	Year	GP	Wins	Losses	PPG	RPG	3PT%	FT%
Boston Celtics	2017-18	82	55	27	104.0	44.5	0.377	0.771
Boston Celtics	2018-19	82	49	33	112.4	44.5	0.365	0.802
Boston Celtics	2019-20	72	48	24	113.7	46.1	0.364	0.801
Boston Celtics	2020-21	72	36	36	112.6	44.3	0.374	0.775
Boston Celtics	2021-22	82	51	31	111.8	46.1	0.356	0.816

Roster

Name	Age	GP	GS	Minutes	FGM	FGA	FG_PCT	FG3_PCT	FT_PCT	REB	AST	STL	BLK	TOV	PTS
Matt Ryan	25	1	0	5.0	1.0	5.0	0.2	0.2	0.0	0.0	0.0	1.0	0.0	0.0	3.0
Jayson Tatum	24	76	76	35.9	9.3	20.6	0.453	0.353	0.853	8.0	4.4	1.0	0.6	2.9	26.9
Jaylen Brown	25	66	66	33.6	8.7	18.4	0.473	0.358	0.758	6.1	3.5	1.1	0.3	2.7	23.6
Malik Fitts	24	7	0	5.0	0.3	1.3	0.222	0.5	0.0	1.4	0.0	0.0	0.0	0.3	0.9

Figure 6: Database Design



Teams	
index	INTEGER
TEAM_NAME	TEXT
TEAM_ID	INTEGER

TeamSeasons	
index	INTEGER
TEAM_ID	INTEGER
YEAR	TEXT
GP	INTEGER
WINS	INTEGER
LOSSES	INTEGER
WIN_PCT	REAL
CONF_RANK	INTEGER
DIV_RANK	INTEGER
FGM	REAL
FGA	REAL
FG_PCT	REAL
FG3M	REAL
FG3A	REAL
FG3_PCT	REAL
FTM	REAL
FTA	REAL
FT_PCT	REAL
OREB	REAL
DREB	REAL
REB	REAL
AST	REAL
PF	REAL
STL	REAL
TOV	REAL
BLK	REAL
PTS	REAL

Players	
index	INTEGER
PERSON_ID	INTEGER
FIRST_NAME	TEXT
LAST_NAME	TEXT
DISPLAY_FIRST_LAST	TEXT
DISPLAY_LAST_COMMA_FIRST	TEXT
DISPLAY_FI_LAST	TEXT
PLAYER_SLUG	TEXT
BIRTHDATE	TEXT
SCHOOL	TEXT
COUNTRY	TEXT
LAST_AFFILIATION	TEXT
HEIGHT	TEXT
WEIGHT	TEXT
SEASON_EXP	INTEGER
JERSEY	TEXT
POSITION	TEXT
ROSTERSTATUS	TEXT
GAMES_PLAYED_CURRENT_SEASON_FLAG	TEXT
TEAM_ID	INTEGER
TEAM_NAME	TEXT
TEAM_ABBREVIATION	TEXT
TEAM_CODE	TEXT
TEAM_CITY	TEXT
PLAYERCODE	TEXT
FROM_YEAR	INTEGER
TO_YEAR	INTEGER
DLEAGUE_FLAG	TEXT
NBA_FLAG	TEXT
GAMES_PLAYED_FLAG	TEXT
DRAFT_YEAR	TEXT
DRAFT_ROUND	TEXT
DRAFT_NUMBER	TEXT
GREATEST_75_FLAG	TEXT

PlayerSeasons	
index	INTEGER
PLAYER_ID	INTEGER
SEASON_ID	TEXT
TEAM_ID	INTEGER
TEAM_ABBREVIATION	TEXT
PLAYER_AGE	REAL
GP	INTEGER
GS	INTEGER
MIN	REAL
FGM	REAL
FGA	REAL
FG_PCT	REAL
FG3M	REAL
FG3A	REAL
FG3_PCT	REAL
FTM	REAL
FTA	REAL
FT_PCT	REAL
OREB	REAL
DREB	REAL
REB	REAL
AST	REAL
STL	REAL
BLK	REAL
TOV	REAL
PF	REAL
PTS	REAL

Standings	
Rank	INTEGER
STANDINGSDATE	TEXT
CONFERENCE	TEXT
TEAM	TEXT
G	INTEGER
W	INTEGER
L	INTEGER
W_PCT	REAL
HOME_RECORD	TEXT
ROAD_RECORD	TEXT

Games_Today	
index	INTEGER
Home_Team	TEXT
Away_Team	TEXT
Home_Pts	INTEGER
Away_Pts	INTEGER
QTR	INTEGER
Time	TEXT
Status	TEXT

Games_This_Week	
index	INTEGER
Game_Date_Time	TEXT
Home_Team	TEXT
Home_Team_id	TEXT
Home_Seed	INTEGER
Away_Team	TEXT
Away_Team_id	TEXT
Away_Seed	INTEGER
Matchup	TEXT
Home_Pts	INTEGER
Away_Pts	INTEGER
Status	TEXT
Game_Scoring_Leader	TEXT
Game_Scoring_Leader_PTS	INTEGER
Day	TEXT