ESTUDIO ESTADÍSTICO DE LA NBA (TEMPORADA 2018 - 2019)

```
import pandas as pd
import sqlite3 as sqlite
import numpy as np
from matplotlib import pyplot as plt
```

CREAR CONEXIÓN A LA BASE DE DATOS (nba_data.db)

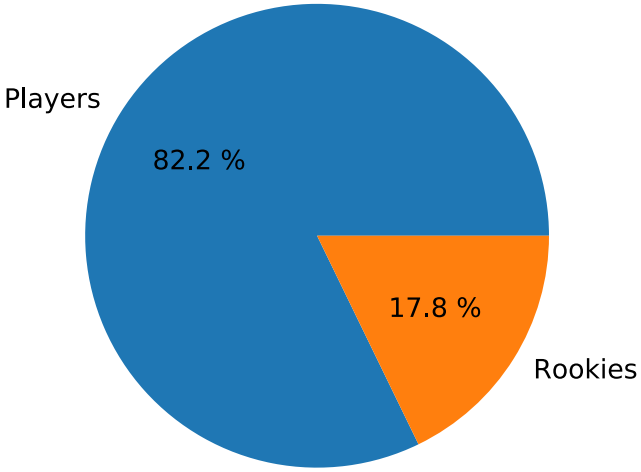
```
sql_data = 'nba_data.db'
connection = sqlite.connect(sql_data)
```

EJEMPLOS DE CONSULTAS:

3. Gráfico porcentaje Jugadores y Rookies de la temporada

```
query = "SELECT * FROM STAT_PT"
playersConf_df = pd.read_sql(query, connection)
noRookies_per = playersConf_df["NoRookies_Perc"].sum()
rookies_per = playersConf_df["Rookies_Perc"].sum()
values = [noRookies_per, rookies_per]
names = ["Players", "Rookies"]

plt.pie(values, labels=names, autopct="%0.1f %%")
plt.show()
```



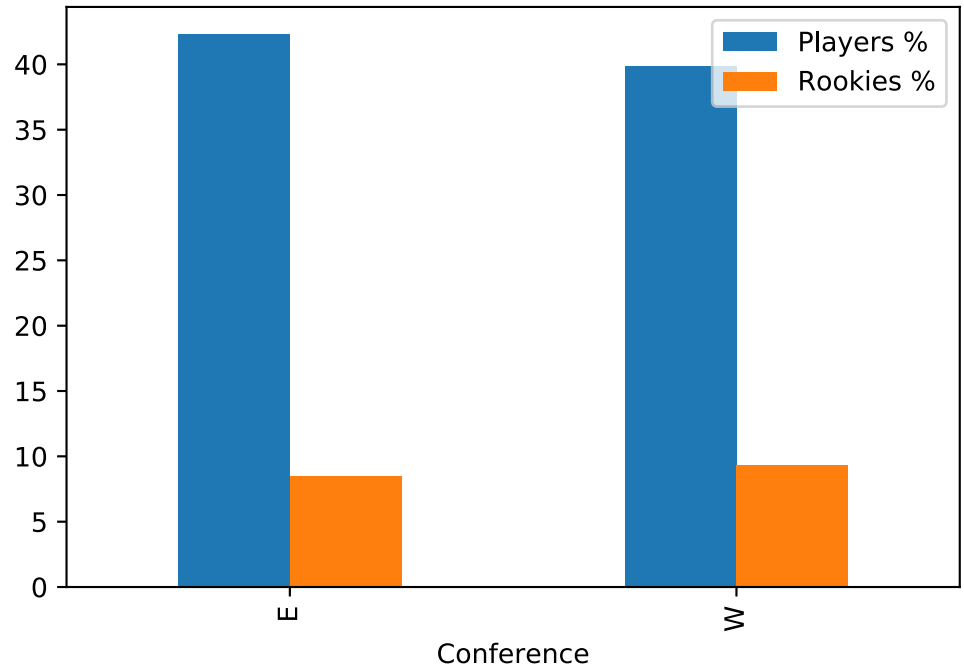
2. Número de jugadores (rookies y no rookies) porcentaje por cada conferencias

```
query = 'SELECT Conference, SUM(NoRookies_Perc) AS "Players %",
SUM(Rookies_Perc) AS "Rookies %" FROM STAT_PT GROUP BY Conference;'
playersConf_df = pd.read_sql(query, connection)
playersConf_df
```

	Conference	Players %	Rookies %
0	E	42.28	8.49
1	W	39.84	9.29

```
# Diagrama de barras
playersConf_df.plot(kind="bar", x="Conference")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x10eda6250>
```



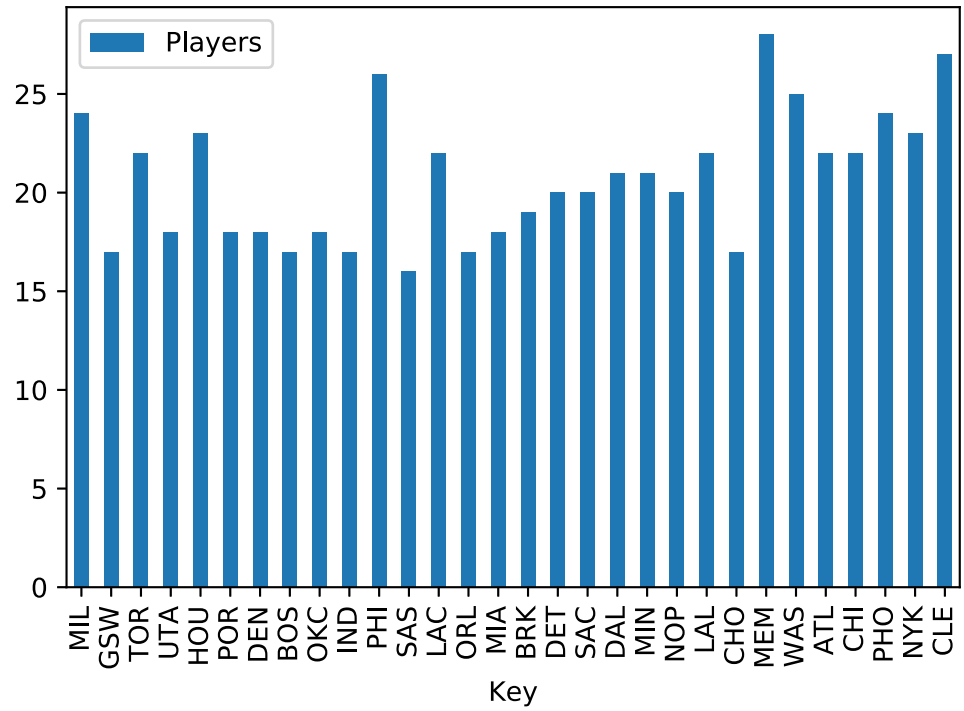
3. Número de jugadores que forman cada equipo, junto con el porcentaje

```
query = 'SELECT * FROM NUM_PT'
playersTeam_df = pd.read_sql(query, connection)
players_df = playersTeam_df.drop(columns=["Players_Perc"])
playersTeam_df.head()
```

	Id	Key	Team	Conference	Players	Players_Perc
0	1	MIL	Milwaukee Bucks	E	24	3.86
1	2	GSW	Golden State Warriors	W	17	2.73
2	3	TOR	Toronto Raptors	E	22	3.54
3	4	UTA	Utah Jazz	W	18	2.89
4	5	HOU	Houston Rockets	W	23	3.70

```
players_df.plot(kind="bar", x="Key", y="Players")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x10efc49d0>
```



4. Simulación de la clasificación final de la temporada (basandonos en el porcentaje de victorias / derrotas)

```
#query = 'SELECT id, Key, Team,  W AS Wins, L AS Losses FROM TEAMS ORDER
BY W/L DESC;'
query = 'SELECT Key, Team, Conf, W, L FROM TEAMS ORDER BY W/L DESC;'
teamsQualy_df = pd.read_sql(query, connection)
teamsQualy_df
```

	Key	Team	Conf	W	L
0	MIL	Milwaukee Bucks	E	60.0	22.0
1	TOR	Toronto Raptors	E	58.0	24.0
2	GSW	Golden State Warriors	W	57.0	25.0
3	DEN	Denver Nuggets	W	54.0	28.0
4	HOU	Houston Rockets	W	53.0	29.0
5	POR	Portland Trail Blazers	W	53.0	29.0
6	PHI	Philadelphia 76ers	E	51.0	31.0
7	UTA	Utah Jazz	W	50.0	32.0
8	BOS	Boston Celtics	E	49.0	33.0
9	OKC	Oklahoma City Thunder	W	49.0	33.0
10	IND	Indiana Pacers	E	48.0	34.0

	Key	Team	Conf	W	L
11	SAS	San Antonio Spurs	W	48.0	34.0
12	LAC	Los Angeles Clippers	W	48.0	34.0
13	ORL	Orlando Magic	E	42.0	40.0
14	BRK	Brooklyn Nets	E	42.0	40.0
15	DET	Detroit Pistons	E	41.0	41.0
16	MIA	Miami Heat	E	39.0	43.0
17	SAC	Sacramento Kings	W	39.0	43.0
18	CHO	Charlotte Hornets	E	39.0	43.0
19	LAL	Los Angeles Lakers	W	37.0	45.0
20	MIN	Minnesota Timberwolves	W	36.0	46.0
21	DAL	Dallas Mavericks	W	33.0	49.0
22	NOP	New Orleans Pelicans	W	33.0	49.0
23	MEM	Memphis Grizzlies	W	33.0	49.0
24	WAS	Washington Wizards	E	32.0	50.0
25	ATL	Atlanta Hawks	E	29.0	53.0
26	CHI	Chicago Bulls	E	22.0	60.0
27	PHO	Phoenix Suns	W	19.0	63.0
28	CLE	Cleveland Cavaliers	E	19.0	63.0
29	NYK	New York Knicks	E	17.0	65.0

4. Primer equipo de cada conferencia

```
#query = 'SELECT Conf AS Conference, ID, Key, Team, W AS Wins, L AS Losses
FROM TEAMS GROUP BY CONF HAVING MAX(W/L);'
query = 'SELECT * FROM TEAMS GROUP BY CONF HAVING MAX(W/L);'
bestTeamConf_df = pd.read_sql(query, connection)
bestTeamConf_df.drop(columns=["id"])
```

	Key	Team	Conf	W	L	W/L%	MOV	ORtg	DRtg	NRtg
0	MIL	Milwaukee Bucks	E	60.0	22.0	0.732	8.87	114.23	105.76	8.47
1	GSW	Golden State Warriors	W	57.0	25.0	0.695	6.46	116.63	110.24	6.39

5. Todos los jugadores

```
query = 'SELECT Player, FGA, FTA, PTS FROM PLAYERS;'
players_df = pd.read_sql(query, connection)
players_df
```

	Player	FGA	FTA	PTS
0	Álex Abrines	157.0	13.0	165.0
1	Quincy Acy	18.0	10.0	17.0
2	Jaylen Adams	110.0	9.0	108.0
3	Steven Adams	809.0	292.0	1108.0
4	Bam Adebayo	486.0	226.0	729.0
...	...	...	...	...
703	Tyler Zeller	28.0	18.0	46.0
704	Ante Žižić	331.0	132.0	459.0
705	Ivica Zubac	379.0	126.0	525.0
706	Ivica Zubac	193.0	66.0	281.0
707	Ivica Zubac	186.0	60.0	244.0

708 rows × 4 columns

Correlación

```
players_df.corr()
```

	FGA	FTA	PTS	TA	PTS_pred
FGA	1.000000	0.883147	0.990928	0.993240	0.993240
FTA	0.883147	1.000000	0.922593	0.931628	0.931628
PTS	0.990928	0.922593	1.000000	0.995973	0.995973
TA	0.993240	0.931628	0.995973	1.000000	1.000000
PTS_pred	0.993240	0.931628	0.995973	1.000000	1.000000

Modelo de regresión lineal

Con los datos obtenidos de nuestra base de datos almacenados en el dataframe **players_df** se buscará predecir cuanto anotará cada jugador, creando un modelo de regresión lineal basado en los intentos de tiro de la temporada almacenada en la base de datos

Creacion de la columna con todos los intentos de tiro (tiros de campo + tiros libres)

```
players_df['TA'] = players_df['FGA']+players_df['FTA']
```

```
import statsmodels.formula.api as smf
```

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)

<ipython-input-46-f897a2d817de> in <module>
----> 1 import statsmodels.formula.api as smf


ModuleNotFoundError: No module named 'statsmodels'
```

```
lm = smf.ols(formula='PTS~TA', data=players_df).fit()
```

```
lm.summary()
```

OLS Regression Results

Dep. Variable:	PTS	**R-squared:**	0.992
Model:	OLS	**Adj. R-squared:**	0.992
Method:	Least Squares	**F-statistic:**	8.712e+04
Date:	Tue, 25 Feb 2020	**Prob (F-statistic):**	0.00
Time:	22:32:21	**Log-Likelihood:**	-3631.3
No. Observations:	708	**AIC:**	7267.
Df Residuals:	706	**BIC:**	7276.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
Intercept	-5.2094	2.157	-2.415	0.016	-9.444	-0.975
TA	1.0001	0.003	295.159	0.000	0.993	1.007

Omnibus:	107.277	**Durbin-Watson:**	1.898

Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	1253.854
Skew:	0.175	**Prob(JB):**	5.36e-273
Kurtosis:	9.510	**Cond. No.**	893.

Warnings:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

```
lm.params
```

```
Intercept   -5.209429
TA           1.000137
dtype: float64
```

```
#pts_pred = lm.predict(pd.DataFrame(players_df['TA']))
players_df['PTS_pred'] = lm.params[0] + lm.params[1] * players_df['TA']
```

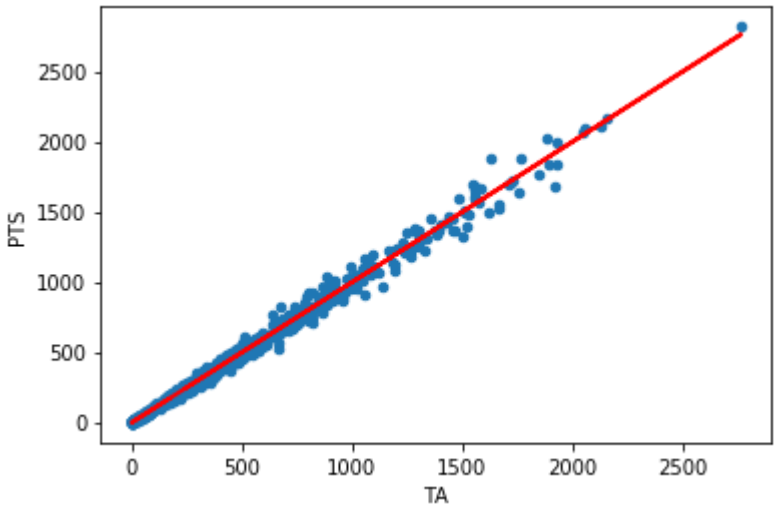
Formula de modelo: PTS = -5.209429 + 1.000137 * TA

```
SSD = sum((players_df["PTS"]-players_df['PTS_pred'])**2)
RSE = np.sqrt(SSD/len(players_df)-1)
RSE/np.mean(players_df['PTS'])
```

```
0.09255001071121378
```

```
%matplotlib inline
players_df.plot(kind = "scatter", x = "TA", y ="PTS")
plt.plot(pd.DataFrame(players_df["TA"]), players_df['PTS_pred'], c="red",
linewidth = 2)
```

```
[<matplotlib.lines.Line2D at 0x121c94750>]
```




```
players_df
```

	Player	FGA	FTA	PTS	TA	PTS_pred
0	Álex Abrines	157.0	13.0	165.0	170.0	164.813847
1	Quincy Acy	18.0	10.0	17.0	28.0	22.794404
2	Jaylen Adams	110.0	9.0	108.0	119.0	113.806864
3	Steven Adams	809.0	292.0	1108.0	1101.0	1095.941316
4	Bam Adebayo	486.0	226.0	729.0	712.0	706.888055
...	...	...	...	...	...	...
703	Tyler Zeller	28.0	18.0	46.0	46.0	40.796869
704	Ante Žižić	331.0	132.0	459.0	463.0	457.853963
705	Ivica Zubac	379.0	126.0	525.0	505.0	499.859714
706	Ivica Zubac	193.0	66.0	281.0	259.0	253.826032
707	Ivica Zubac	186.0	60.0	244.0	246.0	240.824252

708 rows × 6 columns