

## SF & LA Restaurant Inspection Scores

First, we had to decide on what dataset we wanted to use and agreed to do our project on restaurant inspections in San Francisco and Los Angeles. We found our datasets on the Kaggle website.

(<https://www.kaggle.com/san-francisco/sf-restaurant-scores-lives-standard> and <https://www.kaggle.com/cityofLA/la-restaurant-market-health-data> )

From there, we extracted the CSV files and used Jupyter Notebook to organize our data by importing pandas, combining both datasets and storing them into a dataframe.

```
In [1]: import pandas as pd
        from sqlalchemy import create_engine
```

### Store CSV into DataFrame

```
In [2]: csv_file = "Resources/LA.csv"
        LA_df = pd.read_csv(csv_file)
        LA_df.head()
```

```
Out[2]:
```

	serial_number	activity_date	facility_name	score	grade	service_code	service_description	employee_id	facility_address	facility_city	facility_id	facility_star
0	DAJ00E07B	2017-12-29T00:00:00	HABITAT COFFEE SHOP	95	A	1	ROUTINE INSPECTION	EE0000923	3708 N EAGLE ROCK BLVD	LOS ANGELES	FA0170465	C
1	DAQOKRFZB	2017-12-29T00:00:00	REILLY'S	92	A	1	ROUTINE INSPECTION	EE0000633	100 WORLD WAY # 120	LOS ANGELES	FA0244690	C
2	DASJI4LUR	2017-12-29T00:00:00	STREET CHURROS	93	A	1	ROUTINE INSPECTION	EE0000835	6801 HOLLYWOOD BLVD # 253	LOS ANGELES	FA0224109	C
3	DAWVA0CY3	2017-12-29T00:00:00	RIO GENTLEMANS CLUB	93	A	1	ROUTINE INSPECTION	EE0000958	13124 S FIGUEROA ST	LOS ANGELES	FA0046462	C
4	DAKFCHD0L	2017-12-29T00:00:00	LE PAIN QUOTIDIEN	93	A	1	ROUTINE INSPECTION	EE0000629	13050 SAN VICENTE BLVD STE 114	LOS ANGELES	FA0034788	C

Once we created a dataframe, we created a new dataframe from selecting certain columns we felt were important to keep. The columns we chose are as follows:

- Business
- City
- State
- Score
- Date

	business	city	state	score	date
0	The Clift Hotel (Spanish Suite)	San Francisco	CA	100.0	2016-02-16T00:00:00
1	KABABAYAN FAST FOOD	San Francisco	CA	83.0	2016-02-16T00:00:00
2	Extreme Pizza	San Francisco	CA	96.0	2016-02-16T00:00:00
3	Hahn's Hibachi	San Francisco	CA	75.0	2016-02-16T00:00:00
4	Miller's East Coast Deli	San Francisco	CA	75.0	2016-02-16T00:00:00
5	Emporor's Kitchen	San Francisco	CA	NaN	2016-02-16T00:00:00
6	KABABAYAN FAST FOOD	San Francisco	CA	83.0	2016-02-16T00:00:00
7	KABABAYAN FAST FOOD	San Francisco	CA	83.0	2016-02-16T00:00:00
8	China Express Deli	San Francisco	CA	86.0	2016-02-16T00:00:00
9	Miller's East Coast Deli	San Francisco	CA	75.0	2016-02-16T00:00:00

```
combined_df.isna().sum()
```

```
business      0
city          0
state         0
score      13935
date          0
dtype: int64
```

```
combined_df=combined_df.dropna(how='any')
```

```
combined_df['score'].count()
```

```
98322
```

```
combined_df.isna().sum()
```

```
business      0
city          0
state         0
score         0
date          0
dtype: int64
```

```
combined_df['date']=pd.to_datetime(combined_df['date'])
```

While transforming our data, we had to change column names to keep them uniform. We also ran into the issue of how to drop NaN's in the Scores column. Because the rest of the rows contained data, we could not just use `.dropna(how='all')`. Instead, we had to use `.dropna(how='any')`. To make sure there were no more Nan's in our dataset, we used the `isna()` method.

We then chose to load our data into the relational database, MySQL. Because of the fact that the type of data we have frequently gets updated, a relational database makes sense - it's simple and flexible with organizing data.

We created a MySQL schema to upload our transformed data and proceeded to connect to the local database we created in MySQL called "combined\_df." The final table we chose to use for the production database is called "sfla."

We did have difficulty pushing our CSV file to the MySQL table. We had to test the connection to see if it would pull up to the table, but we kept running into not only syntax errors, but operational errors as well.

```
1 • USE combined_db;
2
3 • CREATE TABLE sfla (
4     business_name TEXT
5     city_name TEXT,
6     state_name TEXT,
7     score INT,
8     date DATETIME
9
10 );
11
12 • SELECT * FROM sfla;
```

```
1 CREATE DATABASE combined_db;
2
3
4 • USE combined_db;
5
6 • CREATE TABLE sfla (
7     business TEXT,
8     city TEXT,
9     state TEXT,
10    score INT,
11    date DATETIME
12 );
13
14
15 • SELECT * FROM sfla;
```

Due to the fact that MySQL defaults to encoding latin-1 we ran into character specific issues from our data. We resolved the error by adding "utf-8" to our connection engine code and were able to complete the project.

Once the data was imported we ran a SQL query to check that all the data was imported successfully.