Obtaining Our Data - Lab

Introduction

In this lab you'll practice your munging and transforming skills in order to load in your data to solve a regression problem.

Objectives

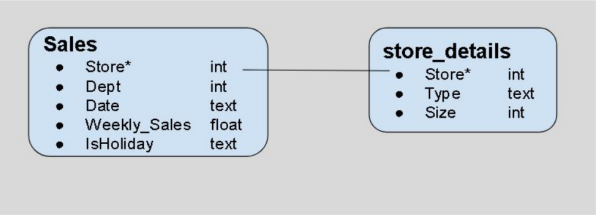
You will be able to:

- Understand the ETL process and the steps it consists of
- Understand the challenges of working with data from multiple sources

Task Description

Your boss gives you a general description of some of the datasets at your disposal for analyzing weekly store sales. They're eventually looking for you to build a model to help determine what factors impact sales, and model future sales forecasting for business planning.

Most of the properietary store data sits in the company sql database, accessible by all managers and above. The database is called **Walmart.db** Your boss provides you with the following basic schema:



She then tells you that she's put together a second dataset on general economy statistics for the various dates that she would also like you to incorporate in your analysis. That data, she says, is stored in a file **economy_data.csv**.

As a first step in creating your model for providing recommendations and projections, load and synthesize these disperate datasets into a singular unified DataFrame. Then save your results to a file **Merged_Store_Data.csv**.

Make sure you check the various data types and merge appropriately.

In [1]:
```
import sqlite3
import pandas as pd
```

In [7]:
```
con = sqlite3.connect('Walmart.db')
cur = con.cursor()
cur.execute("""select * from sales join store_details using(store);""")
df1 = pd.DataFrame(cur.fetchall())
df1.columns = [i[0] for i in cur.description]
print(df1.shape)
df1.head()
```

```
(452192, 7)
```

Out[7]:

	Store	Dept	Date	Weekly_Sales	IsHoliday	Type	Size
0	1	1	2010-02-05	24924.50	False	A	151315
1	1	1	2010-02-12	46039.49	True	A	151315
2	1	1	2010-02-19	41595.55	False	A	151315
3	1	1	2010-02-26	19403.54	False	A	151315
4	1	1	2010-03-05	21827.90	False	A	151315

In [8]:
```
df2 = pd.read_csv('economy_data.csv')
print(df2.shape)
df2.head()
```

```
(8190, 12)
```

Out[8]:

	Store	Date	Temperature	Fuel_Price	MarkDown1	MarkDown2	MarkDown3	MarkDown4	MarkDown5	CPI	Unemployment	IsHoliday
0	1	2010-02-05	42.31	2.572	NaN	NaN	NaN	NaN	NaN	211.096358	8.106	False
1	1	2010-02-12	38.51	2.548	NaN	NaN	NaN	NaN	NaN	211.242170	8.106	True
2	1	2010-02-19	39.93	2.514	NaN	NaN	NaN	NaN	NaN	211.289143	8.106	False
3	1	2010-02-26	46.63	2.561	NaN	NaN	NaN	NaN	NaN	211.319643	8.106	False
4	1	2010-03-05	46.50	2.625	NaN	NaN	NaN	NaN	NaN	211.350143	8.106	False

In [11]:
```
#Naive Merge is Faulty
```

```
merged = pd.merge(df1, df2)
print(merged.shape)
merged.head()
```

```
(0, 16)
```

Out[11]:

Store	Dept	Date	Weekly_Sales	IsHoliday	Type	Size	Temperature	Fuel_Price	MarkDown1	MarkDown2	MarkDown3	MarkDown4	MarkDown5	CPI

In [12]:

```
#Investigating
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 452192 entries, 0 to 452191
Data columns (total 7 columns):
Store           452192 non-null int64
Dept            452192 non-null int64
Date            452192 non-null object
Weekly_Sales    452192 non-null float64
IsHoliday       452192 non-null object
Type            452192 non-null object
Size            452192 non-null int64
dtypes: float64(1), int64(3), object(3)
memory usage: 24.1+ MB
```

In [13]:

```
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8190 entries, 0 to 8189
Data columns (total 12 columns):
Store           8190 non-null int64
Date            8190 non-null object
Temperature     8190 non-null float64
Fuel_Price      8190 non-null float64
MarkDown1       4032 non-null float64
MarkDown2       2921 non-null float64
MarkDown3       3613 non-null float64
MarkDown4       3464 non-null float64
MarkDown5       4050 non-null float64
CPI             7605 non-null float64
Unemployment    7605 non-null float64
IsHoliday       8190 non-null bool
dtypes: bool(1), float64(9), int64(1), object(1)
memory usage: 711.9+ KB
```

In [19]:

```
common = [col for col in df1.columns if col in df2.columns]
common
```

Out[19]:

```
['Store', 'Date', 'IsHoliday']
```

In [20]:

```
for col in common:
    ex1 = df1[col].iloc[0]
    ex2 = df2[col].iloc[0]
    print(col)
    print('Types:')
    print('df1: {}, df2: {}'.format(type(ex1), type(ex2)))
    print('\n')
```

```
Store
Types:
df1: <class 'numpy.int64'>, df2: <class 'numpy.int64'>


Date
Types:
df1: <class 'str'>, df2: <class 'str'>


IsHoliday
Types:
df1: <class 'str'>, df2: <class 'numpy.bool_'>
```

IsHoliday seems to be the culprit here; one is a string, the other a boolean.

In [21]:

```
#Converting the datatype
df1.IsHoliday = df1.IsHoliday.astype(bool)
```

In [22]:

```
#Remerging
merged = pd.merge(df1, df2)
print(merged.shape)
merged.head()
```

```
(31817, 16)
```

Out[22]:

	Store	Dept	Date	Weekly_Sales	IsHoliday	Type	Size	Temperature	Fuel_Price	MarkDown1	MarkDown2	MarkDown3	MarkDown4	MarkDown5
0	1	1	2010-02-12	46039.49	True	A	151315	38.51	2.548	NaN	NaN	NaN	NaN	NaN
1	1	2	2010-02-12	44682.74	True	A	151315	38.51	2.548	NaN	NaN	NaN	NaN	NaN
2	1	3	2010-02-12	10887.84	True	A	151315	38.51	2.548	NaN	NaN	NaN	NaN	NaN
3	1	4	2010-02-12	35351.21	True	A	151315	38.51	2.548	NaN	NaN	NaN	NaN	NaN

4	1	5	2010-02-12	29620.81	True	A	151315	38.51	2.548	NaN	NaN	NaN	NaN	NaN	2

In [23]:
```
merged.to_csv('Merged_Store_Data.csv', index=False)
```

Summary

Nice work! You're working more and more independently through the workflow, and ensuring data integrity!