Exploratory Data Analysis

Capstone Step 5

Choices Regarding Data Cleaning/Transformation

During prototyping I observed a few high level improvements possible via treating the raw data. Specific choices are annoted in screenshots on the subsequent slides.

1. Location file is entirely redundant and will be descoped.

2. A number of columns are redundant and will be removed after verifying the other locations offer equivalent data

3. A number of columns can be joined/consolidated

4. A number of columns can be converted from varchar to numerical type

Choices Regarding Pipeline Automation

The prototyping phase helped form the following approach to automation in order to achieve the optimal reliability:

1. To handle embedded commas within csv columns, use Linux csvkit’s csvcut tool from Python (cut and awk both have tremendous difficulty).

2 . Use sqlalchemy create_engine, pandas read_csv and dataframe.to_sql instead of looping through each row to execute mysql_connector_python insert statement. The latter fails in cases of NULL values.

3. Dot not declare any columns NOT NULL, and err on the side of caution with strings (TEXT instead of VARCHAR).

4. Delete files after processing to keep to manageable level.

Pipeline Script: transformdata.py

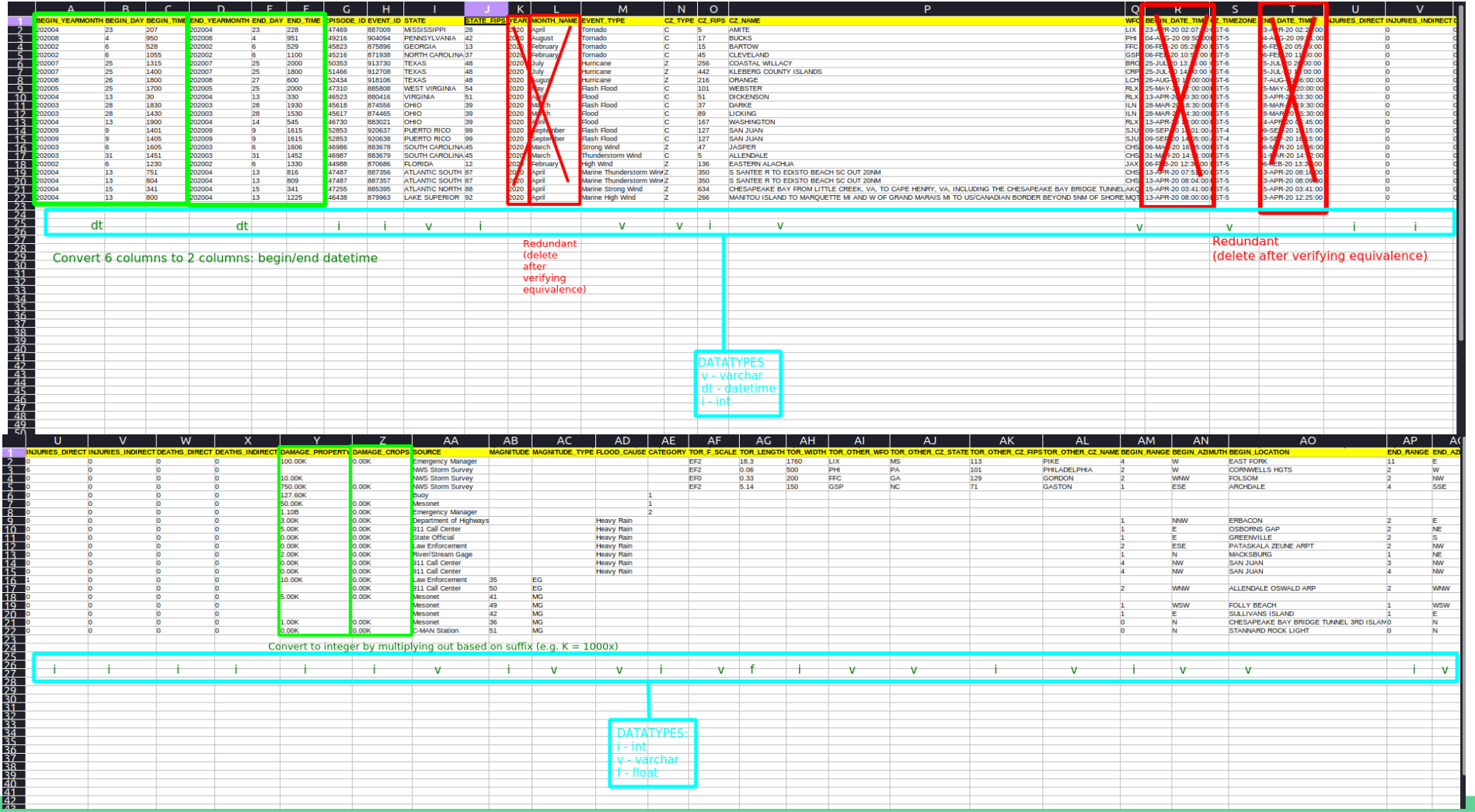
transformdata.py

```
"""
Script to transform raw csv files into clean final format and load to mysql
"""

import mysql.connector
from mysql.connector import errorcode
from pathlib import Path
import os
import glob
import pandas as pd
from sqlalchemy import  create_engine, types
import csv

def get_db_connection():
    """
    2_1 Setup database connection
    In order to make a query against the database table, we need to first connect to it. A connection
    can be established only when the user provides the proper target host, port, and user
    credentials
    """
    connection = None
    try:
        connection = mysql.connector.connect(user='root',password='',host='127.0.0.1',port='3306')
    except Exception as error:
        print("Error while connecting to database for job tracker", error)
    return connection

def initialize_database():
    cnx = get_db_connection()
    cursor = cnx.cursor()
    try:
        cursor.execute("DROP DATABASE IF EXISTS {}".format(DB_NAME))
        cursor.execute("CREATE DATABASE {}".format(DB_NAME))
    except mysql.connector.Error as err:
        print("Failed creating database: {}".format(err))
        exit(1)
    try:
        cursor.execute("USE {}".format(DB_NAME))
    except mysql.connector.Error as err:
        print("Database {} does not exists.".format(DB_NAME))
        if err.errno == errorcode.ER_BAD_DB_ERROR:
            create_database(cursor)
            print("Database {} created successfully.".format(DB_NAME))
            cnx.database = DB_NAME
        else:
            print(err)
            exit(1)

    TABLES = {}
    TABLES['details'] = (
        "CREATE TABLE details ("
        "  BEGIN_YEARMONTH VARCHAR(6),"
        "  BEGIN_DAY VARCHAR(2),"
        "  BEGIN_TIME VARCHAR(4),"
        "  END_YEARMONTH VARCHAR(6),"
        "  END_DAY VARCHAR(2),"
        "  END_TIME VARCHAR(4),"
        "  EPISODE_ID INT,"
        "  EVENT_ID INT,"
        "  STATE TEXT,"
```

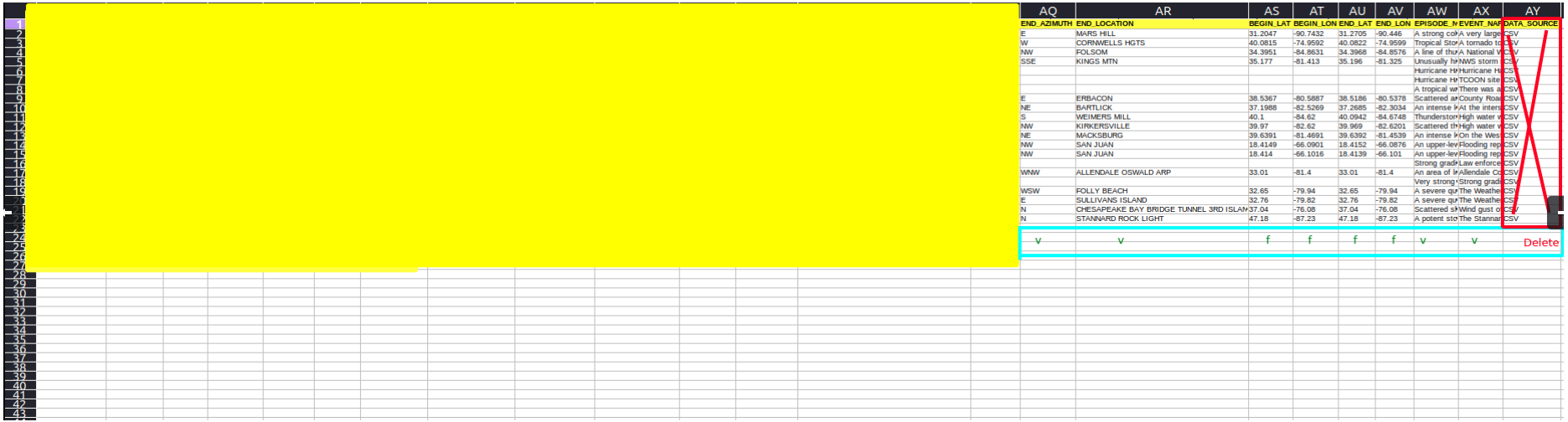
transformdata.py

```
    TABLES['fatalities'] = (
        "CREATE TABLE fatalities ("
        "  FATALITY_ID INT,"
        "  EVENT_ID INT,"
        "  FATALITY_TYPE VARCHAR(1),"
        "  FATALITY_DATE VARCHAR(19),"
        "  FATALITY_AGE INT DEFAULT NULL,"
        "  FATALITY_SEX ENUM('M','F'),"
        "  FATALITY_LOCATION TEXT,"
        "  EVENT_YEARMONTH VARCHAR(6)"
        ")")

    for table_name in TABLES:
        table_description = TABLES[table_name]
        try:
            print("Creating table {}: ".format(table_name), end='')
            cursor.execute(table_description)
        except mysql.connector.Error as err:
            if err.errno == errorcode.ER_TABLE_EXISTS_ERROR:
                print("already exists.")
            else:
                print(err.msg)
        else:
            print("OK")

    return cnx

def load_csv(filepath, table):
    df = pd.read_csv(filepath, sep = ',')
    df.to_sql(table,con=engine,index=False,if_exists='append')

def cut_details(filepath):
    #details needs cols 11 12 18 20 51 removed
    os.system("csvcut --columns=1,2,3,4,5,6,7,8,9,10,13,14,15,16,17,19,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50 " + filepath + " > details" + filepath[-18:])

def cut_fatals(filepath):
    #fatalities needs cols 1 2 3 removed
    os.system("csvcut --columns=4,5,6,7,8,9,10,11 " + filepath + " > fatalities" + filepath[-10:])

if __name__ == "__main__":
    print('***** Start of data transform script *****\n')
    DB_NAME = 'Capstone'
    cnx = initialize_database()
    engine = create_engine('mysql+pymysql://root@localhost/Capstone')
    detlist = sorted(glob.glob("/home/conner/Capstone/data/unzipped/storm_details_*"))
    fatlist = sorted(glob.glob("/home/conner/Capstone/data/unzipped/storm_fatalities_*"))
    for d in detlist:
        print('Cutting ' + d)
        cut_details(d)
        print('Loading ' + d[-25:])
        load_csv(d[-25:],'details')
        os.system('rm ' + d[-25:])
    for f in fatlist:
        print('Cutting ' + f)
        cut_fatals(f)
        print('Loading ' + f[-28:])
        load_csv(f[-28:],'fatalities')
        os.system('rm ' + f[-28:])
    print('***** End of script *****')
```

Details


dt dt i i v i v v i v v v i i
Convert 6 columns to 2 columns: begin/end datetime
Redundant (delete after verifying equivalence)
Redundant (delete after verifying equivalence)
DATATYPES
v - varchar
dt - datetime
i - int
Convert to integer by multiplying out based on suffix (e.g. K = 1000x)
i i i i i i v i v v i v f i v v i v i v v i v
DATATYPES:
i - int
v - varchar
f - float

Details (cont)



	AQ	AR	AS	AT	AU	AV	AW	AX	AY
1	END_AZIMUTH	END_LOCATION	BEGIN_LAT	BEGIN_LON	END_LAT	END_LON	EPISODE_N	EVENT_NAR	DATA_SOURCE
2	E	MARS HILL	31.2047	-90.7432	31.2705	-90.446	A strong co	A very large	CSV
3	W	CORNWELLS HGTS	40.0815	-74.9592	40.0822	-74.9599	Tropical Sto	A tornado to	CSV
4	NW	FOLSOM	34.3951	-84.8631	34.3968	-84.8576	A line of thu	A National V	CSV
5	SSE	KINGS MTN	35.177	-81.413	35.196	-81.325	Unusually hi	NWS storm	CSV
6							Hurricane Hi	Hurricane Hi	CSV
7							Hurricane Hi	TCOON site	CSV
8							A tropical wa	There was a	CSV
9	E	ERBACON	38.5367	-80.5887	38.5186	-80.5378	Scattered a	County Roa	CSV
10	NE	BARTLICK	37.1988	-82.5269	37.2685	-82.3034	An intense l	At the inters	CSV
11	S	WEIMERS MILL	40.1	-84.62	40.0942	-84.6748	Thunderstor	High water v	CSV
12	NW	KIRKERSVILLE	39.97	-82.62	39.969	-82.6201	Scattered th	High water v	CSV
13	NE	MACKSBURG	39.6391	-81.4691	39.6392	-81.4539	An intense l	On the Wes	CSV
14	NW	SAN JUAN	18.4149	-66.0901	18.4152	-66.0876	An upper-lev	Flooding rep	CSV
15	NW	SAN JUAN	18.414	-66.1016	18.4139	-66.101	An upper-lev	Flooding rep	CSV
16							Strong grad	Law enforce	CSV
17	WNW	ALLENDALE OSWALD ARP	33.01	-81.4	33.01	-81.4	An area of l	Allendale Co	CSV
18							Very strong	Strong grad	CSV
19	WSW	FOLLY BEACH	32.65	-79.94	32.65	-79.94	A severe qu	The Weathe	CSV
20	E	SULLIVANS ISLAND	32.76	-79.82	32.76	-79.82	A severe qu	The Weathe	CSV
21	N	CHESAPEAKE BAY BRIDGE TUNNEL 3RD ISLAN	37.04	-76.08	37.04	-76.08	Scattered sh	Wind gust o	CSV
22	N	STANNARD ROCK LIGHT	47.18	-87.23	47.18	-87.23	A potent sto	The Stannar	CSV
23									
24	v	v	f	f	f	f	v	v	Delete
25									
26									
27									
28									
29									
30									
31									
32									
33									
34									
35									
36									
37									
38									
39									
40									
41									
42									
43									

Fatalities

	A	B	C	D	E	F	G	H	I	J	K
1	FAT_YEARMONTH	FAT_DAY	FAT_TIME	FATALITY_ID	EVENT_ID	FATALITY_TYPE	FATALITY_DATE	FATALITY_AGE	FATALITY_SEX	FATALITY_LOCATION	EVENT_YEARMONTH
2	202106	9	0	42960	953511	D	06/09/2021 00:00:00	70	M	Golfing	202106
3	202107	20	0	43206	961309	I	07/20/2021 00:00:00	32	M	Other	202107
4	202107	20	0	43207	961309	I	07/20/2021 00:00:00	31	M	Other	202107
5	202107	20	0	43208	961309	I	07/20/2021 00:00:00		M	Other	202107
6	202107	27	0	44279	964033	I	07/27/2021 00:00:00		F	Other	202107
7	202103	28	0	42898	951050	D	03/28/2021 00:00:00	61	F	In Water	202103
8	202104	24	0	42962	954336	D	04/24/2021 00:00:00	2		Permanent Home	202104
9	202106	10	0	42963	954408	D	06/10/2021 00:00:00		M	In Water	202106
10	202106	14	0	44478	970299	D	06/14/2021 00:00:00	23	M	Outside/Open Areas	202106
11	202106	13	0	44482	970319	D	06/13/2021 00:00:00	32	F	Outside/Open Areas	202106
12	202106	13	0	44483	970319	D	06/13/2021 00:00:00	42	M	Outside/Open Areas	202106
13	202106	13	0	44484	970319	D	06/13/2021 00:00:00	29	M	Outside/Open Areas	202106
14	202106	15	0	44485	970319	D	06/15/2021 00:00:00	34	F	Outside/Open Areas	202106
15	202106	17	0	44487	970319	D	06/17/2021 00:00:00	28	M	Outside/Open Areas	202106
16	202106	20	0	44488	970319	D	06/20/2021 00:00:00	19	M	Outside/Open Areas	202106
17	202106	20	0	44489	970319	D	06/20/2021 00:00:00	42	M	Outside/Open Areas	202106
18	202106	15	0	44490	970319	D	06/15/2021 00:00:00	35	M	Outside/Open Areas	202106
19	202106	17	0	44491	970319	D	06/17/2021 00:00:00	36	M	Outside/Open Areas	202106
20	202106	19	0	44492	970319	D	06/19/2021 00:00:00	54	M	Outside/Open Areas	202106
21	202106	13	0	44479	970320	D	06/13/2021 00:00:00	35	F	Outside/Open Areas	202106
22	202106	12	0	44480	970320	D	06/12/2021 00:00:00	22	M	Outside/Open Areas	202106
23	202106	15	0	44481	970320	D	06/15/2021 00:00:00	37	M	Outside/Open Areas	202106
24	202106	2	0	43761	970487	D	06/02/2021 00:00:00		M	Outside/Open Areas	202106
25	202106	2	0	43762	970487	D	06/02/2021 00:00:00	20	M	Outside/Open Areas	202106
26	202106	7	0	43770	970488	D	06/07/2021 00:00:00	40	M	Outside/Open Areas	202106
27	202106	6	0	43771	970490	D	06/06/2021 00:00:00	47	M	Outside/Open Areas	202106
28											
29				I	I	V	DT	I	V	V	I

Redundant
(Will remove after verifying equivalent to FATALITY_DATE)

Datatypes:
I - Int
V - Varchar
DT - Datetime

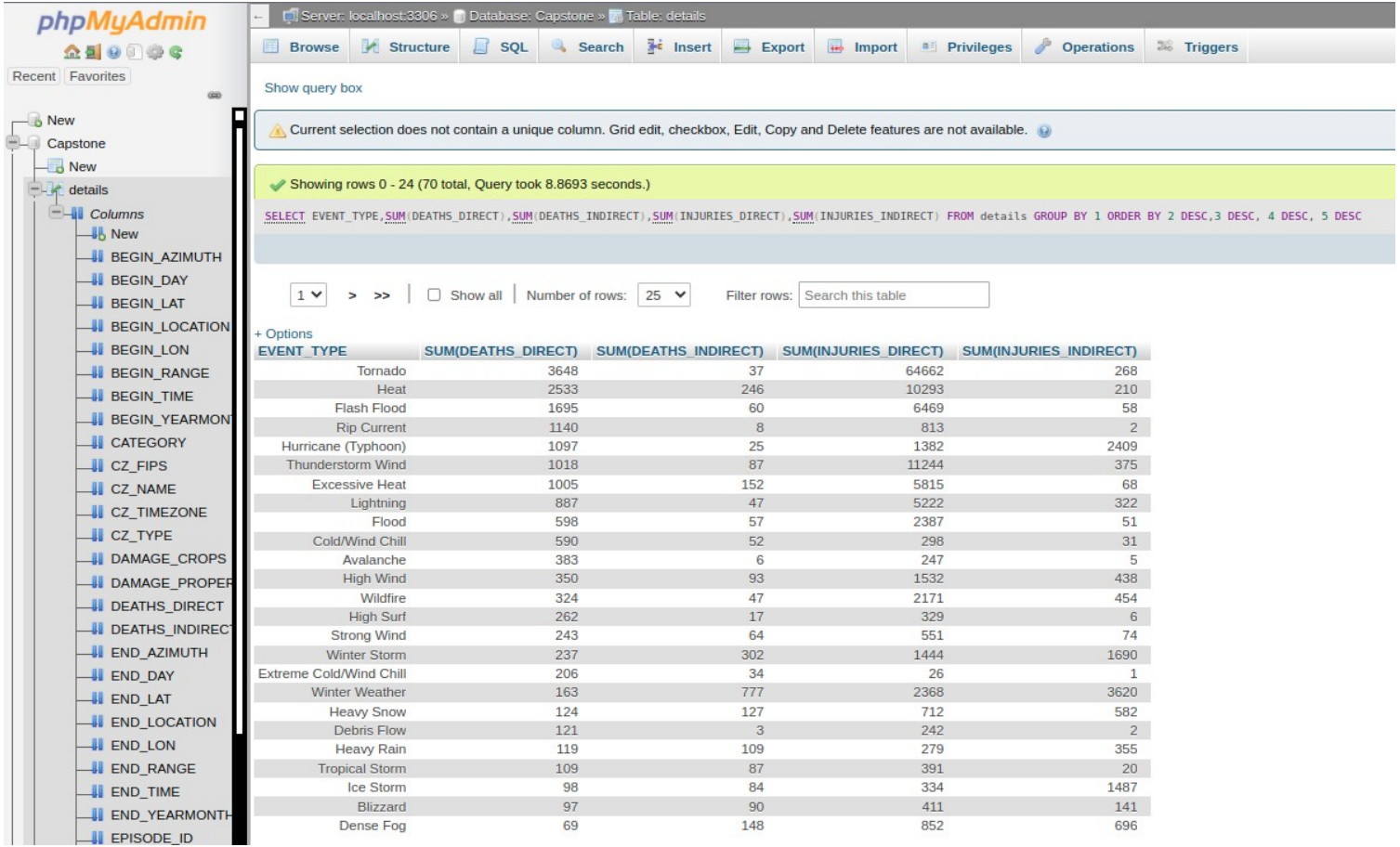
The event can occur prior to the fatality, so this is not redundant

After running transform.py, both tables are fully populated in mysql:

```
mysql> select count(*) from fatalities;
+----------+
| count(*) |
+----------+
|    19288 |
+----------+
1 row in set (0.01 sec)

mysql> select count(*) from details;
+----------+
| count(*) |
+----------+
|  1682037 |
+----------+
1 row in set (0.64 sec)

mysql> select BEGIN_YEARMONTH,BEGIN_DAY,BEGIN_TIME,END_YEARMONTH,END_DAY,END_TIME, EPISODE_ID,EVENT_ID,STATE,STATE_FIPS,EVENT_TYPE,CZ_TYPE from details limit 5;
+-----------------+-----------+------------+---------------+---------+----------+------------+----------+-------------+------------+-------------------+---------+
| BEGIN_YEARMONTH | BEGIN_DAY | BEGIN_TIME | END_YEARMONTH | END_DAY | END_TIME | EPISODE_ID | EVENT_ID | STATE       | STATE_FIPS | EVENT_TYPE        | CZ_TYPE |
+-----------------+-----------+------------+---------------+---------+----------+------------+----------+-------------+------------+-------------------+---------+
| 197009          | 4         | 1650       | 197009        | 4       | 1650     |       NULL |  9981065 | ARKANSAS    |          5 | Thunderstorm Wind | C       |
| 197010          | 8         | 1655       | 197010        | 8       | 1655     |       NULL | 10058129 | MISSOURI    |         29 | Tornado           | C       |
| 197006          | 4         | 1615       | 197006        | 4       | 1615     |       NULL | 10047491 | MISSISSIPPI |         28 | Thunderstorm Wind | C       |
| 197006          | 11        | 2200       | 197006        | 11      | 2200     |       NULL | 10056944 | MISSOURI    |         29 | Thunderstorm Wind | C       |
| 197007          | 1         | 2100       | 197007        | 1       | 2100     |       NULL | 10052308 | MINNESOTA   |         27 | Thunderstorm Wind | C       |
+-----------------+-----------+------------+---------------+---------+----------+------------+----------+-------------+------------+-------------------+---------+
5 rows in set (0.00 sec)

mysql> select CZ_FIPS,CZ_NAME,WFO,CZ_TIMEZONE,INJURIES_DIRECT,INJURIES_INDIRECT,DEATHS_DIRECT,DEATHS_INDIRECT,DAMAGE_PROPERTY,DAMAGE_CROPS,SOURCE,MAGNITUDE,MAGNITUDE_TYPE,FLOOD_CAUSE,CATEGORY from details limit 5;
+---------+-------------+------+-------------+-----------------+-------------------+---------------+-----------------+-----------------+--------------+--------+-----------+----------------+-------------+----------+
| CZ_FIPS | CZ_NAME     | WFO  | CZ_TIMEZONE | INJURIES_DIRECT | INJURIES_INDIRECT | DEATHS_DIRECT | DEATHS_INDIRECT | DAMAGE_PROPERTY | DAMAGE_CROPS | SOURCE | MAGNITUDE | MAGNITUDE_TYPE | FLOOD_CAUSE | CATEGORY |
+---------+-------------+------+-------------+-----------------+-------------------+---------------+-----------------+-----------------+--------------+--------+-----------+----------------+-------------+----------+
|      93 | MISSISSIPPI | NULL | CST         |               0 |                 0 |             0 |               0 | 0               | 0            | NULL   |         0 | NULL           | NULL        |     NULL |
|     197 | SCHUYLER    | NULL | CST         |               0 |                 0 |             0 |               0 | 25K             | 0            | NULL   |         0 | NULL           | NULL        |     NULL |
|     159 | WINSTON     | NULL | CST         |               0 |                 0 |             0 |               0 | 0               | 0            | NULL   |         0 | NULL           | NULL        |     NULL |
|       9 | BARRY       | NULL | CST         |               0 |                 0 |             0 |               0 | 0               | 0            | NULL   |         0 | NULL           | NULL        |     NULL |
|      93 | MEEKER      | NULL | CST         |               0 |                 0 |             0 |               0 | 0               | 0            | NULL   |         0 | NULL           | NULL        |     NULL |
+---------+-------------+------+-------------+-----------------+-------------------+---------------+-----------------+-----------------+--------------+--------+-----------+----------------+-------------+----------+
5 rows in set (0.00 sec)

mysql> select BEGIN_RANGE,BEGIN_AZIMUTH,BEGIN_LOCATION,END_RANGE,END_AZIMUTH,END_LOCATION,BEGIN_LAT,BEGIN_LON,END_LAT,END_LON from details limit 5;
+-------------+---------------+----------------+-----------+-------------+--------------+-----------+-----------+---------+---------+
| BEGIN_RANGE | BEGIN_AZIMUTH | BEGIN_LOCATION | END_RANGE | END_AZIMUTH | END_LOCATION | BEGIN_LAT | BEGIN_LON | END_LAT | END_LON |
+-------------+---------------+----------------+-----------+-------------+--------------+-----------+-----------+---------+---------+
|           0 | NULL          | NULL           |         0 | NULL        | NULL         |        36 |       -90 |    NULL |    NULL |
|           0 | NULL          | NULL           |         0 | NULL        | NULL         |        40 |       -93 |      40 |     -93 |
|           0 | NULL          | NULL           |         0 | NULL        | NULL         |        33 |       -89 |    NULL |    NULL |
|           0 | NULL          | NULL           |         0 | NULL        | NULL         |        37 |       -94 |    NULL |    NULL |
|           0 | NULL          | NULL           |         0 | NULL        | NULL         |        45 |       -94 |    NULL |    NULL |
+-------------+---------------+----------------+-----------+-------------+--------------+-----------+-----------+---------+---------+
5 rows in set (0.00 sec)

mysql> select  EVENT_NARRATIVE,EPISODE_NARRATIVE from details ORDER BY BEGIN_YEARMONTH desc limit 1;
+------------------------------------------------------------------------------------------+------------------------------------------------------------------------------------------------------------------------------------------------------------------
------+
| EVENT_NARRATIVE                                                                          | EPISODE_NARRATIVE
      |
+------------------------------------------------------------------------------------------+------------------------------------------------------------------------------------------------------------------------------------------------------------------
------+
| In Danvers, water was up to the floor boards of a SUV (about 8 to 12 inches deep) on Village Post Road. | Strong southerly wind flow out ahead of an advancing cold front produced strong to damaging wind gusts and some heavy rain that caused some street and basement floo
ding. |
+------------------------------------------------------------------------------------------+------------------------------------------------------------------------------------------------------------------------------------------------------------------
------+
1 row in set (2.29 sec)

mysql>
```

conner@Linuxclevo: ~/testgz/data... | [Solve DtypeWarning: Columns ha... | [observations.txt — ~/Capstone —... | [Step4] | [WhatSie: (1) WhatsApp]

Data is returning meaningful queries for analysis:



EVENT_TYPE	SUM(DEATHS_DIRECT)	SUM(DEATHS_INDIRECT)	SUM(INJURIES_DIRECT)	SUM(INJURIES_INDIRECT)
Tornado	3648	37	64662	268
Heat	2533	246	10293	210
Flash Flood	1695	60	6469	58
Rip Current	1140	8	813	2
Hurricane (Typhoon)	1097	25	1382	2409
Thunderstorm Wind	1018	87	11244	375
Excessive Heat	1005	152	5815	68
Lightning	887	47	5222	322
Flood	598	57	2387	51
Cold/Wind Chill	590	52	298	31
Avalanche	383	6	247	5
High Wind	350	93	1532	438
Wildfire	324	47	2171	454
High Surf	262	17	329	6
Strong Wind	243	64	551	74
Winter Storm	237	302	1444	1690
Extreme Cold/Wind Chill	206	34	26	1
Winter Weather	163	777	2368	3620
Heavy Snow	124	127	712	582
Debris Flow	121	3	242	2
Heavy Rain	119	109	279	355
Tropical Storm	109	87	391	20
Ice Storm	98	84	334	1487
Blizzard	97	90	411	141
Dense Fog	69	148	852	696