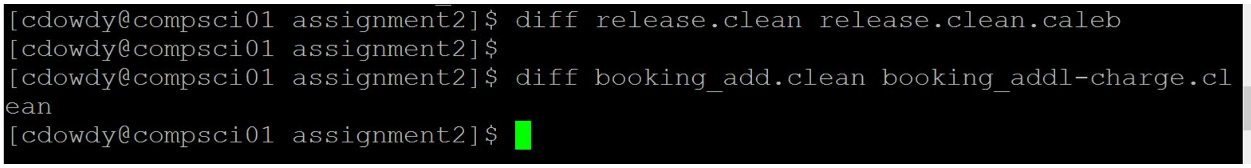
Caleb Dowdy
Data Cleaning and SQLite

1. a
assignment2releasefinal.py --- python program to read in the raw data and produce release.clean.caleb

assignment2add_charges.py --- python program to read in the raw data and produce booking_add.clean

Working Directory: /usr/share/databases/cdowdy/assignment2

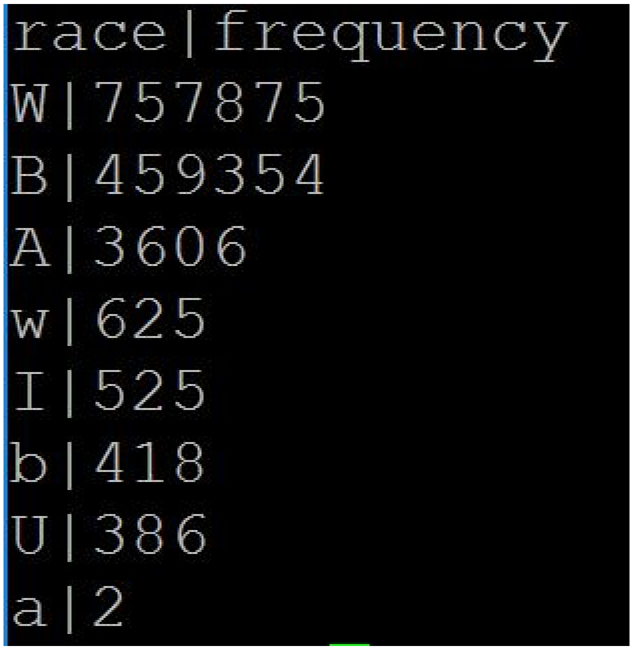


```
[cdowdy@compsci01 assignment2]$ diff release.clean release.clean.caleb
[cdowdy@compsci01 assignment2]$
[cdowdy@compsci01 assignment2]$ diff booking_add.clean booking_addl-charge.cl
ean
[cdowdy@compsci01 assignment2]$
```

1.b
booking_addl-charge.clean lacks much of the categorical data present in *bookings.clean*.

Each row in *booking_addl-charge.clean* contains the inmate's booking number, charge type, charge, court, and case number. It seems that the rows are grouped by the booking number; if an inmate had “n” charges in this set then “n” rows of data would be grouped together. Grouping by booking number isn’t present in *bookings.clean* -- if you intend to document repeat offenses you need to order the entire dataset by booking number. SQLite will interpret the names in *bookings.clean* as multiple columns due to the commas present in that field. In addition having multiple offenses present in a given line (bookings.clean) creates difficulty if you intend to use those offenses in your research. The convenience of *booking_addl-charge.clean* is going to be dependent on what questions you are trying to investigate. *booking_addl-charge.clean* lacks much of the categorical data present in *bookings.clean* and as a result is roughly 48% the size. Obtaining the average number of incidents per inmate or learning the frequency distribution of charges would be possible with both datasets, but would take up less memory with *booking_addl-charge.clean* .

How many arrests by race:



```
race|frequency
W|757875
B|459354
A|3606
w|625
I|525
b|418
U|386
a|2
```

```
select race, count (*) as frequency
from bookingsB
group by race
order by count(*) desc;
```

The answer to how many arrests are made for any race present in our dataset is given by the SQL command above. I believe that no further booking data would be required to answer this question better, however the addition of other data (population totals by race for Tampa) would be welcomed.

How do we know if they are homeless?

Using the addresses listed under the table “*homeless_addresses*” : select * from homeless_addresses; we can identify which individuals in our sample population are homeless. While I didn’t create the table “*homeless_addresses”* I can assume that these addresses were selected based upon their reputation as havens for the homeless (some may even be homeless shelters).

For the Homeless how many arrests are there for each charge?

```
First:
create table charge1 as select * from bookingsB
where address in (select address from homeless_addresses);
```

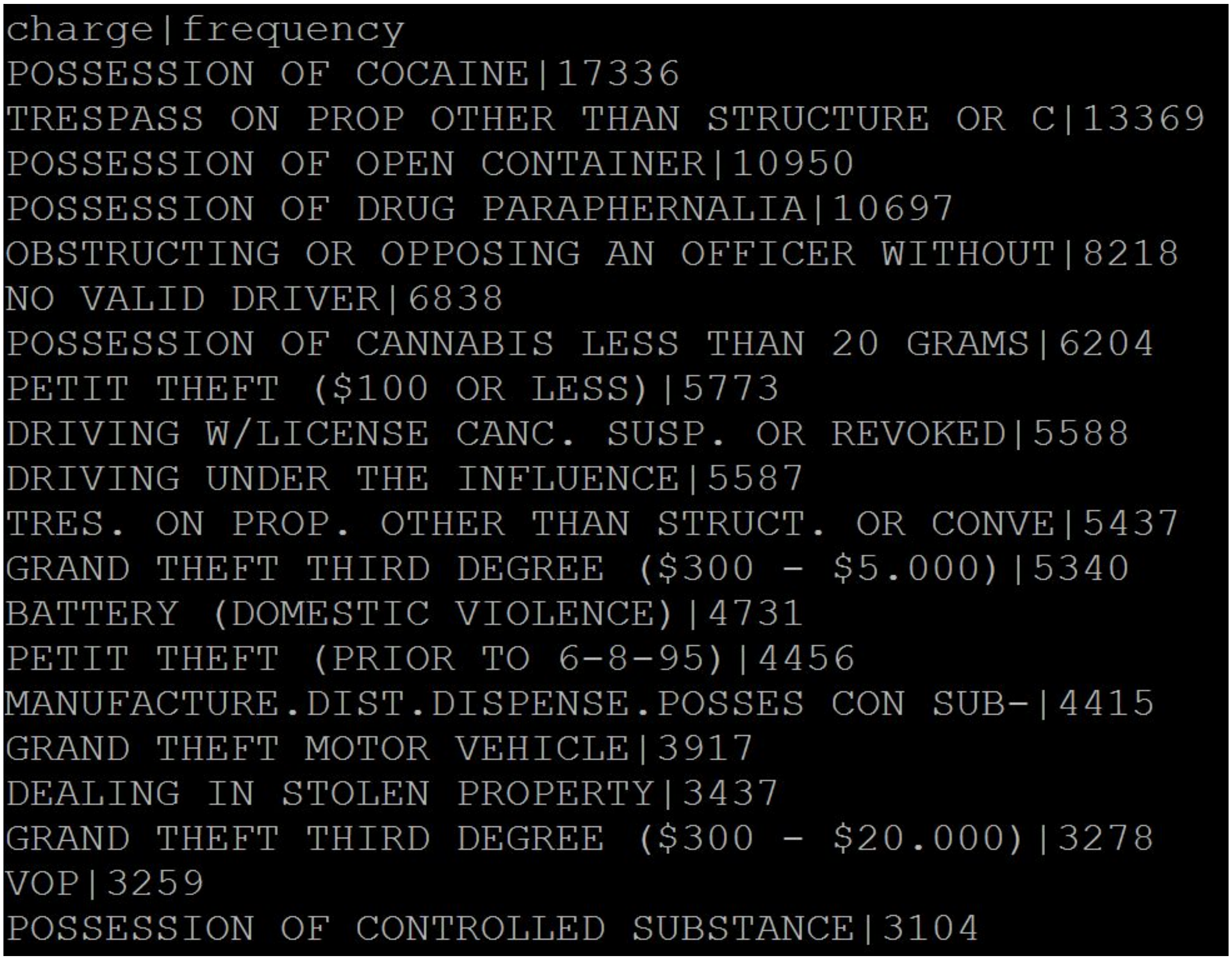
```
# all the homeless offenses in bookingsB

second :
create table charge2 as select * from booking_addl_charge
where bookingNumber in (select bookingNumber from charge1);
# All the homeless offenses in booking_addl_charge

INSERT INTO charge2 SELECT
bookingNumber,chargeType,charge,court,caseNumber FROM charge1;


SELECT charge, count(*) as frequency
FROM charge2
GROUP BY charge
ORDER BY count(*) desc;
```

FREQUENT CHARGES AGAINST THE HOMELESS



```
charge|frequency
POSSESSION OF COCAINE|17336
TRESPASS ON PROP OTHER THAN STRUCTURE OR C|13369
POSSESSION OF OPEN CONTAINER|10950
POSSESSION OF DRUG PARAPHERNALIA|10697
OBSTRUCTING OR OPPOSING AN OFFICER WITHOUT|8218
NO VALID DRIVER|6838
POSSESSION OF CANNABIS LESS THAN 20 GRAMS|6204
PETIT THEFT ($100 OR LESS)|5773
DRIVING W/LICENSE CANC. SUSP. OR REVOKED|5588
DRIVING UNDER THE INFLUENCE|5587
TRES. ON PROP. OTHER THAN STRUCT. OR CONVE|5437
GRAND THEFT THIRD DEGREE ($300 - $5.000)|5340
BATTERY (DOMESTIC VIOLENCE)|4731
PETIT THEFT (PRIOR TO 6-8-95)|4456
MANUFACTURE.DIST.DISPENSE.POSSES CON SUB-|4415
GRAND THEFT MOTOR VEHICLE|3917
DEALING IN STOLEN PROPERTY|3437
GRAND THEFT THIRD DEGREE ($300 - $20.000)|3278
VOP|3259
POSSESSION OF CONTROLLED SUBSTANCE|3104
```

Please notice that my approach did not use a UNION statement as I wanted to obtain the charge count NOT charge count / arrest. Suppose the following, a police officer arrested a homeless individual and charged them with two counts of battery; a UNION statement would flatten the two counts of battery under the one booking number, thus in my opinion, understating the impact of the booking. Additional data could potentially change the order that charges appear in the table above.

What do homeless get charged with in greater proportion than the entire population?

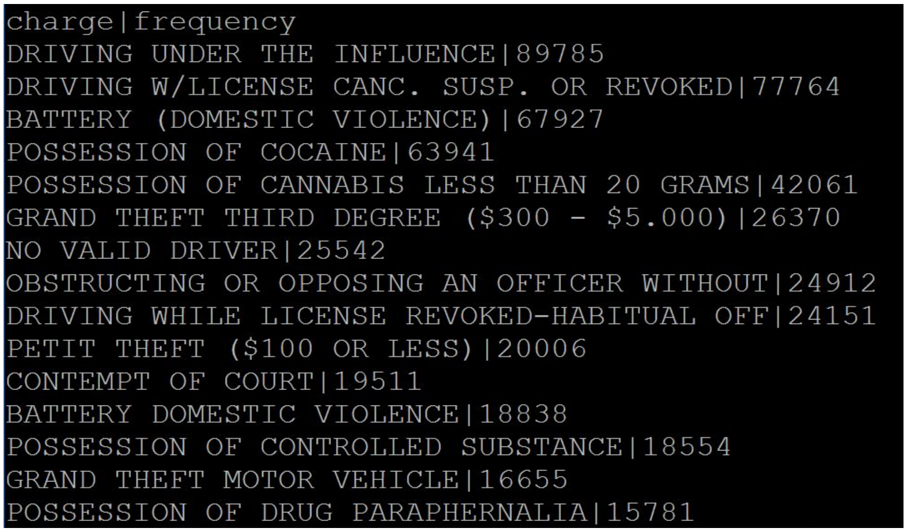
```
First:
create table gencharge1 as select * from bookingsB
where address not in (select address from homeless_addresses);

Second:
create table gencharge2 as select * from booking_addl_charge
where bookingNumber not in (select bookingNumber from gencharge1);


INSERT INTO gencharge2 SELECT
bookingNumber,chargeType,charge,court,caseNumber FROM gencharge1;

SELECT charge, count(*) as frequency
FROM gencharge2
GROUP BY charge
ORDER BY count(*) desc limit 30;
```

FREQUENT CHARGES AGAINST THE NON-HOMELESS



```
charge|frequency
DRIVING UNDER THE INFLUENCE|89785
DRIVING W/LICENSE CANC. SUSP. OR REVOKED|77764
BATTERY (DOMESTIC VIOLENCE)|67927
POSSESSION OF COCAINE|63941
POSSESSION OF CANNABIS LESS THAN 20 GRAMS|42061
GRAND THEFT THIRD DEGREE ($300 - $5.000)|26370
NO VALID DRIVER|25542
OBSTRUCTING OR OPPOSING AN OFFICER WITHOUT|24912
DRIVING WHILE LICENSE REVOKED-HABITUAL OFF|24151
PETIT THEFT ($100 OR LESS)|20006
CONTEMPT OF COURT|19511
BATTERY DOMESTIC VIOLENCE|18838
POSSESSION OF CONTROLLED SUBSTANCE|18554
GRAND THEFT MOTOR VEHICLE|16655
POSSESSION OF DRUG PARAPHERNALIA|15781
```

The SQL code above is for determining the frequency distribution of charges against the non-homeless population in our dataset. Unfortunately, the code as it stands does not address the question of proportion. I attempted divide the individual charge count by the total population to obtain a proportion, however the result was a non-integer value which lead to a rounding error.

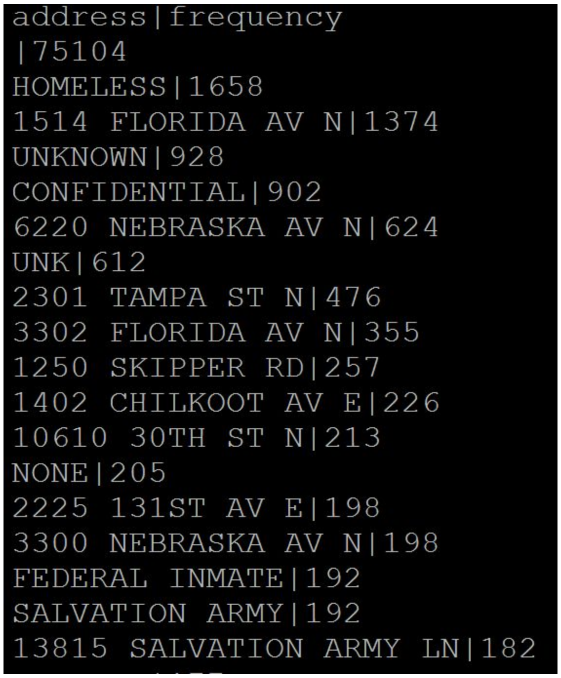
What is the average stay in jail?

```
SELECT AVG(julianday(releaseDate) - julianday(arrestDate)) FROM
bookingsA;
```

The SQL statement above returns that the average stay in jail is 25 days. The data as it stands provides a useful metric so no further data is required to improve this answer. Tracking the average stay in jail over the years would be an interesting expansion of the question above.

Where Do Arrests Take Place?

```
SELECT address, count(*) as frequency
FROM bookingsB
GROUP BY address
ORDER BY count(*) desc limit 20;
```



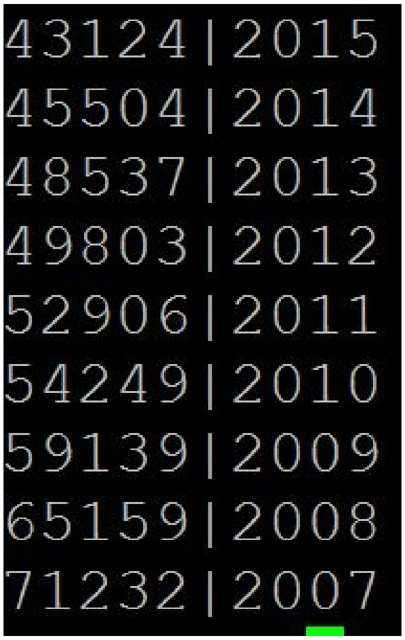
```
address|frequency
|75104
HOMELESS|1658
1514 FLORIDA AV N|1374
UNKNOWN|928
CONFIDENTIAL|902
6220 NEBRASKA AV N|624
UNK|612
2301 TAMPA ST N|476
3302 FLORIDA AV N|355
1250 SKIPPER RD|257
1402 CHILKOOT AV E|226
10610 30TH ST N|213
NONE|205
2225 131ST AV E|198
3300 NEBRASKA AV N|198
FEDERAL INMATE|192
SALVATION ARMY|192
13815 SALVATION ARMY LN|182
```

The SQL code aboves answers the following question: “How many arrests are there for the given address variable?”. The results would indicate which addresses produce the most number of incidents. Additional data may include the address of where the arrests actually occurred.

Arrests per Year Open Container

```
SELECT COUNT(bookingNumber) AS arrests, SUBSTR(arrestDate,1,4) AS
year FROM bookingsA
WHERE bookingNumber IN(SELECT bookingNumber FROM homelessCharges
WHERE charge LIKE "POSSESSION OF OPEN CONTAINER")
GROUP BY SUBSTR(arrestDate,1,4)
ORDER BY year
LIMIT 10;
```

ARRESTS PER YEAR FOR OPEN CONTAINER



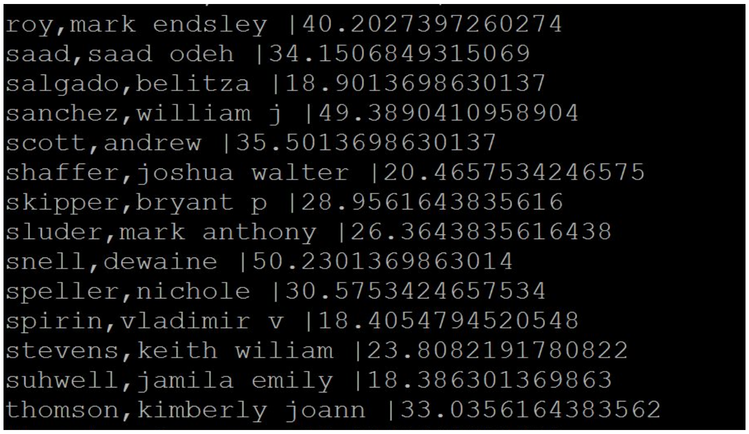
```
43124|2015
45504|2014
48537|2013
49803|2012
52906|2011
54249|2010
59139|2009
65159|2008
71232|2007
```

The SQL statement above addresses the question of how many arrests per year are related to open container charges. Additional data in the form of new entries would be a welcome addition, but unnecessary.

How old was each homeless person when first arrested?

```
SELECT name, MIN(julianday(arrestDate) - julianday(DOB))/365 AS age
FROM bookingsA
```

```
WHERE bookingNumber IN (SELECT bookingNumber FROM homelessCharges)
GROUP BY name;
```



```
roy,mark endsley |40.2027397260274
saad,saad odeh |34.1506849315069
salgado,belitza |18.9013698630137
sanchez,william j |49.3890410958904
scott,andrew |35.5013698630137
shaffer,joshua walter |20.4657534246575
skipper,bryant p |28.9561643835616
sluder,mark anthony |26.3643835616438
snell,dewaine |50.2301369863014
speller,nichole |30.5753424657534
spirin,vladimir v |18.4054794520548
stevens,keith wiliam |23.8082191780822
suhwell,jamila emily |18.386301369863
thomson,kimberly joann |33.0356164383562
```

The SQL statement above addresses the question of how old was each homeless person was when they were first arrested. Minimization was required in the SQL query to detect which booking number occurred first and to then return their age at that point in time.