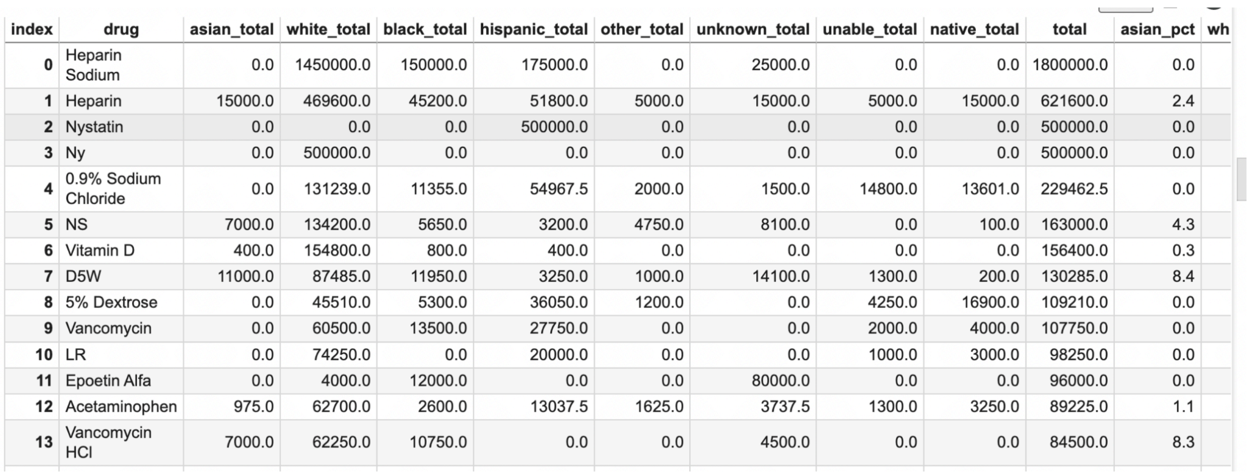
Part I (Relational)

1.)

```
WITH cleaned AS (
 SELECT
   drug,
   ethnicity,
   -- remove commas and lowercase everything
   lower(replace(dose_val_rx, ',', '')) AS dose_str
 FROM prescriptions
 JOIN admissions USING (hadm_id)
 WHERE dose_val_rx IS NOT NULL
),

ranges AS (
 SELECT
   drug,
   ethnicity,
   -- extract ranges like "10-20" and take average
   CASE
     WHEN dose_str ~ '^[0-9.]+-[0-9.]+$' THEN
       (CAST(split_part(dose_str, '-', 1) AS DOUBLE) +
CAST(split_part(dose_str, '-', 2) AS DOUBLE)) / 2
     WHEN dose_str ~ '^[0-9.]+$' THEN
       CAST(dose_str AS DOUBLE)
     ELSE NULL
   END AS dose_avg
 FROM cleaned
),

filtered AS (
 SELECT * FROM ranges
 WHERE dose_avg IS NOT NULL
),

agg AS (
 SELECT
   drug,
   SUM(CASE WHEN ethnicity LIKE '%ASIAN%' THEN dose_avg ELSE 0 END) AS
asian_total,
   SUM(CASE WHEN ethnicity LIKE '%WHITE%' THEN dose_avg ELSE 0 END) AS
white_total,
   SUM(CASE WHEN ethnicity LIKE '%BLACK%' THEN dose_avg ELSE 0 END) AS
black_total,
   SUM(CASE WHEN ethnicity LIKE '%HISPANIC%' THEN dose_avg ELSE 0 END) AS
hispanic_total,
   SUM(CASE WHEN ethnicity LIKE '%OTHER%' THEN dose_avg ELSE 0 END) AS
other_total,
   SUM(CASE WHEN ethnicity LIKE '%UNKNOWN%' THEN dose_avg ELSE 0 END) AS
unknown_total,
   SUM(CASE WHEN ethnicity LIKE '%UNABLE%' THEN dose_avg ELSE 0 END) AS
unable_total,
   SUM(CASE WHEN ethnicity LIKE '%AMERICAN INDIAN%' OR ethnicity LIKE
'%ALASKA%' THEN dose_avg ELSE 0 END) AS native_total
 FROM filtered
 GROUP BY drug
),

with_totals AS (
 SELECT *,
   asian_total + white_total + black_total + hispanic_total + other_total
+ unknown_total + unable_total + native_total AS total
 FROM agg
)

SELECT
 drug,
 asian_total,
 white_total,
 black_total,
 hispanic_total,
 other_total,
 unknown_total,
 unable_total,
 native_total,
 total,
 ROUND(asian_total / total * 100, 1) AS asian_pct,
 ROUND(white_total / total * 100, 1) AS white_pct,
 ROUND(black_total / total * 100, 1) AS black_pct,
 ROUND(hispanic_total / total * 100, 1) AS hispanic_pct,
 ROUND(other_total / total * 100, 1) AS other_pct,
 ROUND(unknown_total / total * 100, 1) AS unknown_pct,
 ROUND(unable_total / total * 100, 1) AS unable_pct,
 ROUND(native_total / total * 100, 1) AS native_pct
FROM with_totals
ORDER BY total DESC
LIMIT 50;


WITH cleaned AS (
 SELECT
   drug,
   ethnicity,
   lower(replace(dose_val_rx, ',', '')) AS dose_str
 FROM prescriptions
 JOIN admissions USING (hadm_id)
 WHERE dose_val_rx IS NOT NULL
),

parsed AS (
 SELECT
   drug,
   ethnicity,
   CASE
     WHEN dose_str ~ '^[0-9.]+-[0-9.]+$' THEN
       (CAST(split_part(dose_str, '-', 1) AS DOUBLE) +
CAST(split_part(dose_str, '-', 2) AS DOUBLE)) / 2
     WHEN dose_str ~ '^[0-9.]+$' THEN
       CAST(dose_str AS DOUBLE)
     ELSE NULL
   END AS dose_avg
 FROM cleaned
),

filtered AS (
 SELECT * FROM parsed WHERE dose_avg IS NOT NULL
),

aggregated AS (
 SELECT
```

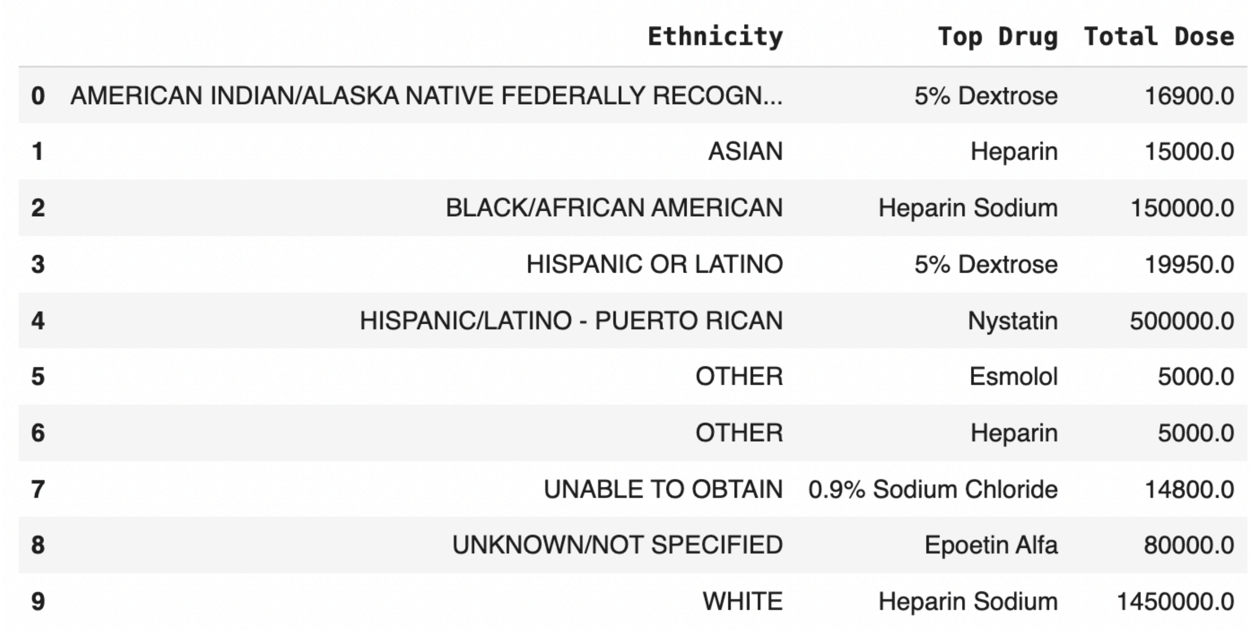
```
   ethnicity,
   drug,
   SUM(dose_avg) AS total_dose
 FROM filtered
 GROUP BY ethnicity, drug
),

ranked AS (
 SELECT *,
   RANK() OVER (PARTITION BY ethnicity ORDER BY total_dose DESC) AS rnk
 FROM aggregated
)

SELECT
 ethnicity AS "Ethnicity",
 drug AS "Top Drug",
 total_dose AS "Total Dose"
FROM ranked
WHERE rnk = 1
ORDER BY ethnicity;
```

First, I pulled from the CSVs to create tables for prescriptions and admissions that contain the relevant fields for this analysis: ethnicity, drug, and dose_val_rx. I cleaned the dose_val_rx column by removing commas and converting everything to lowercase, which helps standardize the format. For entries that contained ranges (like 10-20), I calculated the average of the two numbers. For single-value doses, I cast them directly to numeric form. Any values that didn’t match these patterns were excluded. After cleaning, I grouped the dataset by both drug and ethnicity and summed the average dose values. I created a table where each row is a drug and each column represents the total dose for an ethnicity. I then added a total column to represent the overall usage of the drug and calculated the percentage contribution of each ethnicity to that drug's total. This helps account for relative usage, not just absolute totals. I also made a second query that focuses on each ethnicity. I ranked all drugs by total dose per ethnicity and selected only the top-ranked drug for each group. This final table shows the top drug by usage for each ethnicity, along with the total dose they received.



index	drug	asian_total	white_total	black_total	hispanic_total	other_total	unknown_total	unable_total	native_total	total	asian_pct	wh
0	Heparin Sodium	0.0	1450000.0	150000.0	175000.0	0.0	25000.0	0.0	0.0	1800000.0	0.0	
1	Heparin	15000.0	469600.0	45200.0	51800.0	5000.0	15000.0	5000.0	15000.0	621600.0	2.4	
2	Nystatin	0.0	0.0	0.0	500000.0	0.0	0.0	0.0	0.0	500000.0	0.0	
3	Ny	0.0	500000.0	0.0	0.0	0.0	0.0	0.0	0.0	500000.0	0.0	
4	0.9% Sodium Chloride	0.0	131239.0	11355.0	54967.5	2000.0	1500.0	14800.0	13601.0	229462.5	0.0	
5	NS	7000.0	134200.0	5650.0	3200.0	4750.0	8100.0	0.0	100.0	163000.0	4.3	
6	Vitamin D	400.0	154800.0	800.0	400.0	0.0	0.0	0.0	0.0	156400.0	0.3	
7	D5W	11000.0	87485.0	11950.0	3250.0	1000.0	14100.0	1300.0	200.0	130285.0	8.4	
8	5% Dextrose	0.0	45510.0	5300.0	36050.0	1200.0	0.0	4250.0	16900.0	109210.0	0.0	
9	Vancomycin	0.0	60500.0	13500.0	27750.0	0.0	0.0	2000.0	4000.0	107750.0	0.0	
10	LR	0.0	74250.0	0.0	20000.0	0.0	0.0	1000.0	3000.0	98250.0	0.0	
11	Epoetin Alfa	0.0	4000.0	12000.0	0.0	0.0	80000.0	0.0	0.0	96000.0	0.0	
12	Acetaminophen	975.0	62700.0	2600.0	13037.5	1625.0	3737.5	1300.0	3250.0	89225.0	1.1	
13	Vancomycin HCl	7000.0	62250.0	10750.0	0.0	0.0	4500.0	0.0	0.0	84500.0	8.3	

Above shows the general table with every drug along with each ethnicity's drug usage.



	Ethnicity	Top Drug	Total Dose
0	AMERICAN INDIAN/ALASKA NATIVE FEDERALLY RECOGN...	5% Dextrose	16900.0
1	ASIAN	Heparin	15000.0
2	BLACK/AFRICAN AMERICAN	Heparin Sodium	150000.0
3	HISPANIC OR LATINO	5% Dextrose	19950.0
4	HISPANIC/LATINO - PUERTO RICAN	Nystatin	500000.0
5	OTHER	Esmolol	5000.0
6	OTHER	Heparin	5000.0
7	UNABLE TO OBTAIN	0.9% Sodium Chloride	14800.0
8	UNKNOWN/NOT SPECIFIED	Epoetin Alfa	80000.0
9	WHITE	Heparin Sodium	1450000.0

This is the polished table that shows an overview of each ethnicity's most used drug.

Heparin is the most common drug among all the races, with white patients receiving the most total dose (which makes sense, as in the US, white is the predominant race). Heparin sodium, a related product, is also common for black/African American patients. 5% dextrose is the most common drug for American Indian/Alaska native and Hispanic/Latinos.

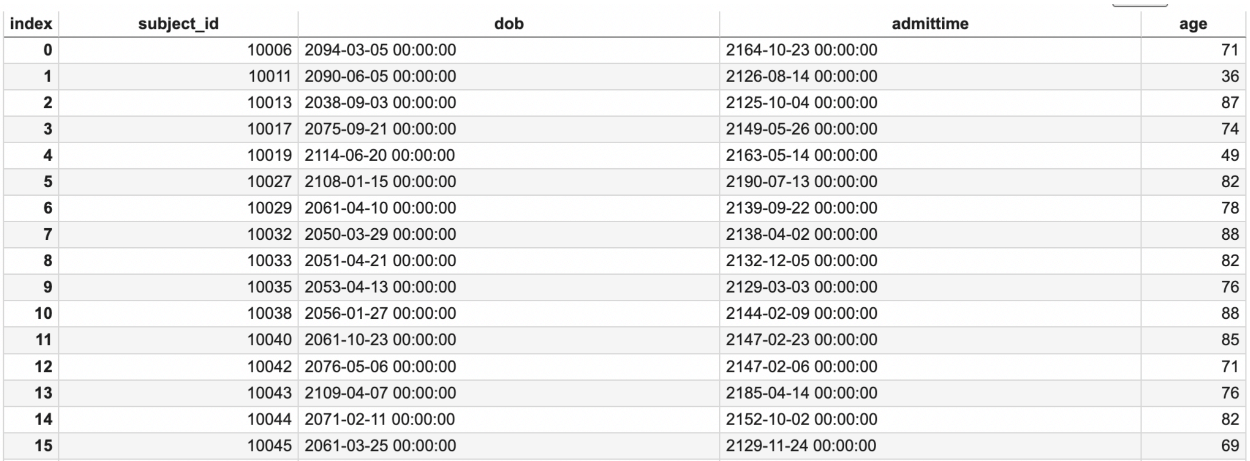
2.)

```
SELECT
 p.subject_id,
 p.dob,
 a.admittime,
 CAST(DATEDIFF('day', p.dob, a.admittime) / 365.25 AS INTEGER) AS age
FROM patients p
```

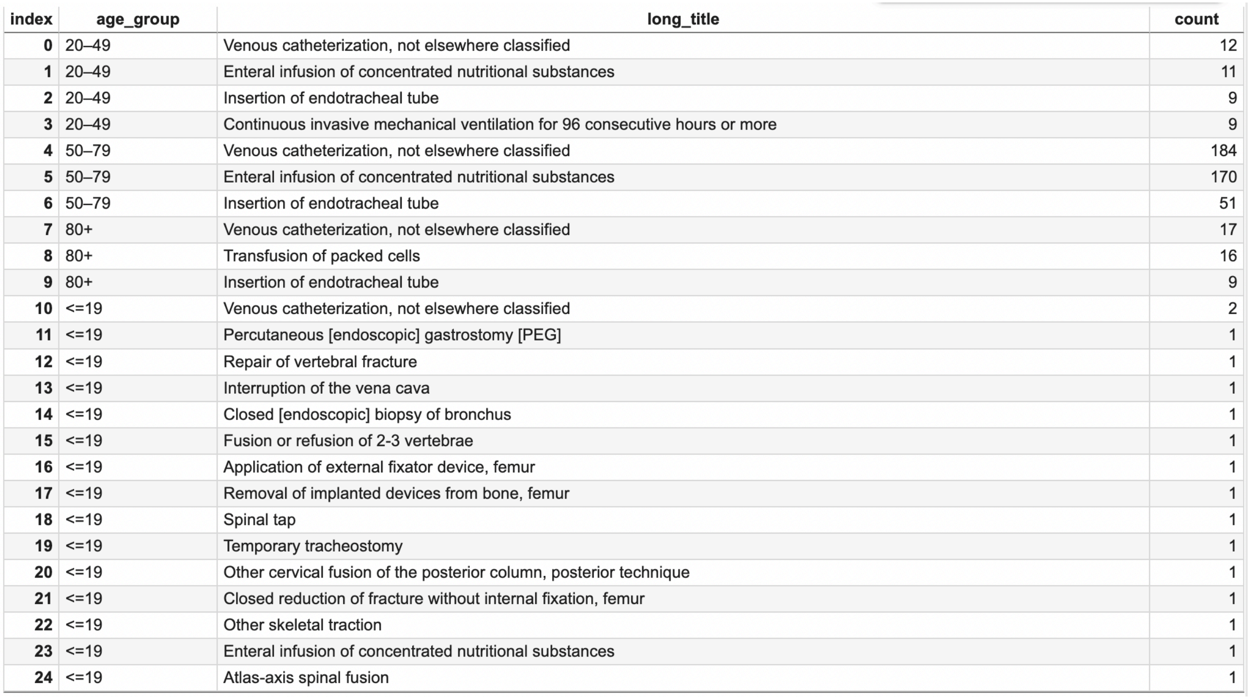
```
JOIN admissions a
 ON p.subject_id = a.subject_id
WHERE EXTRACT(YEAR FROM p.dob) >= 1900


WITH patients_with_age AS (
 SELECT
   p.subject_id,
   CAST(DATEDIFF('day', p.dob, a.admittime) / 365.25 AS INTEGER) AS age
 FROM patients p
 JOIN admissions a ON p.subject_id = a.subject_id
 WHERE EXTRACT(YEAR FROM p.dob) >= 1900
),
procedures_with_age AS (
 SELECT
   pr.subject_id,
   pr.icd9_code,
   CASE
     WHEN pa.age <= 19 THEN '<=19'
     WHEN pa.age BETWEEN 20 AND 49 THEN '20-49'
     WHEN pa.age BETWEEN 50 AND 79 THEN '50-79'
     ELSE '80+'
   END AS age_group
 FROM procedures_icd pr
 JOIN patients_with_age pa ON pr.subject_id = pa.subject_id
),
procedure_titles AS (
 SELECT
   pwa.age_group,
   d.long_title,
   COUNT(*) AS count
 FROM procedures_with_age pwa
 LEFT JOIN d_icd_procedures d ON pwa.icd9_code = d.icd9_code
 GROUP BY pwa.age_group, d.long_title
),
ranked AS (
 SELECT *,
        RANK() OVER (PARTITION BY age_group ORDER BY count DESC) AS rnk
 FROM procedure_titles
)
```

```
SELECT age_group, long_title, count
FROM ranked
WHERE rnk <= 3
ORDER BY age_group, count DESC;
```

First, I calculated each patient's age at the time of their hospital admission by subtracting their date of birth from their admission time and dividing by 365.25 to convert the difference into years. I filtered out any patients born before 1900 to avoid erroneous or incomplete data. This is because there were patients that were calculated to be over 200 years old, and all these patients had a logged date of birth of before 1900. Then, I joined this age information with the procedures table so that each procedure is tagged with the patient's corresponding age. I grouped the patients into age brackets: ≤19, 20–49, 50–79, and 80+. Next, I joined in the procedure titles from the d_icd_procedures reference table to make the output more readable. I grouped by both age group and procedure title to count how many times each procedure occurred within each age bracket. After that, I ranked the procedures within each age group based on how frequently they occurred. Finally, I selected the top three procedures for each age group and displayed them alongside the count of how many times each one appeared. Because there were some ties in the top three, the table contains more than 12 entries.



index	subject_id	dob	admittime	age
0	10006	2094-03-05 00:00:00	2164-10-23 00:00:00	71
1	10011	2090-06-05 00:00:00	2126-08-14 00:00:00	36
2	10013	2038-09-03 00:00:00	2125-10-04 00:00:00	87
3	10017	2075-09-21 00:00:00	2149-05-26 00:00:00	74
4	10019	2114-06-20 00:00:00	2163-05-14 00:00:00	49
5	10027	2108-01-15 00:00:00	2190-07-13 00:00:00	82
6	10029	2061-04-10 00:00:00	2139-09-22 00:00:00	78
7	10032	2050-03-29 00:00:00	2138-04-02 00:00:00	88
8	10033	2051-04-21 00:00:00	2132-12-05 00:00:00	82
9	10035	2053-04-13 00:00:00	2129-03-03 00:00:00	76
10	10038	2056-01-27 00:00:00	2144-02-09 00:00:00	88
11	10040	2061-10-23 00:00:00	2147-02-23 00:00:00	85
12	10042	2076-05-06 00:00:00	2147-02-06 00:00:00	71
13	10043	2109-04-07 00:00:00	2185-04-14 00:00:00	76
14	10044	2071-02-11 00:00:00	2152-10-02 00:00:00	82
15	10045	2061-03-25 00:00:00	2129-11-24 00:00:00	69



index	age_group	long_title	count
0	20–49	Venous catheterization, not elsewhere classified	12
1	20–49	Enteral infusion of concentrated nutritional substances	11
2	20–49	Insertion of endotracheal tube	9
3	20–49	Continuous invasive mechanical ventilation for 96 consecutive hours or more	9
4	50–79	Venous catheterization, not elsewhere classified	184
5	50–79	Enteral infusion of concentrated nutritional substances	170
6	50–79	Insertion of endotracheal tube	51
7	80+	Venous catheterization, not elsewhere classified	17
8	80+	Transfusion of packed cells	16
9	80+	Insertion of endotracheal tube	9
10	<=19	Venous catheterization, not elsewhere classified	2
11	<=19	Percutaneous [endoscopic] gastrostomy [PEG]	1
12	<=19	Repair of vertebral fracture	1
13	<=19	Interruption of the vena cava	1
14	<=19	Closed [endoscopic] biopsy of bronchus	1
15	<=19	Fusion or refusion of 2-3 vertebrae	1
16	<=19	Application of external fixator device, femur	1
17	<=19	Removal of implanted devices from bone, femur	1
18	<=19	Spinal tap	1
19	<=19	Temporary tracheostomy	1
20	<=19	Other cervical fusion of the posterior column, posterior technique	1
21	<=19	Closed reduction of fracture without internal fixation, femur	1
22	<=19	Other skeletal traction	1
23	<=19	Enteral infusion of concentrated nutritional substances	1
24	<=19	Atlas-axis spinal fusion	1

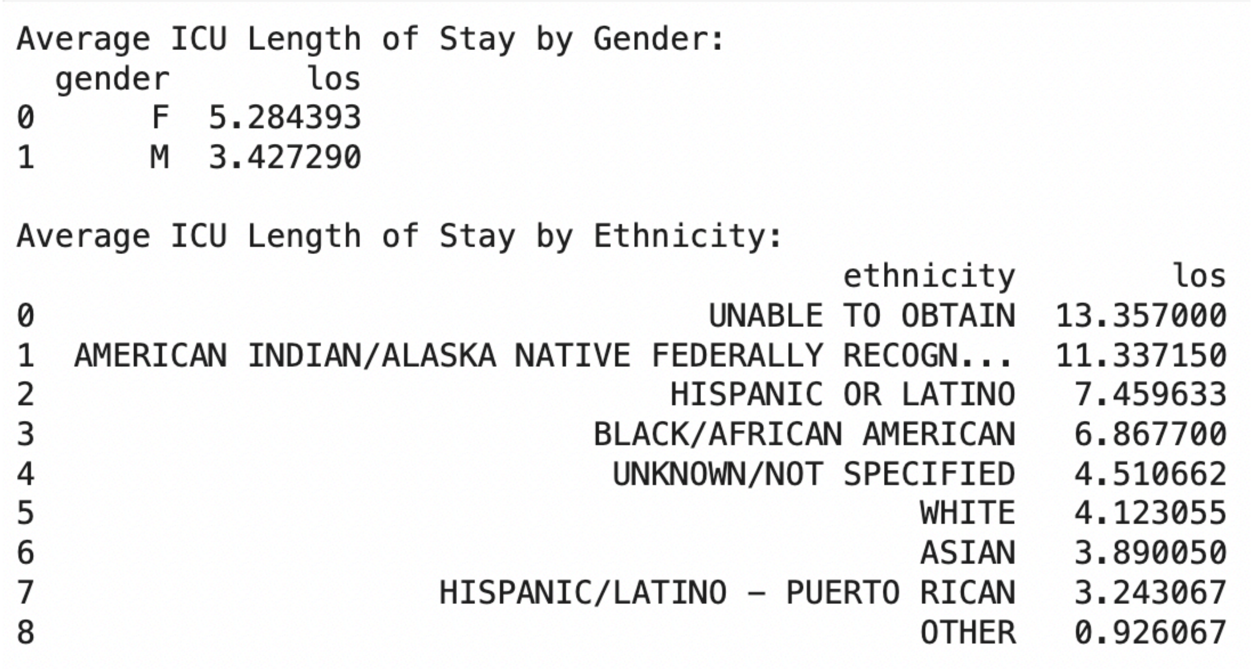
Venous catheterization is one of the most common procedures, and it was found among all age groups. It was the most common procedure for 19 and under, 20-49, and 50-79. Enteral infusion of nutritional substances becomes more and more common with age. The reason why this is probably not as common in the 80+ range is because people have died, so there are not enough people to have this procedure done to. The pediatric procedures also are much fewer in count than the rest, either due to low representation in the data set or lower procedure complexity.

3.)

```
SELECT
 p.gender,
 AVG(i.los) AS Average_LengthofStay
FROM icustays i
JOIN patients p ON i.subject_id = p.subject_id
WHERE i.los IS NOT NULL
GROUP BY p.gender



SELECT
 a.ethnicity,
 AVG(i.los) AS avg_los
FROM icustays i
JOIN admissions a ON i.subject_id = a.subject_id
```

```
WHERE i.los IS NOT NULL
GROUP BY a.ethnicity
ORDER BY avg_los DESC
```

First, I calculated the average ICU length of stay (LOS) for patients based on gender by joining the ICU stays table with the patients table. I filtered out any rows where LOS was missing to ensure accurate results, then grouped the data by gender and took the average for each group. Next, I did a similar analysis for ethnicity. I joined the ICU stays table with the admissions table to get each patient's ethnicity, again filtering out any rows with missing LOS. I grouped by ethnicity and calculated the average length of stay for each group, ordering the results from highest to lowest to see which ethnic groups had the longest ICU stays on average.



```
Average ICU Length of Stay by Gender:
  gender       los
0      F  5.284393
1      M  3.427290

Average ICU Length of Stay by Ethnicity:
                                           ethnicity        los
0                                   UNABLE TO OBTAIN  13.357000
1  AMERICAN INDIAN/ALASKA NATIVE FEDERALLY RECOGN...  11.337150
2                                 HISPANIC OR LATINO   7.459633
3                             BLACK/AFRICAN AMERICAN   6.867700
4                             UNKNOWN/NOT SPECIFIED   4.510662
5                                              WHITE   4.123055
6                                              ASIAN   3.890050
7                     HISPANIC/LATINO - PUERTO RICAN   3.243067
8                                              OTHER   0.926067
```

This dataset suggests that females on average spend about 2 more days than males in the ICU. This could be related to gender-related clinical decision-making, such as pregnancies/giving birth. American Indians/Alaska natives stayed in the ICU for the longest time on average while Puerto Ricans stayed in the ICU for the shortest time. Something of note is that the "ethnicity" with the longest stay were "unable to obtain," maybe suggesting that the most vulnerable patients are not being properly documented although this is something that would have to be investigated deeper. This also suggests that there might be some health disparities in critical care between ethnicities, with some groups staying longer in the ICU. Further analysis would have to be done to examine this relationship.

Part II (Non-Relational)

1.)

part a)

```
session.execute("""
CREATE TABLE IF NOT EXISTS drug_usage_by_ethnicity (
    ethnicity text,
    drug text,
    total_dosage double,
    PRIMARY KEY ((ethnicity), drug)
)
""")
```

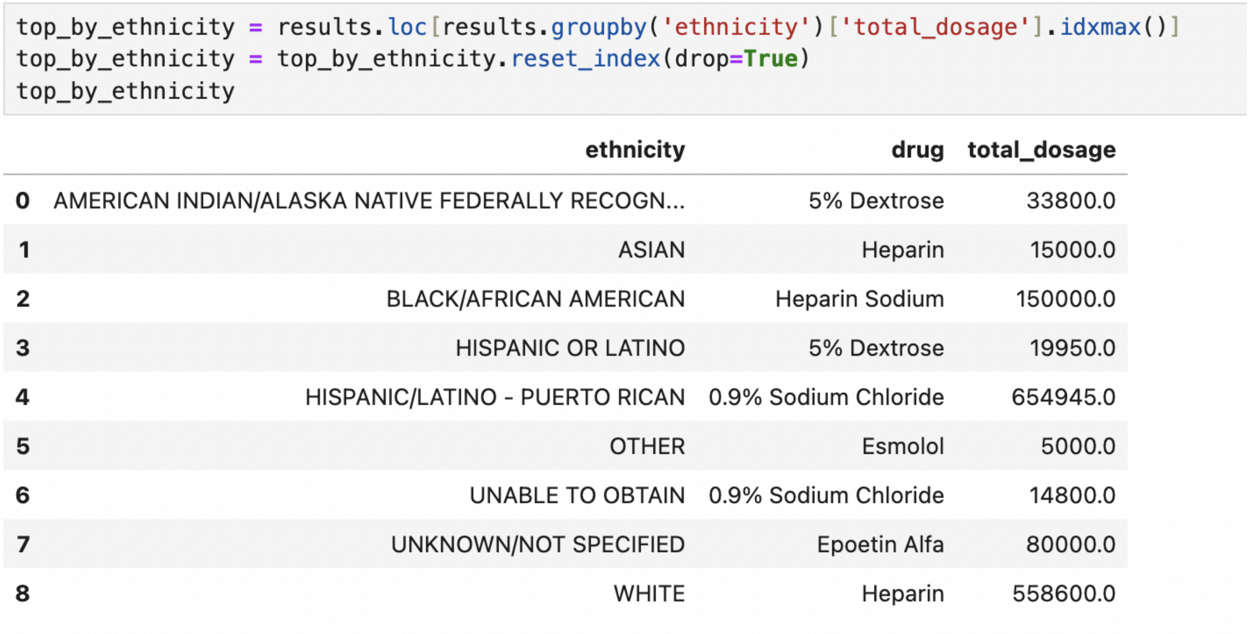
part b)

```
from cassandra import ConsistencyLevel

insert_query = session.prepare("""
    INSERT INTO drug_usage_by_ethnicity (ethnicity, drug, total_dosage)
    VALUES (?, ?, ?)
""")

for _, row in summary.iterrows():
    bound = insert_query.bind((row['ethnicity'], row['drug'], float(row['total_dosage'])))
    bound.consistency_level = ConsistencyLevel.LOCAL_QUORUM
    session.execute(bound)
```

part c)

```
rows = session.execute("SELECT * FROM drug_usage_by_ethnicity")
results = pd.DataFrame(rows.all())

# Get top drug per ethnicity by total dosage
top_by_ethnicity = results.loc[results.groupby('ethnicity')['total_dosage'].idxmax()]
top_by_ethnicity = top_by_ethnicity.reset_index(drop=True)
top_by_ethnicity
```

part d)



```
top_by_ethnicity = results.loc[results.groupby('ethnicity')['total_dosage'].idxmax()]
top_by_ethnicity = top_by_ethnicity.reset_index(drop=True)
top_by_ethnicity
```

	ethnicity	drug	total_dosage
0	AMERICAN INDIAN/ALASKA NATIVE FEDERALLY RECOGN...	5% Dextrose	33800.0
1	ASIAN	Heparin	15000.0
2	BLACK/AFRICAN AMERICAN	Heparin Sodium	150000.0
3	HISPANIC OR LATINO	5% Dextrose	19950.0
4	HISPANIC/LATINO - PUERTO RICAN	0.9% Sodium Chloride	654945.0
5	OTHER	Esmolol	5000.0
6	UNABLE TO OBTAIN	0.9% Sodium Chloride	14800.0
7	UNKNOWN/NOT SPECIFIED	Epoetin Alfa	80000.0
8	WHITE	Heparin	558600.0

2.)

part a)

```
session.execute("""
CREATE TABLE IF NOT EXISTS de300.patient_age_info (
    subject_id text,
    dob date,
    admittime timestamp,
    age int,
    age_group text,
    PRIMARY KEY (subject_id)
);
""")
```

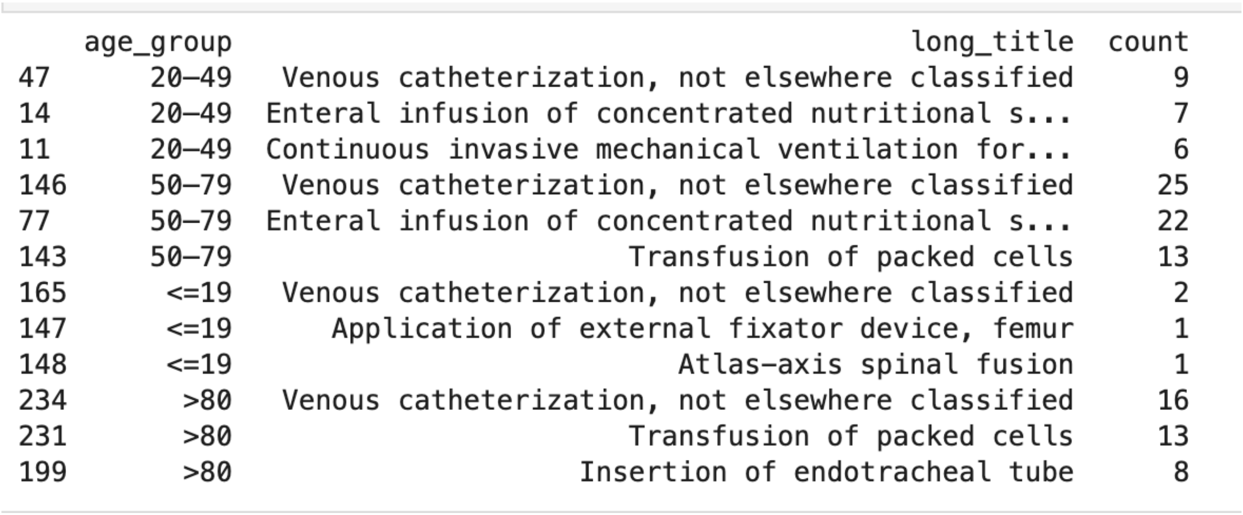
part b)

```
import pandas as pd

patients = pd.read_csv('PATIENTS.csv')
admissions = pd.read_csv('ADMISSIONS.csv')
procedures_icd = pd.read_csv('PROCEDURES_ICD.csv')
d_icd_procedures = pd.read_csv('D_ICD_PROCEDURES.csv')

result = pd.read_csv('result.csv')
result['dob'] = pd.to_datetime(result['dob'])
result['admittime'] = pd.to_datetime(result['admittime'])
def assign_age_group(age):
    if age <= 19:
        return '<=19'
    elif age <= 49:
        return '20–49'
    elif age <= 79:
        return '50–79'
    else:
        return '>80'

result['age_group'] = result['age'].apply(assign_age_group)
insert_query = session.prepare("""
    INSERT INTO de300.patient_age_info (subject_id, dob, admittime, age, age_group)
    VALUES (?, ?, ?, ?, ?)
""")
result['dob'] = pd.to_datetime(result['dob'])
result['admittime'] = pd.to_datetime(result['admittime'])
result['age'] = ((result['admittime'] - result['dob']).dt.days / 365.25).astype(int)
from cassandra import ConsistencyLevel

for _, row in result.iterrows():
    bound = insert_query.bind((
        str(row['subject_id']),
        row['dob'].date(),
        row['admittime'].to_pydatetime(),
        int(row['age']),
        row['age_group']
    ))
    bound.consistency_level = ConsistencyLevel.LOCAL_QUORUM
    session.execute(bound)
```

part c)

```
rows = session.execute("SELECT subject_id, age_group FROM de300.patient_age_info;")
patient_df = pd.DataFrame(rows)
procedures_icd['subject_id'] = procedures_icd['subject_id'].astype(str)
patient_df['subject_id'] = patient_df['subject_id'].astype(str)
merged = procedures_icd.merge(patient_df, on='subject_id', how='inner')
full = merged.merge(d_icd_procedures[['icd9_code', 'long_title']], on='icd9_code', how='left')
grouped = full.groupby(['age_group', 'long_title']).size().reset_index(name='count')
grouped['rank'] = grouped.groupby('age_group')['count'].rank(method='first', ascending=False)
top3 = grouped[grouped['rank'] <= 3].drop(columns='rank')
print(top3.sort_values(['age_group', 'count'], ascending=[True, False]))
```

part d)



```
    age_group                                          long_title  count
47      20–49   Venous catheterization, not elsewhere classified      9
14      20–49  Enteral infusion of concentrated nutritional s...      7
11      20–49  Continuous invasive mechanical ventilation for...      6
146     50–79   Venous catheterization, not elsewhere classified     25
77      50–79  Enteral infusion of concentrated nutritional s...     22
143     50–79                         Transfusion of packed cells     13
165      <=19   Venous catheterization, not elsewhere classified      2
147      <=19      Application of external fixator device, femur      1
148      <=19                            Atlas–axis spinal fusion      1
234       >80   Venous catheterization, not elsewhere classified     16
231       >80                         Transfusion of packed cells     13
199       >80                      Insertion of endotracheal tube      8
```

3.)

part a)

```
session.execute("""
CREATE TABLE IF NOT EXISTS de300.icu_stay_info (
   subject_id text,
   gender text,
   ethnicity text,
   los float,
   PRIMARY KEY (subject_id)
);
""")
```

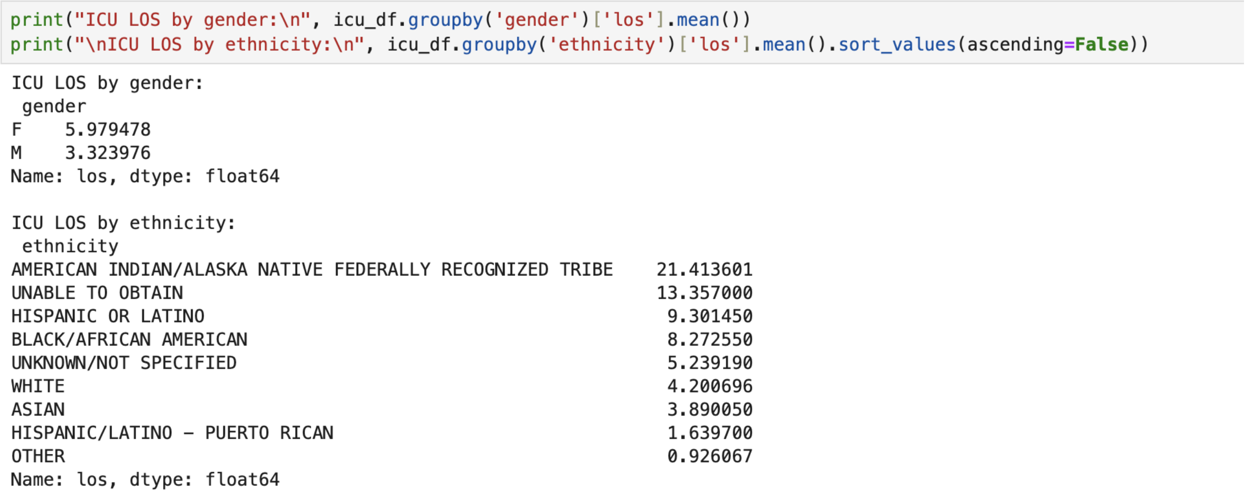
part b)

```
import pandas as pd

icustays = pd.read_csv('ICUSTAYS.csv')
patients = pd.read_csv('PATIENTS.csv')
admissions = pd.read_csv('ADMISSIONS.csv')
icu = icustays.merge(patients[['subject_id', 'gender']], on='subject_id')
icu = icu.merge(admissions[['subject_id', 'ethnicity']], on='subject_id')
icu = icu[['subject_id', 'gender', 'ethnicity', 'los']]
icu['subject_id'] = icu['subject_id'].astype(str)
from cassandra import ConsistencyLevel

insert_query = session.prepare("""
   INSERT INTO de300.icu_stay_info (subject_id, gender, ethnicity, los)
   VALUES (?, ?, ?, ?)
""")

for _, row in icu.iterrows():
   bound = insert_query.bind((
       row['subject_id'],
       row['gender'],
       row['ethnicity'],
       float(row['los'])
   ))
   bound.consistency_level = ConsistencyLevel.LOCAL_QUORUM
   session.execute(bound)
```

part c)

```
rows = session.execute("SELECT subject_id, gender, ethnicity, los FROM
de300.icu_stay_info;")
icu_df = pd.DataFrame(rows)
icu_df.groupby('gender')['los'].mean()
icu_df.groupby('ethnicity')['los'].mean().sort_values(ascending=False)
```

part d)



```
print("ICU LOS by gender:\n", icu_df.groupby('gender')['los'].mean())
print("\nICU LOS by ethnicity:\n", icu_df.groupby('ethnicity')['los'].mean().sort_values(ascending=False))
```

```
ICU LOS by gender:
 gender
F    5.979478
M    3.323976
Name: los, dtype: float64

ICU LOS by ethnicity:
 ethnicity
AMERICAN INDIAN/ALASKA NATIVE FEDERALLY RECOGNIZED TRIBE    21.413601
UNABLE TO OBTAIN                                            13.357000
HISPANIC OR LATINO                                           9.301450
BLACK/AFRICAN AMERICAN                                       8.272550
UNKNOWN/NOT SPECIFIED                                        5.239190
WHITE                                                        4.200696
ASIAN                                                        3.890050
HISPANIC/LATINO - PUERTO RICAN                               1.639700
OTHER                                                        0.926067
Name: los, dtype: float64
```

For this assignment, generative AI was used mainly for the purposes of understanding which functions can be used to achieve a certain action. If there wasn’t a clear way to do it with the limited amount of computer science I have taken, then it was used to help find code to do so.