1.1.1_create_groups

March 11, 2021

1 1.1.x_create_groups

1.0.1 Create a table `inputs_all` of adult admissions and their input events from the mimic iii database

- postgres = 9.4 or higher

1.1 import libraries, connect to mimic database

[64]:
```
# only run this cell if you need to reset the connection to postgres database␣
 ↪after an error
conn.commit();
cur.close();
conn.close();
```

[1]:
```
import sys
from datetime import datetime
import pandas as pd

from importlib_metadata import version

# things to connect to the posgres database
import psycopg2
from sqlalchemy import create_engine, update

POSTGRES_CONNECT = os.environ.get("POSTGRES_CONNECT")
POSTGRES_ENGINE = os.environ.get("POSTGRES_ENGINE")
conn = psycopg2.connect(POSTGRES_CONNECT)
cur = conn.cursor();
cur.execute("""SET search_path = mimiciii;""")
engine = create_engine(POSTGRES_ENGINE)

libraries = ['pandas','sqlalchemy','psycopg2','tqdm']
print('last ran: ',datetime.now() )
print("Python Version:", sys.version[0:7])
print( "operating system:", sys.platform)

for lib in libraries:
    print(lib + ' version: ' + version(lib))
```

```
last ran:  2019-12-24 16:22:17.780177
Python Version: 3.7.3 (
operating system: darwin
pandas version: 0.24.2
sqlalchemy version: 1.3.3
psycopg2 version: 2.7.6.1
tqdm version: 4.32.1
```

1.2 1 Create Adult Tables

1.2.1 1.1 Make Adult Patient Table

- adults = **patients** that were 16 years or more at the time of admission
- total admissions in MIMICIII database = 58,976
- total adult admissions (16 or older) = 50,857

[2]:

```
cur.execute("""
DROP TABLE IF EXISTS mimiciii.patients_adult;
WITH
first_admission_time AS
(   SELECT pp.subject_id
           ,MIN (a.admittime) AS first_admittime
        , MIN( ROUND( (CAST(a.admittime AS date) - CAST(pp.dob AS date))/365.
 ↪242,2))AS first_admit_age
    FROM mimiciii.patients pp
        INNER JOIN mimiciii.admissions a
        ON pp.subject_id = a.subject_id
    GROUP BY pp.subject_id
    ORDER BY pp.subject_id)
, age AS
(   SELECT subject_id, first_admit_age
        , CASE
            WHEN first_admit_age >= 16
                THEN 'adult'
            ELSE 'pediatric'
            END AS age_group
    FROM first_admission_time)

SELECT p.*,f.first_admit_age
    INTO mimiciii.patients_adult
FROM mimiciii.patients p
    INNER JOIN age f
    ON p.subject_id = f.subject_id
    WHERE f.age_group LIKE 'adult';""")
```

```
conn.commit()
```

print total number of unique patient admissions & unique adult patient admissions

[3]:

```
cur.execute("""
SELECT COUNT(DISTINCT a.hadm_id) AS admissions_total
FROM  admissions a;""")

print(pd.DataFrame(cur.fetchall(), columns=['admissions_total']).
 ↪to_string(index=False))

cur.execute("""
SELECT COUNT(DISTINCT a.hadm_id) AS adult_admissions_total
FROM patients_adult p
INNER JOIN admissions a
    ON a.subject_id = p.subject_id;""")

print(pd.DataFrame(cur.fetchall(), columns=['adult_admissions_total']).
 ↪to_string(index=False))
```

```
admissions_total
           58976
adult_admissions_total
                 50857
```

print the stats from the dataframe we just created. the minimum `first_admit_age` should be 16.0. The maximum is 300. If the patient is older than 89, the patient's age is fixed to 300 to de-identify.

[6]:

```
cur.execute("""
DROP TABLE IF EXISTS mimiciii.inputevents_mv_adult;

SELECT i.*
    INTO inputevents_mv_adult
FROM mimiciii.inputevents_mv i
    INNER JOIN mimiciii.patients_adult p
    ON p.subject_id = i.subject_id;""")


# print number of unique adult mv admissions
cur.execute("""
SELECT COUNT(*) AS total_adult_mv_inputs
     , COUNT(DISTINCT hadm_id) AS unique_adult_admissions_mv_inputs
FROM mimiciii.inputevents_mv_adult;""")

print(pd.DataFrame(cur.fetchall(), columns=[␣
 ↪'total_adult_mv_inputs','unique_adult_admissions_mv_inputs']).
 ↪to_string(index=False))
```

```
total_adult_mv_inputs  unique_adult_admissions_mv_inputs
              3618905                              21876
```

1.2.2 1.3 Make Adult CV (CareVue) Input Events Table

- extract all the inputs (input_events = items put into the patient) for adults (**patients_adult**) in CareVue (**inputevents_cv**)

[7]:
```
cur.execute("""
DROP TABLE IF EXISTS mimiciii.inputevents_cv_adult;

SELECT i.*
    INTO mimiciii.inputevents_cv_adult
FROM mimiciii.inputevents_cv i
    INNER JOIN mimiciii.patients_adult p
    ON p.subject_id = i.subject_id;""")


# print number of total inputs (rows) & unique adult cv admissions
cur.execute("""
SELECT COUNT(*) AS total_adult_cv_inputs
     , COUNT(DISTINCT hadm_id) AS unique_adult_admissions_cv_inputs
FROM mimiciii.inputevents_cv_adult;""")

print(pd.DataFrame(cur.fetchall(), columns=['total_adult_cv_inputs',␣
 ↪'unique_adult_admissions_cv_inputs']).to_string(index=False))
```

```
total_adult_cv_inputs  unique_adult_admissions_cv_inputs
             15229603                              27138
```

1.2.3 1.4 Make Adult Chart Events Table

- takes a long time (more than an hour or 2)
- extract all chart events for adult patients (inner join **chartevents** to **patients_adult**)
- totaladult chart events = 280231912
- unique adult admissions chart events = 49,282

[8]:
```
cur.execute("""
DROP TABLE IF EXISTS mimiciii.chartevents_adult;

SELECT c.*
    INTO mimiciii.chartevents_adult
FROM mimiciii.chartevents c
    INNER JOIN mimiciii.patients_adult p
    ON p.subject_id = c.subject_id;""")
```

[9]:
```
# print number of inputs (rows) &  unique adult cv admissions
conn.commit()
cur.execute("""
SELECT COUNT(*) AS total_adult_chartevents
     , COUNT(distinct hadm_id) AS unique_adult_admissions_chartevents
FROM mimiciii.chartevents_adult;""")

print(pd.DataFrame(cur.fetchall(), columns=[␣
↪'total_adult_chartevents','unique_adult_admissions_chartevents']).
↪to_string(index=False))
```

```
total_adult_chartevents  unique_adult_admissions_chartevents
              280231912                                49282
```

1.2.4 1.5 Create a Table of All Adult Inputs (from mv and cv) and Adult Chartevents by merging

- **inputevents_cv_adult**
- **inputevents_mv_adult**
- **chartevents_adult**
- save as **inputs_all**

[10]:
```
cur.execute("""
DROP TABLE IF EXISTS mimiciii.inputs_all;

SELECT subject_id,
        hadm_id,
        icustay_id,
        charttime AS dt,
       'cv' AS source,
       itemid
INTO mimiciii.inputs_all
FROM mimiciii.inputevents_cv_adult UNION


SELECT subject_id,
        hadm_id,
        icustay_id,
        starttime as dt,
        'mv' as source,
        itemid
FROM mimiciii.inputevents_mv_adult UNION


SELECT subject_id,
        hadm_id,
        icustay_id,
        charttime as dt,
        'ce' as source,
        itemid
FROM mimiciii.chartevents_adult;""")
```

- print number of total inputs (rows) & unique adult admissions for the big inputs_all table
- adult_admissions_count = 49284
- inputs_count = 289509356

[11]:

```
cur.execute("""

SELECT COUNT(DISTINCT hadm_id) AS adult_admissions_count
     , COUNT(*) AS inputs_count
FROM mimiciii.inputs_all;""")

print(pd.DataFrame(cur.fetchall(),␣
 ↪columns=['adult_admissions_count','inputs_count' ]).to_string(index=False))
```

```
adult_admissions_count  inputs_count
                 49284     289509356
```

1.2.5 Clean Up, Commit, and Close

[12]:

```
conn.commit();
cur.close();
conn.close();
```