COMS W4111: Introduction to Databases
Section 002, Spring 2022

Extra Credit Assignment 1

In [27]:

```
%load_ext sql
```

```
The sql extension is already loaded. To reload it, use:
  %reload_ext sql
```

In [28]:

```
%sql mysql+pymysql://root:Xcz990208!@localhost
```

Out[28]:

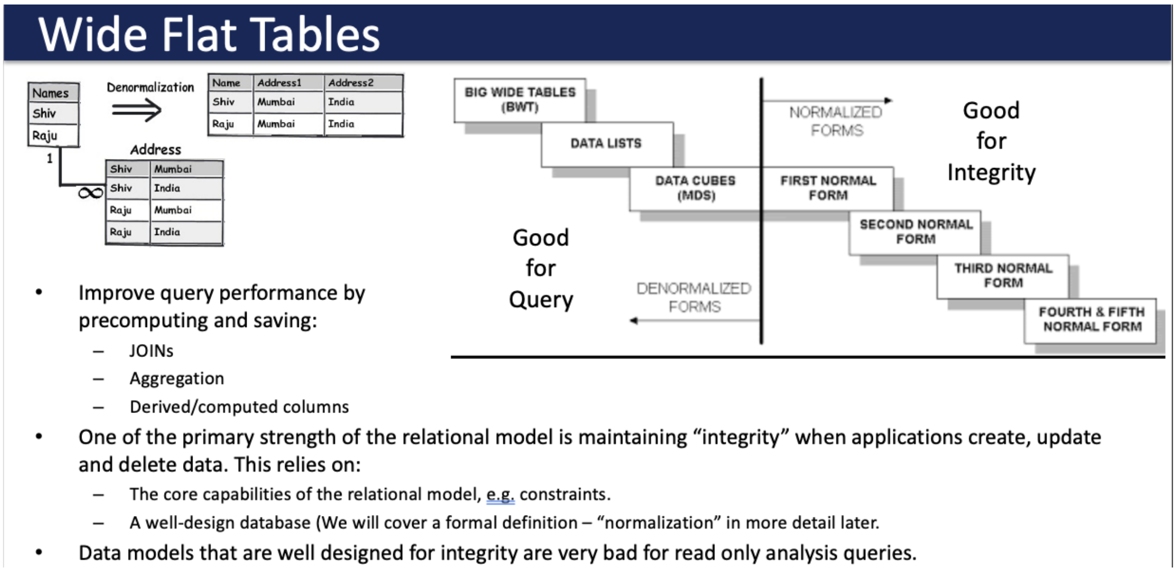
```
'Connected: root@None'
```

Overview

- This extra-credit assignment is relatively simple and work 2.5 points.
- We will create an assignment on Gradescope and a discussion topic on Ed.
- The submission format contains:
  1. A PDF copy of this notebook.
  2. A zip file containing the notebook and any additional files you used.
- Put the SQL statements you used to solve the problem in the identified section of the notebook and execute them. Finally, run the tests.
- You will use the Lahman's Baseball Database for this assignment. You must load the database into MySQL if you have not already done so.

Problem

- We have discussed guidelines for a "good" schema design.
- We will cover the concept of normalization in an upcoming lecture. Normalization tries for formalize and rigorously define rules for producing "a good schema."
- There is a tension in database schema design. No one approach is good for all application scenarios.



Wide Flat Tables

Names
Shiv
Raju

Denormalization

Name	Address1	Address2
Shiv	Mumbai	India
Raju	Mumbai	India

1

∞

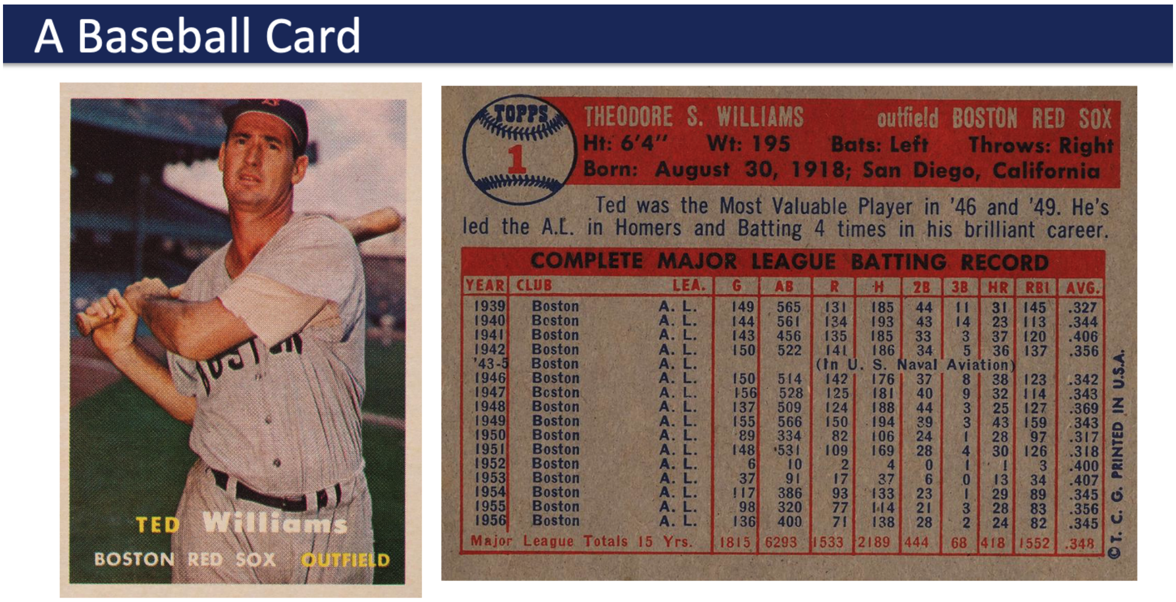
Address

Shiv	Mumbai
Shiv	India
Raju	Mumbai
Raju	India

- Improve query performance by precomputing and saving:
  - JOINs
  - Aggregation
  - Derived/computed columns
- One of the primary strength of the relational model is maintaining "integrity" when applications create, update and delete data. This relies on:
  - The core capabilities of the relational model, e.g. constraints.
  - A well-design database (We will cover a formal definition – "normalization" in more detail later.
- Data models that are well designed for integrity are very bad for read only analysis queries.

Schema Design Tradeoffs

- You will implement two approaches to provide the data necessary for an application that displays baseball-card like web pages.



A Baseball Card

TOPPS 1

THEODORE S. WILLIAMS outfield BOSTON RED SOX

Ht: 6'4" Wt: 195 Bats: Left Throws: Right

Born: August 30, 1918; San Diego, California

Ted was the Most Valuable Player in '46 and '49. He's led the A.L. in Homers and Batting 4 times in his brilliant career.

COMPLETE MAJOR LEAGUE BATTING RECORD

YEAR	CLUB	LEA.	G	AB	R	H	2B	3B	HR	RBI	AVG.
1939	Boston	A. L.	149	565	131	185	44	11	31	145	.327
1940	Boston	A. L.	144	561	134	193	43	14	23	113	.344
1941	Boston	A. L.	143	456	135	185	33	3	37	120	.406
1942	Boston	A. L.	150	522	141	186	34	5	36	137	.356
'43-5	Boston	A. L.			(In U. S. Naval Aviation)						
1946	Boston	A. L.	150	514	142	176	37	8	38	123	.342
1947	Boston	A. L.	156	528	125	181	40	9	32	114	.343
1948	Boston	A. L.	137	509	124	188	44	3	25	127	.369
1949	Boston	A. L.	155	566	150	194	39	3	43	159	.343
1950	Boston	A. L.	89	334	82	106	24	1	28	97	.317
1951	Boston	A. L.	148	531	109	169	28	4	30	126	.318
1952	Boston	A. L.	6	10	2	4	0	1	1	3	.400
1953	Boston	A. L.	37	91	17	37	6	0	13	34	.407
1954	Boston	A. L.	117	386	93	133	23	1	29	89	.345
1955	Boston	A. L.	98	320	77	114	21	3	28	83	.356
1956	Boston	A. L.	136	400	71	138	28	2	24	82	.345
Major League Totals 15 Yrs.			1815	6293	1533	2189	444	68	418	1552	.348

©T. C. G. PRINTED IN U.S.A.

Example Baseball Card

- We will focus on the information on the back of the card.

- You must implement a table of annual performance.

- The annual performance table use the following base tables from Lahman's DB:
  1. Appearances
  2. Batting
  3. Pitching
  4. Fielding
  5. AllStarFull

- The columns in the table are:
  - `playerID`

    - `yearID`
    - `teamID`
    - `G` : Sum of games played for the team in the year (from `G_all` from `Appearances`).
    - `AB` : Sum of `AB` for team in the year (from `Batting`).
    - `H` : Sum of `H` for team in the year (from `Batting`).
    - `R` : Sum of `R` (runs scored) for the team in the year (from `Batting`).
    - `RBI` : Sum of `RBI` for the team in the year (from `Batting`).
    - `GP` : Sum of games pitched for the team in the year. (`G` from `Pitching`).
    - `W` : Sum of wins for the team in the year. (`W` from `Pitching`).
    - `L` : Sum of losses for the team in the year. (`L` from `Pitching`).
    - `POS` : A comma separated list of the positions for the team and year (`POS` from `Fielding`).
    - `AS` : This will be either `Y` or `N` depending on whether or not the play is in the `AllStarFull` table for the team and year.

- My table for Ted Williams looks like ... ...

In [43]:

```
%%sql

select * from player_year_view where playerid='willite01';
```

```
 * mysql+pymysql://dbuser:***@localhost
19 rows affected.
```

Out[43]:

playerID	teamID	yearID	G	AB	H	R	RBI	PD	W	L	POS	AS
willite01	BOS	1939	149	565	185	131	145.0	-	-	-	OF	N
willite01	BOS	1940	144	561	193	134	113.0	1	0	0	OF,P	Y
willite01	BOS	1941	143	456	185	135	120.0	-	-	-	OF	Y
willite01	BOS	1942	150	522	186	141	137.0	-	-	-	OF	Y
willite01	BOS	1946	150	514	176	142	123.0	-	-	-	OF	Y
willite01	BOS	1947	156	528	181	125	114.0	-	-	-	OF	Y
willite01	BOS	1948	137	509	188	124	127.0	-	-	-	OF	Y
willite01	BOS	1949	155	566	194	150	159.0	-	-	-	OF	Y
willite01	BOS	1950	89	334	106	82	97.0	-	-	-	OF	Y
willite01	BOS	1951	148	531	169	109	126.0	-	-	-	OF	Y
willite01	BOS	1952	6	10	4	2	3.0	-	-	-	OF	N
willite01	BOS	1953	37	91	37	17	34.0	-	-	-	OF	Y
willite01	BOS	1954	117	386	133	93	89.0	-	-	-	OF	Y
willite01	BOS	1955	98	320	114	77	83.0	-	-	-	OF	Y
willite01	BOS	1956	136	400	138	71	82.0	-	-	-	OF	Y
willite01	BOS	1957	132	420	163	96	87.0	-	-	-	OF	Y
willite01	BOS	1958	129	411	135	81	85.0	-	-	-	OF	Y
willite01	BOS	1959	103	272	69	32	43.0	-	-	-	OF	Y
willite01	BOS	1960	113	310	98	56	72.0	-	-	-	OF	Y

Solution

Your SQL

- Show the SQL for creating the view and table with a copy of the data. The tests assume that you call the view and table:
  - `player_year_view`
  - `player_year_copy`

In [ ]:

```
# Load the data
```

In [3]:

```
database_user_id = "root"
database_pwd = "Xcz990208!"

database_url = "mysql+pymysql://" + \
    database_user_id + ":" + database_pwd + "@localhost"
database_url
```

Out[3]:

```
'mysql+pymysql://root:Xcz990208!@localhost'
```

In [4]:

```
from sqlalchemy import create_engine
```

In [5]:

```
sqla_engine = create_engine(database_url)
```

In [26]:

```
%sql drop schema if exists S22_W4111_mid_extra;
%sql create schema if not exists S22_W4111_mid_extra;

%sql drop table if exists S22_W4111_mid_extra.Appearances;
%sql drop table if exists S22_W4111_mid_extra.Batting;
%sql drop table if exists S22_W4111_mid_extra.Pitching;
%sql drop table if exists S22_W4111_mid_extra.Fielding;
%sql drop table if exists S22_W4111_mid_extra.AllStarFull;
```

```
 * mysql+pymysql://root:***@localhost
4 rows affected.
 * mysql+pymysql://root:***@localhost
1 rows affected.
 * mysql+pymysql://root:***@localhost
0 rows affected.
 * mysql+pymysql://root:***@localhost
0 rows affected.
 * mysql+pymysql://root:***@localhost
0 rows affected.
 * mysql+pymysql://root:***@localhost
0 rows affected.
 * mysql+pymysql://root:***@localhost
0 rows affected.
```

Out[26]:

```
[]
```

In [27]:

```
%%sql
create table if not exists S22_W4111_mid_extra.appearances
```

```
(
        yearID text null,
        teamID text null,
        lgID text null,
        playerID text null,
        G_all  float null,
        GS float null,
        G_batting float null,
        G_defense float null,
        G_p float null,
        G_c float null,
        G_1b float null,
        G_2b float null,
        G_3b float null,
        G_ss float null,
        G_lf float null,
        G_cf float null,
        G_rf float null,
        G_of float null,
        G_dh float null,
        G_ph float null,
        G_pr float null
);
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
```

Out[27]:

```
[]
```

In [28]:

```
%%sql
create table if not exists S22_W4111_mid_extra.batting
(
        playerID text null,
        yearID text null,
        stint text null,
        teamID text null,
        lgID text null,
        G float null,
        AB float null,
        R float null,
        H float null,
        2B float null,
        3B float null,
        HR float null,
        RBI float null,
        SB float null,
        CS float null,
        BB float null,
        SO float null,
        IBB float null,
        HBP float null,
        SH float null,
        SF float null,
        GIDP float null
);
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
```

Out[28]:

```
[]
```

In [29]:

```
%%sql
create table if not exists S22_W4111_mid_extra.pitching
```

```
(
        playerID text null,
        yearID text null,
        stint text null,
        teamID text null,
        lgID text null,
        W float null,
        L float null,
        G float null,
        GS float null,
        CG float null,
        SHO float null,
        SV float null,
        IPouts float null,
        H float null,
        ER float null,
        HR float null,
        BB float null,
        SO float null,
        BAOpp float null,
        ERA float null,
        IBB float null,
        WP float null,
        HBP float null,
        BK float null,
        BFP float null,
        GF float null,
        R float null,
        SH float null,
        SF float null,
        GIDP float null
);
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
```

Out[29]:

```
[]
```

In [30]:

```
%%sql
create table if not exists S22_W4111_mid_extra.fielding
(
        playerID text null,
        yearID text null,
        stint text null,
        teamID text null,
        lgID text null,
        POS text null,
        G float null,
        GS float null,
        InOuts float null,
        PO float null,
        A float null,
        E float null,
        DP float null,
        PB float null,
        WP float null,
        SB float null,
        CS float null,
        ZR float null
);
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
```

```
Out[30]: []
```

```
In [31]:
%%sql
create table if not exists S22_W4111_mid_extra.allstarfull
(
        playerID text null,
        yearID text null,
        gameNum float null,
        gameID text null,
        teamID text null,
        lgID text null,
        GP float null,
        startingPos float null
);
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
Out[31]: []
```

```
In [6]:
%sql SET GLOBAL local_infile = ’ON’;
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
Out[6]: []
```

```
In [7]:
import pymysql
con = pymysql.connect(host=”localhost”,
                        user=”root”,
                        password=”Xcz990208!”,
                        autocommit=True,
                        local_infile=1)
```

```
In [43]:
sql_appearances = ”””
LOAD DATA LOCAL INFILE
’C://Users//94822//Desktop//Intro_to_databases_4111//midterm-extra//Extra-Credit-1.ipy
INTO TABLE S22_W4111_mid_extra.appearances
    FIELDS TERMINATED BY ’,’
    ENCLOSED BY ’”’
    LINES TERMINATED BY ’\n’
    IGNORE 1 LINES;
”””

sql_batting = ”””
LOAD DATA LOCAL INFILE
’C://Users//94822//Desktop//Intro_to_databases_4111//midterm-extra//Extra-Credit-1.ipy
INTO TABLE S22_W4111_mid_extra.batting
    FIELDS TERMINATED BY ’,’
    ENCLOSED BY ’”’
    LINES TERMINATED BY ’\n’
    IGNORE 1 LINES;
”””

sql_pitching = ”””
LOAD DATA LOCAL INFILE
’C://Users//94822//Desktop//Intro_to_databases_4111//midterm-extra//Extra-Credit-1.ipy
INTO TABLE S22_W4111_mid_extra.pitching
    FIELDS TERMINATED BY ’,’
    ENCLOSED BY ’”’
    LINES TERMINATED BY ’\n’
    IGNORE 1 LINES;
"""

sql_fielding = """
LOAD DATA LOCAL INFILE
'C://Users//94822//Desktop//Intro_to_databases_4111//midterm-extra//Extra-Credit-1.ipy
INTO TABLE S22_W4111_mid_extra.fielding
    FIELDS TERMINATED BY ','
    ENCLOSED BY '"'
    LINES TERMINATED BY '\n'
    IGNORE 1 LINES;
"""

sql_allstarfull = """
LOAD DATA LOCAL INFILE
'C://Users//94822//Desktop//Intro_to_databases_4111//midterm-extra//Extra-Credit-1.ipy
INTO TABLE S22_W4111_mid_extra.allstarfull
    FIELDS TERMINATED BY ','
    ENCLOSED BY '"'
    LINES TERMINATED BY '\n'
    IGNORE 1 LINES;
"""
```

In [44]:

```
cur = con.cursor()
cur.execute(sql_appearances)
cur.execute(sql_batting)
cur.execute(sql_pitching)
cur.execute(sql_fielding)
cur.execute(sql_allstarfull)
cur.close()
```

In [36]:

```
# Tset if loading properly
```

In [45]:

```
%sql select * from S22_W4111_mid_extra.appearances limit 2;
```

```
 * mysql+pymysql://root:***@localhost
2 rows affected.
```

Out[45]:

yearID	teamID	lgID	playerID	G_all	GS	G_batting	G_defense	G_p	G_c	G_1b	G_2b	G_3b
1871	TRO	NA	abercda01	1.0	1.0	1.0	1.0	0.0	0.0	0.0	0.0	0.0
1871	RC1	NA	addybo01	25.0	25.0	25.0	25.0	0.0	0.0	0.0	22.0	0.0

In [46]:

```
%sql select * from S22_W4111_mid_extra.batting limit 2;
```

```
 * mysql+pymysql://root:***@localhost
2 rows affected.
```

Out[46]:

playerID	yearID	stint	teamID	lgID	G	AB	R	H	2B	3B	HR	RBI	SB	CS	BB	S
abercda01	1871	1	TRO	NA	1.0	4.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.
addybo01	1871	1	RC1	NA	25.0	118.0	30.0	32.0	6.0	0.0	0.0	13.0	8.0	1.0	4.0	0.

In [47]:

```
%sql select * from S22_W4111_mid_extra.pitching limit 2;
```

```
 * mysql+pymysql://root:***@localhost
2 rows affected.
```

Out[47]:

playerID	yearID	stint	teamID	lgID	W	L	G	GS	CG	SHO	SV	IPouts	H	ER
bechtge01	1871	1	PH1	NA	1.0	2.0	3.0	3.0	2.0	0.0	0.0	78.0	43.0	23.0
brainas01	1871	1	WS3	NA	12.0	15.0	30.0	30.0	30.0	0.0	0.0	792.0	361.0	132.0

In [48]:

```
%sql select * from S22_W4111_mid_extra.fielding limit 2;
```

```
 * mysql+pymysql://root:***@localhost
2 rows affected.
```

Out[48]:

playerID	yearID	stint	teamID	lgID	POS	G	GS	InOuts	PO	A	E	DP	PB	WP	Sl
abercda01	1871	1	TRO	NA	SS	1.0	1.0	24.0	1.0	3.0	2.0	0.0	0.0	0.0	0.
addybo01	1871	1	RC1	NA	2B	22.0	22.0	606.0	67.0	72.0	42.0	5.0	0.0	0.0	0.

In [49]:

```
%sql select * from S22_W4111_mid_extra.allstarfull limit 2;
```

```
 * mysql+pymysql://root:***@localhost
2 rows affected.
```

Out[49]:

playerID	yearID	gameNum	gameID	teamID	lgID	GP	startingPos
gomezle01	1933	0.0	ALS193307060	NYA	AL	1.0	1.0
ferreri01	1933	0.0	ALS193307060	BOS	AL	1.0	2.0

In [ ]:

```
# Create the view
```

In [32]:

```
%%sql
use S22_W4111_mid_extra;
drop view if exists player_year_view;
create view player_year_view as
with appearances_table as
    (
        select
            playerID,
            yearID,
            teamID,
            sum(G_all) as G
                from appearances
                group by playerID, yearID
    ),

    batting_table as
    (
        select
            playerID,
            yearID,
            sum(AB) as AB,
            sum(H) as H,
            sum(R) as R,
            sum(RBI) as RBI
                from batting
                    group by playerID, yearID
    ),

    app_bat as
    (
        select * from appearances_table join batting_table using (playerID, yearID)
    ),

    pitching_table as
    (
        select
            playerID,
            yearID,
            sum(G) as GP,
            sum(W) as W,
            sum(L) as L
                from pitching
                    group by playerID, yearID
    ),
    app_bat_pit as
    (
        select * from app_bat left join pitching_table using (playerID, yearID)
    ),

    fielding_table as
    (
        select
        playerID,
        yearID,
        group_concat(POS) as POS
            from fielding
                group by playerID, yearID
    ),

    app_bat_pit_fie as
    (
        select * from app_bat_pit join fielding_table using (playerID, yearID)
    ),

    allstar_table as
    (
        select
            playerID,
            yearID,
            yearID as pre
                from allstarfull
                    group by playerID, yearID
    ),

    app_bat_pit_fie_all as
    (
        select
            playerID,
            yearID,
            teamID,
            G,
            AB,
            H,
            R,
            RBI,
            GP,
            W,
            L,
            POS,
            pre as A
```

```
                from app_bat_pit_fie left join allstar_table using (playerID, yearI
    ),
    baseball_card as
    (
        select
            playerID,
            yearID,
            teamID,
            G,
            AB,
            H,
            R,
            RBI,
            GP,
            W,
            L,
            POS,
            (CASE WHEN A > 1900 and A < 2023 THEN 'Y' ELSE 'N' END) `AS`
                from app_bat_pit_fie_all
    )
select * from baseball_card;
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
0 rows affected.
0 rows affected.
```

Out[32]:

```
[]
```

In [ ]:

```
# Create the copy
```

In [37]:

```
%%sql
drop table if exists S22_W4111_mid_extra.play_year_copy;
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
```

Out[37]:

```
[]
```

In [40]:

```
%%sql
create table if not exists S22_W4111_mid_extra.play_year_copy
(
        playerID text null,
        yearID text null,
        teamID text null,
        G float null,
        AB float null,
        H float null,
        R float null,
        RBI float null,
        GP float null,
        W float null,
        L float null,
        POS text null,
        `AS` text null
);
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
```

Out[40]:

```
[]
```

In [41]:

```
%%sql
insert into play_year_copy
   select * from player_year_view
```

```
 * mysql+pymysql://root:***@localhost
101002 rows affected.
```

Out[41]:

```
[]
```

Tests

- Use the code and section below to run tests.
- The following function tests performance.
- Your answer should match the results, but the times will be different.

In [42]:

```
import pymysql
```

In [43]:

```
conn = pymysql.connect(
    host="localhost",
    user="root",
    password="Xcz990208!",
    cursorclass=pymysql.cursors.DictCursor,
    db="S22_W4111_mid_extra",
    autocommit=True)
```

In [44]:

```
import time
```

In [48]:

```
import pandas as pd
```

In [49]:

```
def get_data_performance(table_name, playerID, iterations):
    """
    :param table_name: The name of the table to use for the query.
    :param playerID: The ID of the player whose data to get.
    :param iterations: How many times to run the query.
    """
    sql = """
        select * from {table} where playerID='{playerID}'
    """

    sql = sql.format(table=table_name, playerID=playerID)
    # print(sql)

    cur = conn.cursor()

    start_time = time.time()

    for i in range(0, iterations):
        res = cur.execute(sql)

    res = cur.fetchall()
    res = pd.DataFrame(res)

    end_time = time.time()
    elapsed_time = (end_time-start_time)
    return elapsed_time, res
```

- Perform the test on the view

In [50]:
```
elapsed_time, result = get_data_performance(’player_year_view’,
                                            ’willite01’, 10)
```

In [51]:
```
elapsed_time
```

Out[51]:
```
11.102262496948242
```

In [52]:
```
result
```

Out[52]:

	playerID	yearID	teamID	G	AB	H	R	RBI	GP	W	L	POS	AS
0	willite01	1940	BOS	144.0	561.0	193.0	134.0	113.0	1.0	0.0	0.0	P,OF	Y
1	willite01	1941	BOS	143.0	456.0	185.0	135.0	120.0	NaN	NaN	NaN	OF	Y
2	willite01	1942	BOS	150.0	522.0	186.0	141.0	137.0	NaN	NaN	NaN	OF	Y
3	willite01	1946	BOS	150.0	514.0	176.0	142.0	123.0	NaN	NaN	NaN	OF	Y
4	willite01	1947	BOS	156.0	528.0	181.0	125.0	114.0	NaN	NaN	NaN	OF	Y
5	willite01	1948	BOS	137.0	509.0	188.0	124.0	127.0	NaN	NaN	NaN	OF	Y
6	willite01	1949	BOS	155.0	566.0	194.0	150.0	159.0	NaN	NaN	NaN	OF	Y
7	willite01	1950	BOS	89.0	334.0	106.0	82.0	97.0	NaN	NaN	NaN	OF	Y
8	willite01	1951	BOS	148.0	531.0	169.0	109.0	126.0	NaN	NaN	NaN	OF	Y
9	willite01	1953	BOS	37.0	91.0	37.0	17.0	34.0	NaN	NaN	NaN	OF	Y
10	willite01	1954	BOS	117.0	386.0	133.0	93.0	89.0	NaN	NaN	NaN	OF	Y
11	willite01	1955	BOS	98.0	320.0	114.0	77.0	83.0	NaN	NaN	NaN	OF	Y
12	willite01	1956	BOS	136.0	400.0	138.0	71.0	82.0	NaN	NaN	NaN	OF	Y
13	willite01	1957	BOS	132.0	420.0	163.0	96.0	87.0	NaN	NaN	NaN	OF	Y
14	willite01	1958	BOS	129.0	411.0	135.0	81.0	85.0	NaN	NaN	NaN	OF	Y
15	willite01	1959	BOS	103.0	272.0	69.0	32.0	43.0	NaN	NaN	NaN	OF	Y
16	willite01	1960	BOS	113.0	310.0	98.0	56.0	72.0	NaN	NaN	NaN	OF	Y
17	willite01	1939	BOS	149.0	565.0	185.0	131.0	145.0	NaN	NaN	NaN	OF	N
18	willite01	1952	BOS	6.0	10.0	4.0	2.0	3.0	NaN	NaN	NaN	OF	N

- Now use the copy.

In [54]:
```
elapsed_time, result = get_data_performance(’play_year_copy’,
                                            ’willite01’, 10)
```

In [55]:

```
elapsed_time
```

Out[55]:

```
1.042792797088623
```

In [56]:

```
result
```

Out[56]:

	playerID	yearID	teamID	G	AB	H	R	RBI	GP	W	L	POS	AS
0	willite01	1940	BOS	144.0	561.0	193.0	134.0	113.0	1.0	0.0	0.0	P,OF	Y
1	willite01	1941	BOS	143.0	456.0	185.0	135.0	120.0	NaN	NaN	NaN	OF	Y
2	willite01	1942	BOS	150.0	522.0	186.0	141.0	137.0	NaN	NaN	NaN	OF	Y
3	willite01	1946	BOS	150.0	514.0	176.0	142.0	123.0	NaN	NaN	NaN	OF	Y
4	willite01	1947	BOS	156.0	528.0	181.0	125.0	114.0	NaN	NaN	NaN	OF	Y
5	willite01	1948	BOS	137.0	509.0	188.0	124.0	127.0	NaN	NaN	NaN	OF	Y
6	willite01	1949	BOS	155.0	566.0	194.0	150.0	159.0	NaN	NaN	NaN	OF	Y
7	willite01	1950	BOS	89.0	334.0	106.0	82.0	97.0	NaN	NaN	NaN	OF	Y
8	willite01	1951	BOS	148.0	531.0	169.0	109.0	126.0	NaN	NaN	NaN	OF	Y
9	willite01	1953	BOS	37.0	91.0	37.0	17.0	34.0	NaN	NaN	NaN	OF	Y
10	willite01	1954	BOS	117.0	386.0	133.0	93.0	89.0	NaN	NaN	NaN	OF	Y
11	willite01	1955	BOS	98.0	320.0	114.0	77.0	83.0	NaN	NaN	NaN	OF	Y
12	willite01	1956	BOS	136.0	400.0	138.0	71.0	82.0	NaN	NaN	NaN	OF	Y
13	willite01	1957	BOS	132.0	420.0	163.0	96.0	87.0	NaN	NaN	NaN	OF	Y
14	willite01	1958	BOS	129.0	411.0	135.0	81.0	85.0	NaN	NaN	NaN	OF	Y
15	willite01	1959	BOS	103.0	272.0	69.0	32.0	43.0	NaN	NaN	NaN	OF	Y
16	willite01	1960	BOS	113.0	310.0	98.0	56.0	72.0	NaN	NaN	NaN	OF	Y
17	willite01	1939	BOS	149.0	565.0	185.0	131.0	145.0	NaN	NaN	NaN	OF	N
18	willite01	1952	BOS	6.0	10.0	4.0	2.0	3.0	NaN	NaN	NaN	OF	N

In [ ]: