Extract Transform Load NBA data

Project 2 - ETL

Jasmine Jones, Nicholas Noark, Haydn Whitmyer

Extract: API-NBA

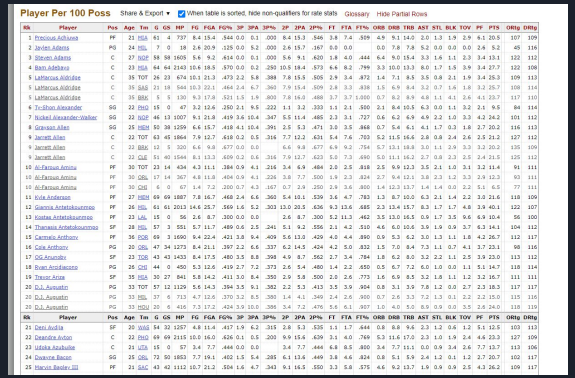
- API-NBA - “freemium” API from rapidapi.com
- Limited to 10 calls/minute and 100 calls/day
- Contains data on NBA Stats DATA, Games, Livescore, Standings, Statistics, Teams, Players, Seasons, Leagues.
- Updated on a semi-regular basis, has info on recent games (up to 10/16/21)
- Selected to pull from Game Details, Team (ID), and Players (ID)

	game_id	season_year	arena	city	country	start_time_UTC	game_duration	home_team	home_score	home_leader_id	away_team	away_score	away_leader_id
0	9061	2020	United Center	Chicago	USA	2021-04-15T00:00:00.000Z	2:07	Chicago Bulls	106	534	Orlando Magic	115	160
1	1067	2015				2016-03-10T02:30:00.000Z	2:18	Oklahoma City Thunder	120	153	LA Clippers	108	207
2	5906	2018	Barclays Center	Brooklyn	USA	2019-03-30T22:00:00.000Z	2:08	Brooklyn Nets	110	462	Boston Celtics	96	227
3	4431	2018	Vivint Smart Home Arena	Salt Lake City	USA	2018-10-23T01:00:00.000Z	2:25	Utah Jazz	84	121	Memphis Grizzlies	92	114
4	3000	2017	American Airlines Center	Dallas	USA	2017-10-29T00:30:00.000Z	2:08	Dallas Mavericks	110	36	Philadelphia 76ers	112	159
5	420	2015	Target Center	Minneapolis		2015-12-08T01:00:00.000Z	2:39	Minnesota Timberwolves	106	308	LA Clippers	110	286
6	1642	2016	Talking Stick Resort Arena	Phoenix		2016-11-10T02:00:00.000Z	2:16	Phoenix Suns	107	59	Detroit Pistons	100	89
7	130	2015	Quicken Loans Arena	Cleveland		2015-10-30T23:00:00.000Z	2:17	Cleveland Cavaliers	102	265	Miami Heat	92	536
8	5240	2018	Bankers Life Fieldhouse	Indianapolis	USA	2019-02-14T00:00:00.000Z	2:10	Indiana Pacers	97	60	Milwaukee Bucks	106	20
9	5123	2018	Chesapeake Energy Arena	Oklahoma City	USA	2019-01-27T23:00:00.000Z	2:20	Oklahoma City Thunder	118	189	Milwaukee Bucks	112	20

```
# game details info API to calls

details_base_url = "https://api-nba-v1.p.rapidapi.com/gameDetails/"

game_numbers = list(np.random.randint(10861, size=10))
game_numbers

game_info = []
game_not_found = []

for number in game_numbers:

    game_url = details_base_url + str(number)

    try:
        game_response = requests.get(game_url,headers=headers).json()

        gameId = game_response['api']['game'][0]['gameId']
        seasonYear = game_response['api']['game'][0]['seasonYear']
        arena = game_response['api']['game'][0]['arena']
        city = game_response['api']['game'][0]['city']
        country = game_response['api']['game'][0]['country']
        startTimeUTC = game_response['api']['game'][0]['startTimeUTC']
        gameDuration = game_response['api']['game'][0]['gameDuration']
        vTeam = game_response['api']['game'][0]['vTeam']['fullName']
        vTeamLeader = game_response['api']['game'][0]['vTeam']['leaders'][0]['playerId']
        awayScore = game_response['api']['game'][0]['vTeam']['score']['points']
        hTeam = game_response['api']['game'][0]['hTeam']['fullName']
        hTeamLeader = game_response['api']['game'][0]['hTeam']['leaders'][0]['playerId']
        homeScore = game_response['api']['game'][0]['hTeam']['score']['points']

        game_info.append({
            'game_id':gameId,
            'season_year':seasonYear,
            'arena':arena,
            'city':city,
            'country':country,
            'start_time_UTC':startTimeUTC,
            'game_duration':gameDuration,
            'home_team':hTeam,
            'home_score':homeScore,
            'home_leader_id':hTeamLeader,
            'away_team':vTeam,
            'away_score':awayScore,
            'away_leader_id':vTeamLeader
        })

# store any values not found and pass to keep the loop running
    except:
        game_not_found.append(gameId)
        pass
```

Extract: Basketball-Reference & SQLite Database

- Using Pandas and BeautifulSoup, scraped per 100 possessions and per 36 minute player statistics for the 2020-21 NBA season
- From a NBA SQLite database found on kaggle.com, pulled player draft data and player salaries for the 2020-21 NBA season
- The SQLite database was found at https://www.kaggle.com/wyattowalsh/basketball

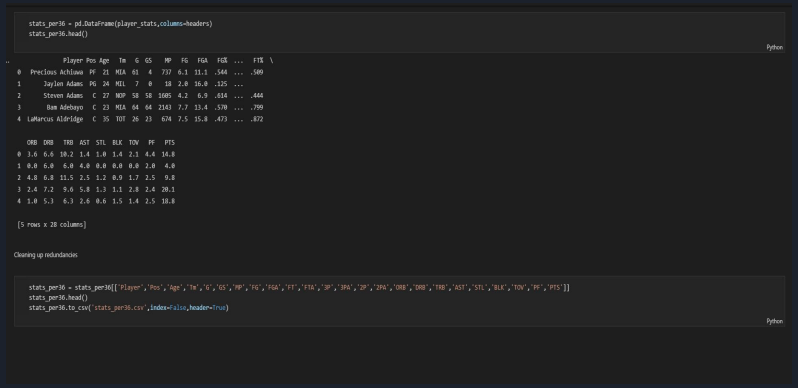

Player Per 100 Poss

Share & Export ▾ When table is sorted, hide non-qualifiers for rate stats Glossary Hide Partial Rows

Rk	Player	Pos	Age	Tm	G	GS	MP	FG	FGA	FG%	3P	3PA	3P%	2P	2PA	2P%	FT	FTA	FT%	ORB	DRB	TRB	AST	STL	BLK	TOV	PF	PTS	ORtg	DRtg
1	Precious Achiuwa	PF	21	MIA	61	4	737	8.4	15.4	.544	0.0	0.1	.000	8.4	15.3	.546	3.8	7.4	.509	4.9	9.1	14.0	2.0	1.3	1.9	2.9	6.1	20.5	107	109
2	Jaylen Adams	PG	24	MIL	7	0	18	2.4	20.9	.125	0.0	5.2	.000	2.4	15.7	.167	0.0	0.0		0.0	7.8	7.8	5.2	0.0	0.0	0.0	2.4	5.2	45	116
3	Steven Adams	C	27	NOP	58	58	1605	5.6	9.2	.614	0.0	0.1	.000	5.6	9.1	.620	1.8	4.0	.444	6.4	9.0	15.4	3.3	1.6	1.1	2.3	3.4	13.1	122	112
4	Bam Adebayo	C	23	MIA	64	64	2143	10.6	18.5	.570	0.0	0.2	.250	10.5	18.4	.573	6.6	8.2	.799	3.3	10.0	13.3	8.0	1.7	1.5	3.9	3.4	27.7	122	108
5	LaMarcus Aldridge	C	35	TOT	26	23	674	10.1	21.3	.473	2.2	5.8	.388	7.8	15.5	.505	2.9	3.4	.872	1.4	7.1	8.5	3.5	0.8	2.1	1.9	3.4	25.3	109	113
5	LaMarcus Aldridge	C	35	SAS	21	19	566	10.3	22.1	.464	2.4	6.7	.360	7.9	15.4	.509	2.8	3.3	.838	1.5	6.9	8.4	3.2	0.7	1.6	1.8	3.2	25.7	108	114
5	LaMarcus Aldridge	C	35	BRK	5	5	130	9.3	17.8	.521	1.5	1.9	.800	7.8	16.0	.488	3.7	3.7	1.000	0.7	8.2	8.9	4.8	1.1	4.1	2.6	4.1	23.7	117	110
6	Tr-Shon Alexander	SG	22	PHO	15	0	47	3.2	12.6	.250	2.1	9.5	.222	1.1	3.2	.333	1.1	2.1	.500	2.1	8.4	10.5	6.3	0.0	1.1	3.2	2.1	9.5	84	114
7	Nickeil Alexander-Walker	SG	22	NOP	46	13	1007	9.1	21.8	.419	3.6	10.4	.347	5.5	11.4	.480	2.3	3.1	.727	0.6	6.2	6.9	4.9	2.2	1.0	3.3	4.2	24.2	101	112
8	Grayson Allen	SG	25	MEM	50	38	1259	6.6	15.7	.418	4.1	10.4	.391	2.5	5.3	.471	3.0	3.5	.868	0.7	5.4	6.1	4.1	1.7	0.3	1.8	2.7	20.2	116	113
9	Jarrett Allen	C	22	TOT	63	63	1864	7.9	12.7	.618	0.2	0.5	.316	7.7	12.2	.631	5.4	7.6	.703	5.2	11.5	16.6	2.8	0.8	2.4	2.6	2.5	21.2	127	112
9	Jarrett Allen	C	22	BRK	63	0	1320	6.6	9.8	.677	0.0	0.0		6.6	9.8	.677	4.9	6.2	.794	5.7	11.5	16.8	3.0	1.1	2.9	3.3	3.2	20.2	135	109
9	Jarrett Allen	C	22	CLE	51	40	1544	8.1	13.3	.609	0.2	0.6	.316	7.9	12.7	.623	5.0	7.3	.690	5.0	11.1	16.2	2.7	0.8	2.3	2.5	2.4	21.5	125	112
10	Al-Farouq Aminu	PF	30	TOT	23	14	434	4.3	11.1	.384	0.9	4.1	.216	3.4	6.9	.484	2.0	2.5	.818	2.5	9.9	12.3	3.5	2.1	1.0	3.1	3.2	11.4	91	111
10	Al-Farouq Aminu	PF	30	ORL	17	14	367	4.8	11.8	.404	0.9	4.1	.226	3.8	7.7	.500	1.9	2.3	.824	2.7	9.4	12.1	3.8	2.3	1.2	3.3	2.9	12.3	93	111
10	Al-Farouq Aminu	PF	30	CHI	6	0	67	1.4	7.2	.200	0.7	4.3	.167	0.7	2.9	.250	2.9	3.6	.800	1.4	12.3	13.7	1.4	1.4	0.0	2.2	5.1	6.5	77	111
11	Kyle Anderson	PF	27	MEM	69	69	1887	7.8	16.7	.468	2.4	6.6	.360	5.4	10.1	.539	3.6	4.7	.783	1.3	8.7	10.0	6.3	2.1	1.4	2.2	3.0	21.6	118	109
12	Giannis Antetokounmpo	PF	26	MIL	61	61	2013	14.6	25.7	.569	1.6	5.2	.303	13.0	20.5	.636	9.2	13.6	.685	2.3	13.4	15.7	8.3	1.7	1.7	4.8	3.9	40.1	122	107
13	Kostas Antetokounmpo	PF	23	LAL	15	0	96	2.6	8.7	.300	0.0	0.0		2.6	8.7	.300	5.2	11.3	.462	3.5	13.0	16.5	0.9	1.7	3.5	9.6	6.9	10.4	56	100
14	Thanasis Antetokounmpo	SF	28	MIL	57	3	551	5.7	11.7	.489	0.6	2.5	.241	5.1	9.2	.556	2.1	4.2	.510	4.6	6.0	10.6	3.9	1.9	0.9	3.7	6.3	14.1	104	112
15	Carmelo Anthony	PF	36	POR	69	3	1690	9.4	22.4	.421	3.8	9.4	.409	5.6	13.0	.429	4.0	4.4	.890	0.9	5.3	6.2	3.0	1.3	1.1	1.8	4.2	26.7	112	117
16	Cole Anthony	PG	20	ORL	47	34	1273	8.4	21.1	.397	2.2	6.6	.337	6.2	14.5	.424	4.2	5.0	.832	1.5	7.0	8.4	7.3	1.1	0.7	4.1	3.7	23.1	98	116
17	OG Anunoby	SF	23	TOR	43	43	1433	8.4	17.5	.480	3.5	8.8	.398	4.9	8.7	.562	2.7	3.4	.784	1.8	6.2	8.0	3.2	2.2	1.1	2.5	3.9	23.0	113	112
18	Ryan Arcidiacono	PG	26	CHI	44	0	450	5.3	12.6	.419	2.7	7.2	.373	2.6	5.4	.480	1.4	2.2	.650	0.5	6.7	7.2	6.0	1.0	0.0	1.1	5.1	14.7	118	114
19	Trevor Ariza	SF	35	PHO	30	27	841	5.8	14.2	.411	3.0	8.4	.350	2.9	5.8	.500	2.0	2.6	.775	1.6	6.9	8.5	3.2	1.8	1.1	1.2	3.2	16.7	111	111
20	D.J. Augustin	PG	33	TOT	57	12	1229	5.6	14.3	.394	3.5	9.1	.382	2.2	5.3	.413	3.5	3.9	.904	0.8	3.1	3.9	7.8	1.2	0.0	2.7	2.3	18.3	117	117
20	D.J. Augustin	PG	33	MIL	37	6	713	4.7	12.6	.370	3.2	8.5	.380	1.4	4.1	.349	2.4	2.6	.900	0.7	2.6	3.3	7.2	1.3	0.1	2.2	2.2	15.0	115	116
20	D.J. Augustin	PG	33	HOU	20	6	416	7.3	17.2	.424	3.9	10.0	.386	3.4	7.2	.476	5.6	6.1	.907	1.0	4.0	5.0	8.9	0.9	0.0	3.5	2.6	24.0	118	119
Rk	Player	Pos	Age	Tm	G	GS	MP	FG	FGA	FG%	3P	3PA	3P%	2P	2PA	2P%	FT	FTA	FT%	ORB	DRB	TRB	AST	STL	BLK	TOV	PF	PTS	ORtg	DRtg
21	Deni Avdija	SF	20	WAS	54	32	1257	4.8	11.4	.417	1.9	6.2	.315	2.8	5.3	.539	1.1	1.7	.644	0.8	6.8	9.6	2.3	1.2	0.6	1.2	3.1	12.5	103	113
22	Deandre Ayton	C	22	PHO	69	69	2115	10.0	16.0	.626	0.1	0.5	.200	9.9	15.6	.629	3.1	4.0	.769	5.3	11.6	17.0	2.3	1.0	1.9	2.4	4.6	23.3	127	109
23	Udoka Azubuike	C	21	UTA	15	0	57	3.4	7.7	.444	0.0	0.0		3.4	7.7	.444	6.8	8.5	.800	3.4	7.7	11.1	0.0	0.9	3.4	2.6	7.7	13.7	113	106
24	Dwayne Bacon	SG	25	ORL	71	55	1853	7.7	19.1	.402	1.5	5.4	.285	6.1	13.6	.449	3.8	4.6	.824	0.8	5.1	5.9	2.4	1.2	0.1	1.2	3.7	20.7	102	117
25	Marvin Bagley III	PF	21	SAC	43	42	1112	10.7	21.2	.504	1.6	4.7	.343	9.1	16.5	.550	3.3	5.0	.675	4.6	9.2	13.7	1.9	0.9	0.9	2.5	4.3	26.2	109	117

Transform

- Once extracted, data was cleaned using pandas. Statistics such as FG%, 3P%, and FT% were removed from the per 36 minutes data since rate stats stay the same across per 100 possession stats and per 36 minutes
- Inactive players filtered out of the SQLite data then merged with the draft data from kaggle.com
- Pulled relevant basic info (name, team, DOB, city, etc.) on players and teams from API; Basketball-Reference contains more complete statistics
- Many players from outside USA lacked data, these rows were removed with .dropna( )
- Reset indices to teamID, playerID, and gameID using .set_index( )



```
stats_per36 = pd.DataFrame(player_stats,columns=headers)
stats_per36.head()
```

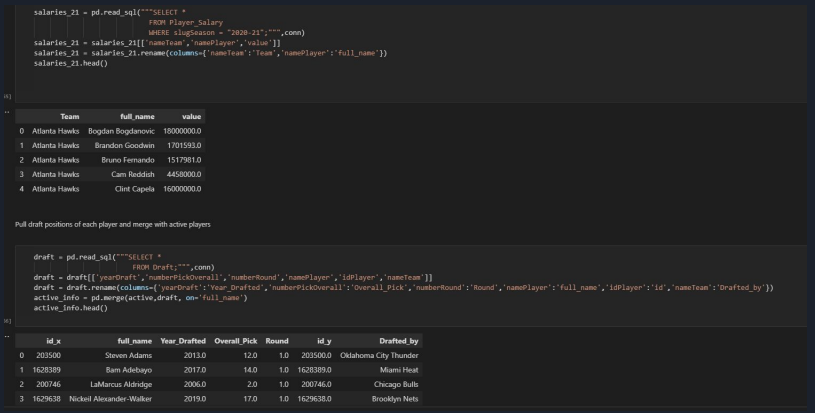
Python

```
              Player Pos Age   Tm  G  GS    MP   FG   FGA   FG%  ...   FT% \
0   Precious Achiuwa  PF  21  MIA  61   4   737  6.1  11.1  .544  ...  .589
1       Jaylen Adams  PG  24  MIL   7   0    18  2.0  16.0  .125  ...
2       Steven Adams   C  27  NOP  58  58  1685  4.2   6.9  .616  ...  .444
3       Bam Adebayo    C  23  MIA  64  64  2143  7.7  13.4  .570  ...  .799
4  LaMarcus Aldridge   C  35  TOT  26  23   674  7.5  15.8  .473  ...  .872

   ORB  DRB  TRB  AST  STL  BLK  TOV   PF   PTS
0  3.6  6.6  10.2  1.4  1.0  1.4  2.1  4.4  14.8
1  0.0  6.0   6.0  4.0  0.0  0.0  0.0  2.0   4.0
2  4.8  6.8  11.5  2.5  1.2  0.9  1.7  2.5   9.8
3  2.4  7.2   9.6  5.8  1.3  1.1  2.8  2.4  20.1
4  1.0  5.3   6.3  2.6  0.6  1.5  1.4  2.5  10.8

[5 rows x 28 columns]
```

Cleaning up redundancies

```
stats_per36 = stats_per36[['Player','Pos','Age','Tm','G','GS','MP','FG','FGA','FT','FTA','3P','3PA','2P','2PA','ORB','DRB','TRB','AST','STL','BLK','TOV','PF','PTS']]
stats_per36.head()
stats_per36.to_csv('stats_per36.csv',index=False,header=True)
```

Python



```
salaries_21 = pd.read_sql("""SELECT *
                             FROM Player_Salary
                             WHERE slugSeason = "2020-21";""",conn)
salaries_21 = salaries_21[['nameTeam','namePlayer','value']]
salaries_21 = salaries_21.rename(columns={'nameTeam':'Team','namePlayer':'full_name'})
salaries_21.head()
```

	Team	full_name	value
0	Atlanta Hawks	Bogdan Bogdanovic	18000000.0
1	Atlanta Hawks	Brandon Goodwin	1701593.0
2	Atlanta Hawks	Bruno Fernando	1517981.0
3	Atlanta Hawks	Cam Reddish	4458000.0
4	Atlanta Hawks	Clint Capela	16000000.0

Pull draft positions of each player and merge with active players

```
draft = pd.read_sql("""SELECT *
                       FROM Draft;""",conn)
draft = draft[['yearDraft','numberPickOverall','numberRound','namePlayer','idPlayer','nameTeam']]
draft = draft.rename(columns={'yearDraft':'Year_Drafted','numberPickOverall':'Overall_Pick','numberRound':'Round','namePlayer':'full_name','idPlayer':'id','nameTeam':'Drafted_by'})
active_info = pd.merge(active,draft, on='full_name')
active_info.head()
```

	id_x	full_name	Year_Drafted	Overall_Pick	Round	id_y	Drafted_by
0	203500	Steven Adams	2013.0	12.0	1.0	203500.0	Oklahoma City Thunder
1	1628389	Bam Adebayo	2017.0	14.0	1.0	1628389.0	Miami Heat
2	200746	LaMarcus Aldridge	2006.0	2.0	1.0	200746.0	Chicago Bulls
3	1629638	Nickeil Alexander-Walker	2019.0	17.0	1.0	1629638.0	Brooklyn Nets

PostgresSQL

- PostgreSQL database, “mybasketball”
- Used the sqlite data from Kaggle’s Basketball Dataset to create csv’s (https://www.kaggle.com/wyattowalsh/basketball)
- created tables related to data from csv files, once they were cleaned and scrapped PostgresSQL
- uploaded the csv files to the corresponding tables using SQLAlchemy

```
import pandas as pd
import psycopg2
```
[3] Python

```
conn = psycopg2.connect(
    database='postgres',user='postgres',password='Kabrija01',host='127.0.0.1',port= '5432'
)
conn.autocommit = True
```
[5] Python

```
cursor = conn.cursor()
sql = '''CREATE database mybasketball'''
cursor.execute(sql)
```
[6] Python

```
CREATE TABLE active_info (
    id INT,
    full_name VARCHAR UNIQUE NOT NULL PRIMARY KEY,
    year_drafted DECIMAL,
    overall DECIMAL,
    pick DECIMAL,
    round_ DECIMAL,
    id_y DECIMAL,
    drafted_by VARCHAR
);


CREATE TABLE salaries_21 (
    team VARCHAR,
    full_name VARCHAR,
    salary DECIMAL,
    FOREIGN KEY (full_name) REFERENCES active_info(full_name)

);
```