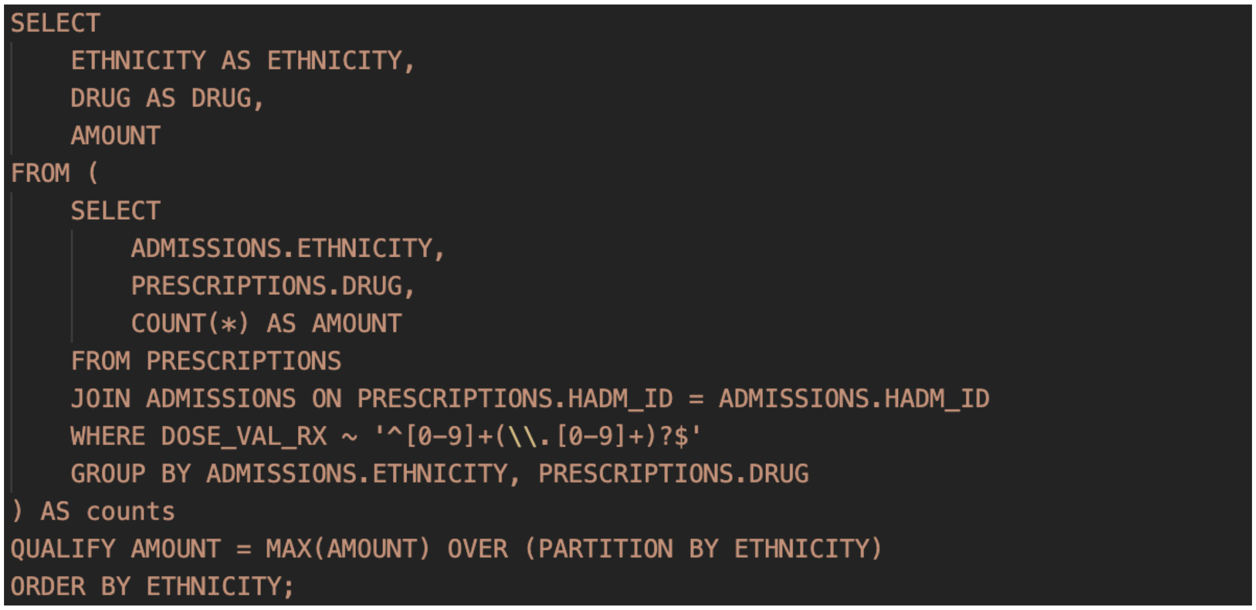
Data Engineering 300
Homework 2 Write Up
Sabian Atmadja (sna3362)
Analysis Questions

1. **Create a summary of type of drugs and their total amount used by ethnicity. Report the top usage in each ethnicity group.** ***You may have to make certain assumptions in calculating their total amount.***

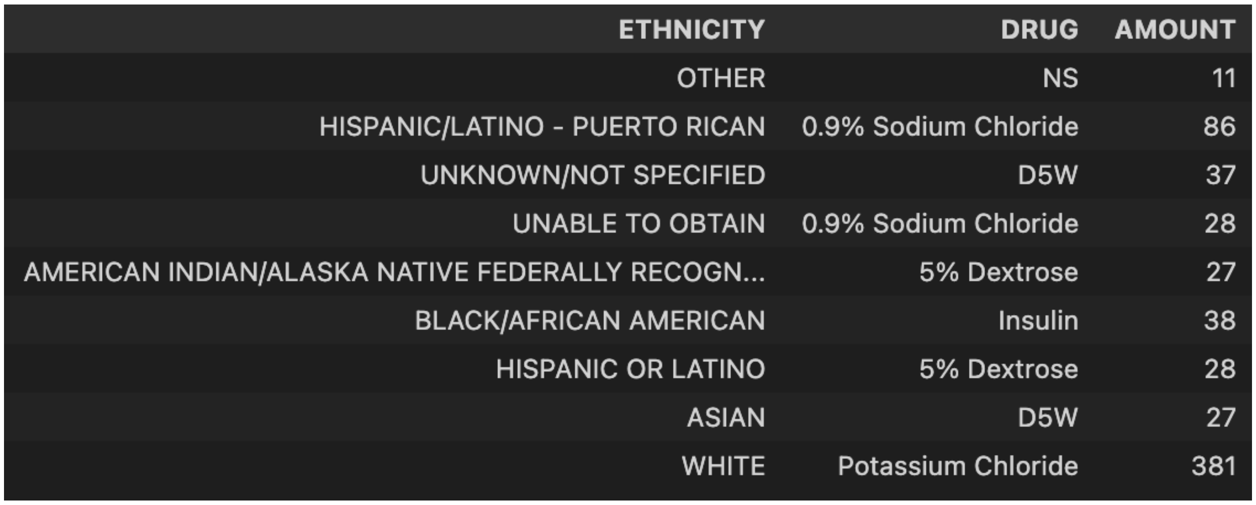
The SQL query is found below:



```
SELECT
    ETHNICITY AS ETHNICITY,
    DRUG AS DRUG,
    AMOUNT
FROM (
    SELECT
        ADMISSIONS.ETHNICITY,
        PRESCRIPTIONS.DRUG,
        COUNT(*) AS AMOUNT
    FROM PRESCRIPTIONS
    JOIN ADMISSIONS ON PRESCRIPTIONS.HADM_ID = ADMISSIONS.HADM_ID
    WHERE DOSE_VAL_RX ~ '^[0-9]+(\\.[0-9]+)?$'
    GROUP BY ADMISSIONS.ETHNICITY, PRESCRIPTIONS.DRUG
) AS counts
QUALIFY AMOUNT = MAX(AMOUNT) OVER (PARTITION BY ETHNICITY)
ORDER BY ETHNICITY;
```

Figure 1: SQL Query for First Problem

This query identifies the most frequently prescribed drug for each ethnic group. It begins by creating a sub-table that joins the PRESCRIPTIONS and ADMISSIONS tables, grouping the data by ethnicity and drug type. For each group, it counts how many times each drug was prescribed, treating this count as the total "amount" prescribed. From this subtable, the main query uses a window function to find the maximum prescription count within each ethnicity group. The QUALIFY clause then filters the results to only include rows where the amount is equal to the maximum value for that ethnicity. In other words, it selects the drug(s) most commonly prescribed to each ethnic group. The final output shows the ethnicity, the drug, and the number of times it was prescribed. The results can be found below:

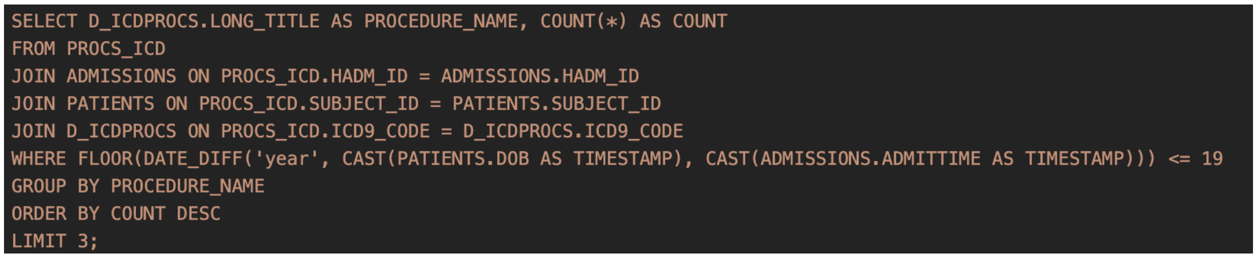


ETHNICITY	DRUG	AMOUNT
OTHER	NS	11
HISPANIC/LATINO - PUERTO RICAN	0.9% Sodium Chloride	86
UNKNOWN/NOT SPECIFIED	D5W	37
UNABLE TO OBTAIN	0.9% Sodium Chloride	28
AMERICAN INDIAN/ALASKA NATIVE FEDERALLY RECOGN...	5% Dextrose	27
BLACK/AFRICAN AMERICAN	Insulin	38
HISPANIC OR LATINO	5% Dextrose	28
ASIAN	D5W	27
WHITE	Potassium Chloride	381

Figure 2: Results table for ethnicities and most often prescribed drugs

The table shows the most commonly given drug for each ethnicity along with how many times it was given. Here we can see a clear discrepancy between different ethnic groups and the amount of times a drug is prescribed. For White patients, Potassium Chloride was given the most, with a count of 381 which is much higher than any other group. For Black/African American patients, the most common drug was Insulin (38), while Asian and Hispanic patients most often received D5W or 5% Dextrose. Hispanic/Latino – Puerto Rican and "Unable to Obtain" groups received 0.9% Sodium Chloride the most. The counts are much lower for all other groups compared to the White patient group. This suggests that the frequency and type of drugs given varies across different ethnicities in this dataset. Overall, because of the sheer amount of times Potassium Chloride is prescribed to White individuals, it dominates the count as the most frequently prescribed drug overall.

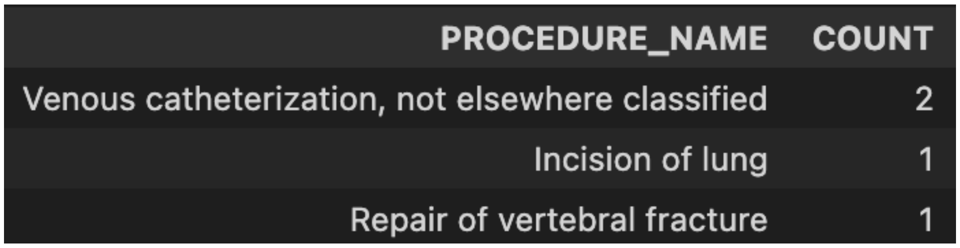
2. **Create a summary of procedures performed on patients by age groups (<=19, 20-49, 50-79, >80). Report the top three procedures, along with the name of the procedures, performed in each age group.**



```
SELECT D_ICDPROCS.LONG_TITLE AS PROCEDURE_NAME, COUNT(*) AS COUNT
FROM PROCS_ICD
JOIN ADMISSIONS ON PROCS_ICD.HADM_ID = ADMISSIONS.HADM_ID
JOIN PATIENTS ON PROCS_ICD.SUBJECT_ID = PATIENTS.SUBJECT_ID
JOIN D_ICDPROCS ON PROCS_ICD.ICD9_CODE = D_ICDPROCS.ICD9_CODE
WHERE FLOOR(DATE_DIFF('year', CAST(PATIENTS.DOB AS TIMESTAMP), CAST(ADMISSIONS.ADMITTIME AS TIMESTAMP))) <= 19
GROUP BY PROCEDURE_NAME
ORDER BY COUNT DESC
LIMIT 3;
```

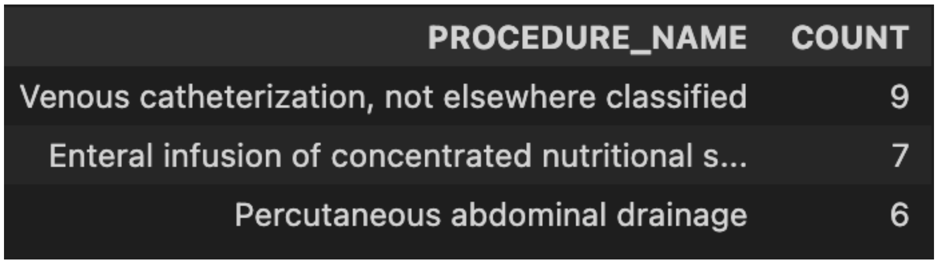
Figure 3: SQL Query for second problem for most common procedures patients under 19

The following query was used, with the exception of the seventh line where the age limits were adjusted depending on the age group being analyzed. The query joins four tables: the procedures table (PROCS_ICD), admissions data (ADMISSIONS), patient demographics (PATIENTS), and a reference table for procedure names (D_ICDPROCS). These joins allow us to match procedure codes with their descriptions and associate each procedure with the patient's age at the time of admission. The patient's age is calculated using the difference in years between their date of birth and admission time. The query then groups the data by procedure name and counts how many times each procedure occurred within a specific age group. Finally, it returns the top three most commonly performed procedures for that age group, sorted by frequency.



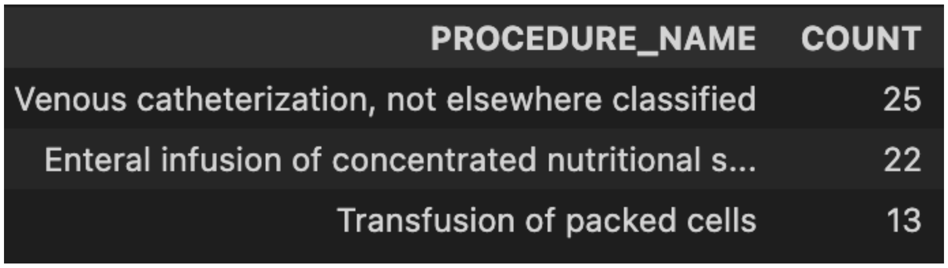
PROCEDURE_NAME	COUNT
Venous catheterization, not elsewhere classified	2
Incision of lung	1
Repair of vertebral fracture	1

Figure 4: Most common procedures for patients under the age of 19



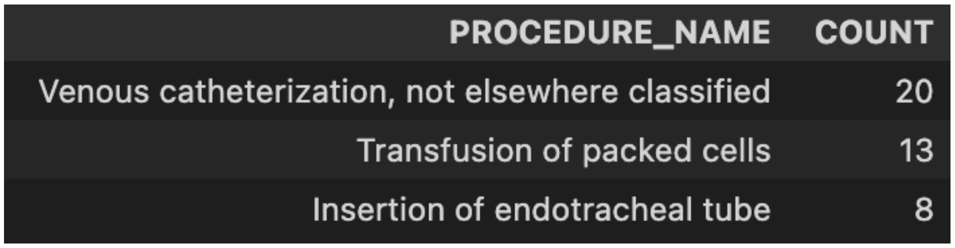
PROCEDURE_NAME	COUNT
Venous catheterization, not elsewhere classified	9
Enteral infusion of concentrated nutritional s...	7
Percutaneous abdominal drainage	6

Figure 5: Most common procedures for patients between 20 and 49



PROCEDURE_NAME	COUNT
Venous catheterization, not elsewhere classified	25
Enteral infusion of concentrated nutritional s...	22
Transfusion of packed cells	13

Figure 6: Most common procedures for patients between 50 and 79

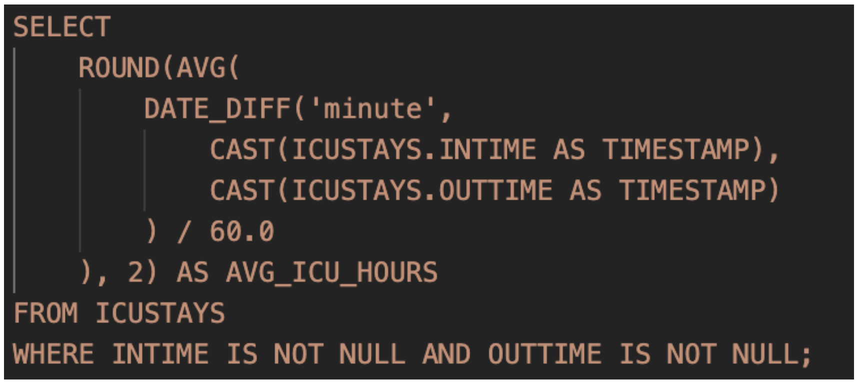


PROCEDURE_NAME	COUNT
Venous catheterization, not elsewhere classified	20
Transfusion of packed cells	13
Insertion of endotracheal tube	8

Figure 7: Most common procedures for patients over age of 80

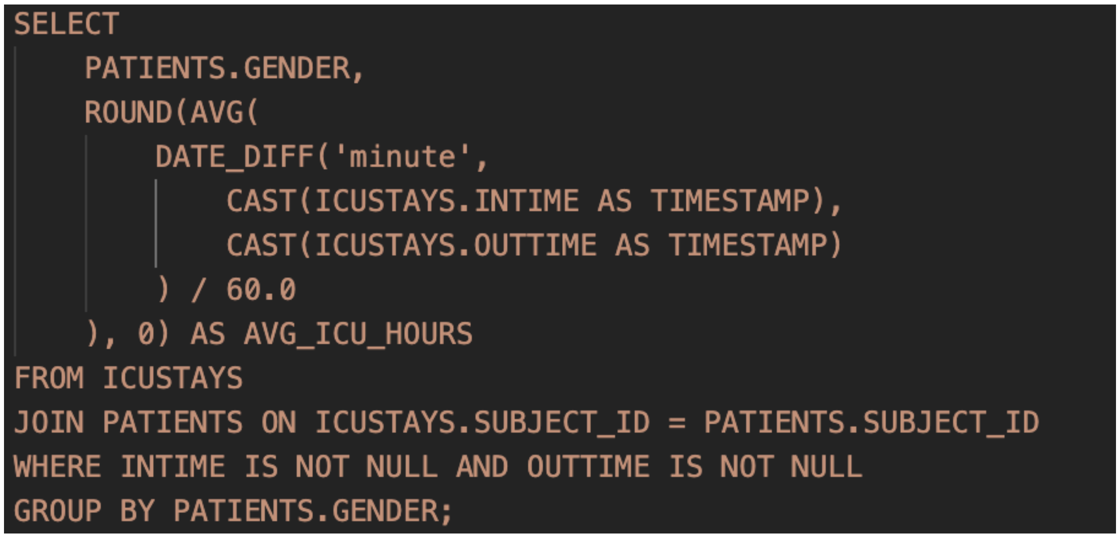
Venous catheterization was the most common procedure across all age groups, suggesting it is a widely used intervention in inpatient care regardless of age. Transfusion of packed cells and insertion of endotracheal tubes were also frequent, especially in patients over 50, reflecting more intensive care needs in older populations. The youngest group (≤19) had significantly fewer procedures overall, possibly due to smaller sample size or less frequent hospitalization.

3. **How long do patients stay in the ICU? Is there a difference in the ICU length of stay among gender or ethnicity?**



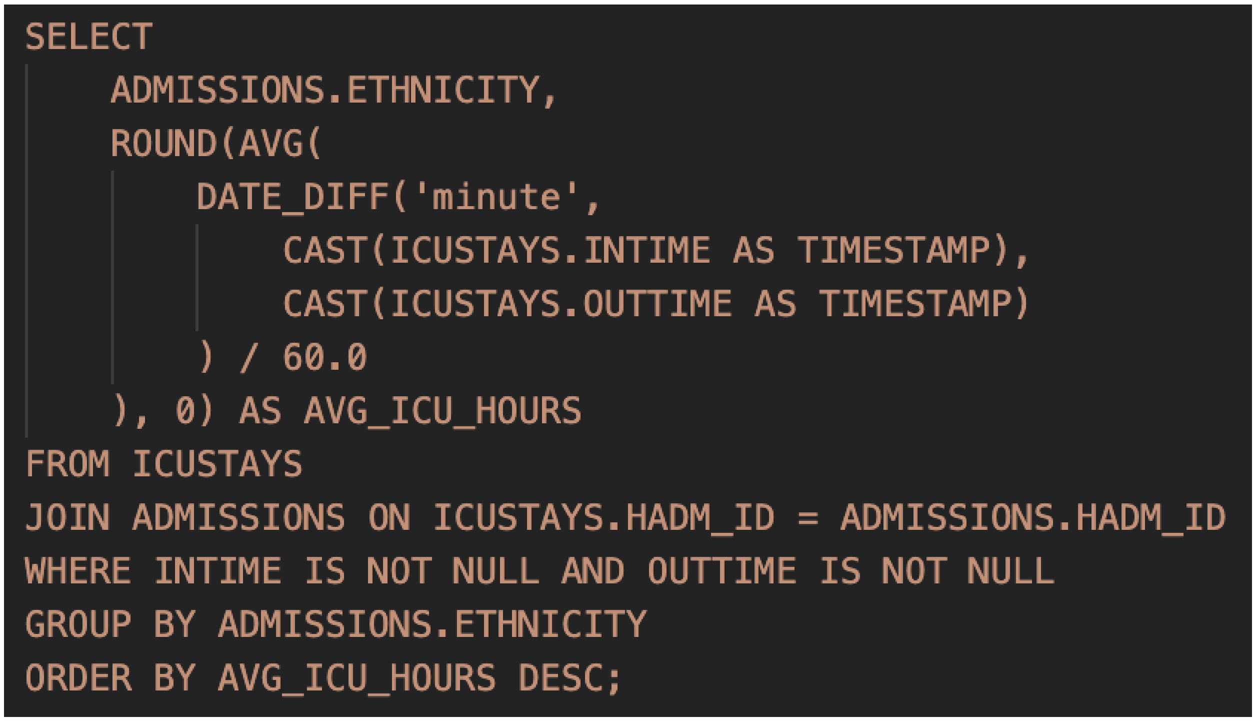
```
SELECT
    ROUND(AVG(
        DATE_DIFF('minute',
            CAST(ICUSTAYS.INTIME AS TIMESTAMP),
            CAST(ICUSTAYS.OUTTIME AS TIMESTAMP)
        ) / 60.0
    ), 2) AS AVG_ICU_HOURS
FROM ICUSTAYS
WHERE INTIME IS NOT NULL AND OUTTIME IS NOT NULL;
```

Figure 8: SQL query for average ICU stay for all patients



```
SELECT
    PATIENTS.GENDER,
    ROUND(AVG(
        DATE_DIFF('minute',
            CAST(ICUSTAYS.INTIME AS TIMESTAMP),
            CAST(ICUSTAYS.OUTTIME AS TIMESTAMP)
        ) / 60.0
    ), 0) AS AVG_ICU_HOURS
FROM ICUSTAYS
JOIN PATIENTS ON ICUSTAYS.SUBJECT_ID = PATIENTS.SUBJECT_ID
WHERE INTIME IS NOT NULL AND OUTTIME IS NOT NULL
GROUP BY PATIENTS.GENDER;
```

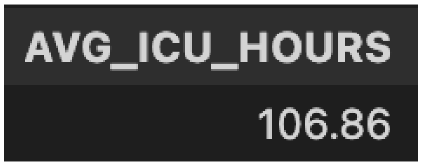
Figure 9: SQL query for average ICU stay by gender



```
SELECT
    ADMISSIONS.ETHNICITY,
    ROUND(AVG(
        DATE_DIFF('minute',
            CAST(ICUSTAYS.INTIME AS TIMESTAMP),
            CAST(ICUSTAYS.OUTTIME AS TIMESTAMP)
        ) / 60.0
    ), 0) AS AVG_ICU_HOURS
FROM ICUSTAYS
JOIN ADMISSIONS ON ICUSTAYS.HADM_ID = ADMISSIONS.HADM_ID
WHERE INTIME IS NOT NULL AND OUTTIME IS NOT NULL
GROUP BY ADMISSIONS.ETHNICITY
ORDER BY AVG_ICU_HOURS DESC;
```

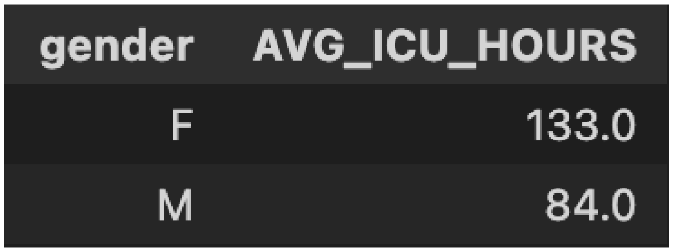
Figure 10: SQL query for average ICU stay by ethnicity

The first query calculates the overall average ICU stay by taking the difference in minutes between ICU admission (INTIME) and discharge (OUTTIME), converting it to hours, and averaging across all patients. The second query joins the ICUSTAYS table with the PATIENTS table to compare average ICU duration by gender. The third query joins ICUSTAYS with ADMISSIONS to group by patient ethnicity and compute the average length of stay for each group. In all cases, we exclude ICU stays with missing time values to ensure accurate calculations.



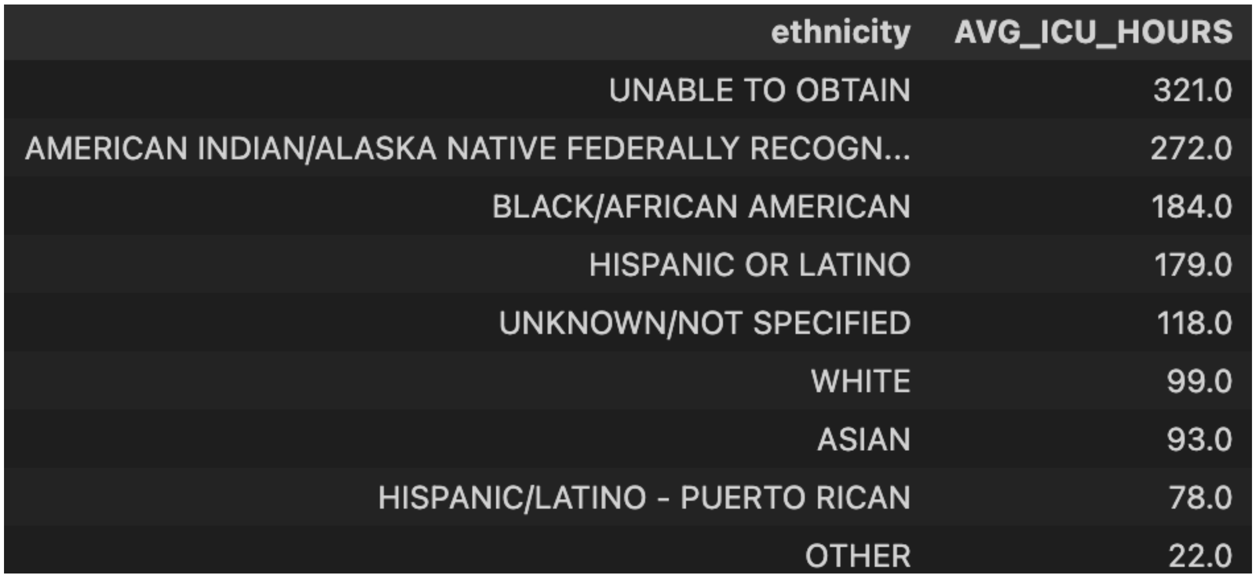
AVG_ICU_HOURS
106.86

Figure 11: Average ICU stay for all patients



gender	AVG_ICU_HOURS
F	133.0
M	84.0

Figure 12: Average ICU stay by gender



ethnicity	AVG_ICU_HOURS
UNABLE TO OBTAIN	321.0
AMERICAN INDIAN/ALASKA NATIVE FEDERALLY RECOGN...	272.0
BLACK/AFRICAN AMERICAN	184.0
HISPANIC OR LATINO	179.0
UNKNOWN/NOT SPECIFIED	118.0
WHITE	99.0
ASIAN	93.0
HISPANIC/LATINO - PUERTO RICAN	78.0
OTHER	22.0

Figure 13: Average ICU stay by ethnicity

The results show that the average ICU stay across all patients is 106.86 hours, or roughly 4.5 days. However, there are notable differences when broken down by gender and ethnicity. Female patients had a significantly higher average ICU stay (133 hours) compared to male patients (84 hours). Ethnic differences were even more pronounced: patients whose ethnicity was listed as "Unable to Obtain" had the longest average stay (321 hours), followed by American Indian/Alaska Native patients (272 hours) and Black/African American patients (184 hours). On the shorter end, Hispanic/Latino – Puerto Rican and Other groups had average stays of 78 and 22 hours, respectively. These differences may reflect variations in care needs, disease severity, or access to resources across demographic groups.

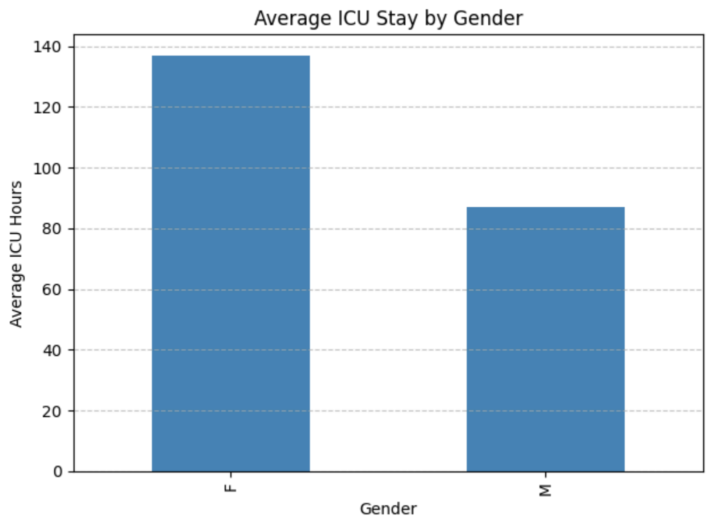


Figure 14: Average ICU stay by gender graph

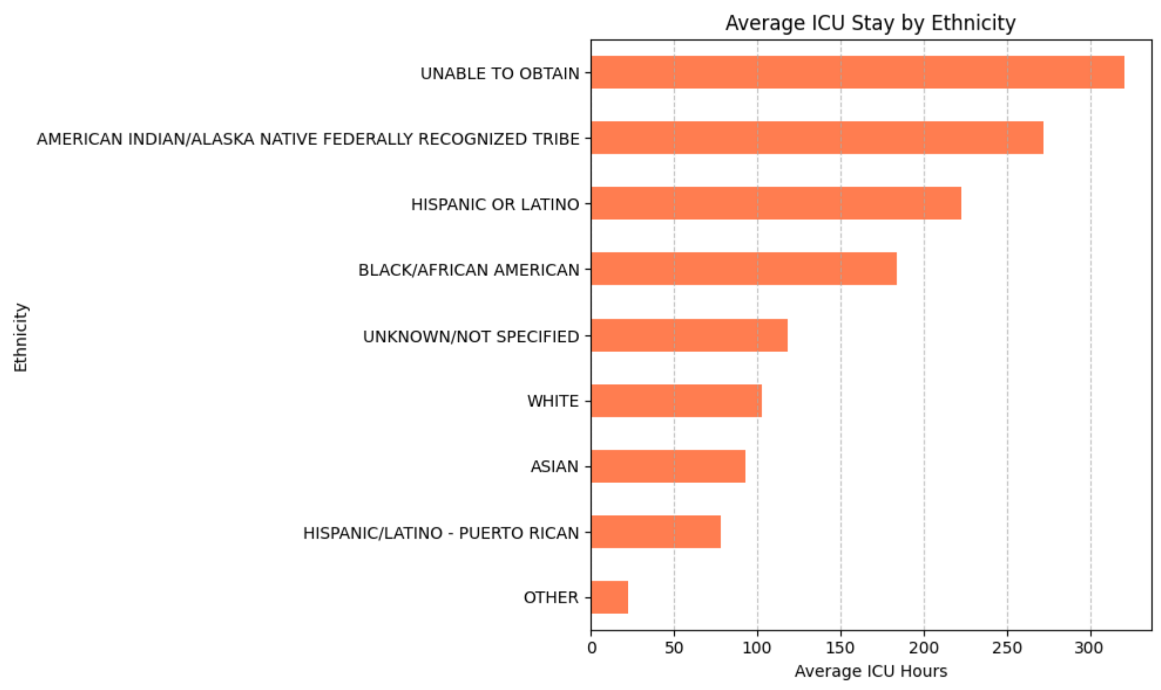


Figure 15: Average ICU stay by ethnicity graph

Cassandra Queries

1. **Drug Use by Ethnicity**

Table Creation:

```
session.set_keyspace('de300_sabian_hw2_part2')

session.execute("""
CREATE TABLE IF NOT EXISTS drug_ethnicity_hadm_3 (
   PRE_ID INT,
   ADM_ID INT,
   Ethnicity TEXT,
   Drug text,
   PRIMARY KEY ((ethnicity), PRE_ID, ADM_ID)
);
""")
```

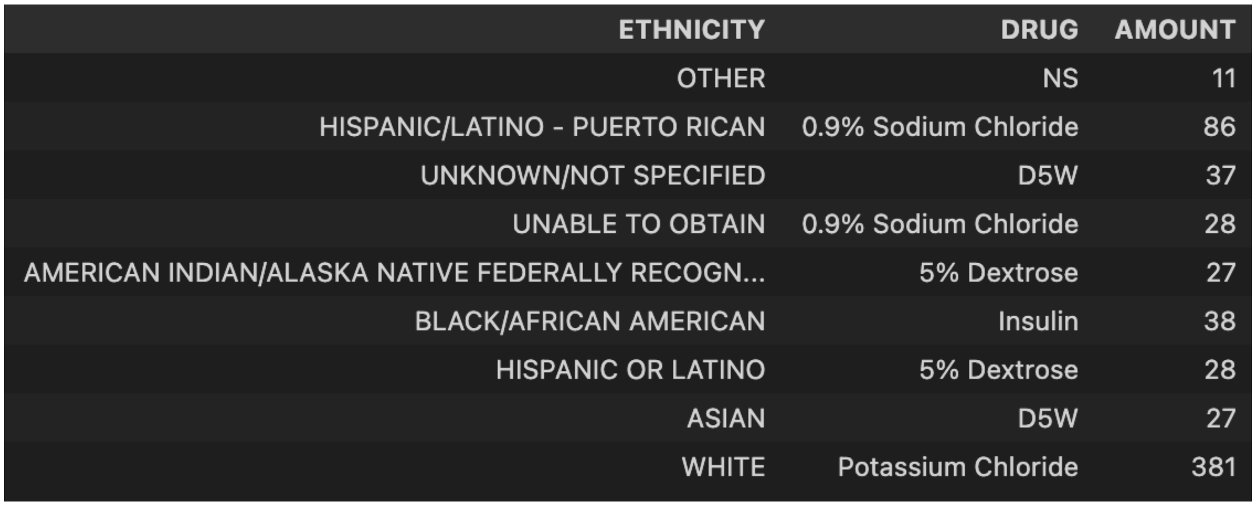
Data Upload:

```
session.set_keyspace('de300_sabian_hw2_part2')

insert_adm = session.prepare("""
   INSERT INTO drug_ethnicity_hadm_3 (PRE_ID, ADM_ID, Ethnicity, Drug)
   VALUES (?, ?, ?, ?)
""")

for _, row in df_top_drugtype.iterrows():
   session.execute(insert_adm, (
       int(row['PRE_ID']),
       int(row['ADM_ID']),
       str(row['ethnicity']),
       str(row['drug'])
   ))
```

```
rows = session.execute("SELECT ethnicity FROM drug_ethnicity_hadm")
ethnicity_list = list(set(row.ethnicity for row in rows))

records = []

for eth in ethnicity_list:
   rows = session.execute("""
       SELECT drug FROM drug_ethnicity_hadm WHERE ethnicity = %s
   """, [eth])

   # Convert rows to DataFrame
   df_eth = pd.DataFrame(rows)


   # Group by drug and count
   df_grouped = (
       df_eth.groupby('drug')
       .size()
       .reset_index(name='amount')
       .sort_values('amount', ascending=False)
   )

   # Select top row (most frequent drug)
   top_row = df_grouped.iloc[0]
   records.append({
       'ETHNICITY': eth,
       'DRUG': top_row['drug'],
       'AMOUNT': top_row['amount']
   })
```

```
# Final result
df_top_drugs = pd.DataFrame(records)

df_top_drugs
```

Verify same output:



ETHNICITY	DRUG	AMOUNT
OTHER	NS	11
HISPANIC/LATINO - PUERTO RICAN	0.9% Sodium Chloride	86
UNKNOWN/NOT SPECIFIED	D5W	37
UNABLE TO OBTAIN	0.9% Sodium Chloride	28
AMERICAN INDIAN/ALASKA NATIVE FEDERALLY RECOGN...	5% Dextrose	27
BLACK/AFRICAN AMERICAN	Insulin	38
HISPANIC OR LATINO	5% Dextrose	28
ASIAN	D5W	27
WHITE	Potassium Chloride	381

2. **Age and Procedure**

Table Creation:

```
session.execute("""
CREATE TABLE IF NOT EXISTS procs (
   PROCEDURE_NAME TEXT,
   AGE INT,
   PROCS_ID INT,
   ADM_ID INT,
   PAT_ID INT,
   D_ICD_ID INT,
   PRIMARY KEY ((AGE), PROCS_ID,ADM_ID,PAT_ID,D_ICD_ID)
);
""")
```

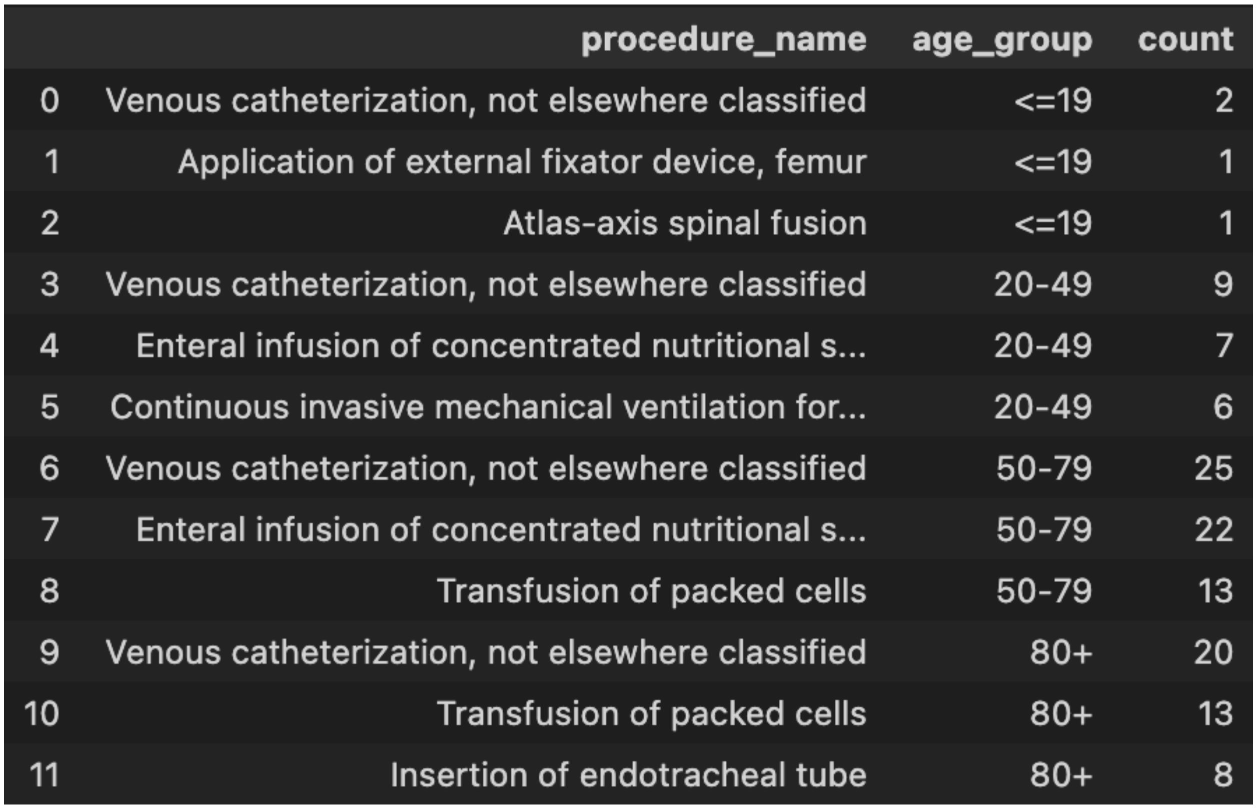
Data Upload:

```
insert_pat = session.prepare("""
   INSERT INTO procs (PROCEDURE_NAME, age, procs_id, adm_id, pat_id, d_icd_id)
   VALUES (?, ?, ?, ?,?,?)
""")

for _, row in df_procs.iterrows():
   session.execute(insert_pat, (
       str(row['PROCEDURE_NAME']),
       int(row['AGE']),
       int(row['PROCS_ID']),
       int(row['ADM_ID']),
       int(row['PAT_ID']),
       int(row['D_ICD_ID']),
   ))
```

Data Extraction:

```
rows = session.execute("SELECT procedure_name, age FROM procs")
df = pd.DataFrame(rows)

def age_group(age):
    if age <= 19:
        return "<=19"
    elif age <= 49:
        return "20-49"
    elif age <= 79:
        return "50-79"
    else:
        return "80+"

df['age_group'] = df['age'].apply(age_group)


df_grouped = (
    df.groupby(['procedure_name', 'age_group'])
    .size()
    .reset_index(name='count')
)



age_order = ['<=19', '20-49', '50-79', '80+']
df_top_procs['age_group'] = pd.Categorical(df_top_procs['age_group'],
categories=age_order, ordered=True)
df_top_procs = df_top_procs.sort_values(['age_group', 'count'], ascending=[True,
False]).reset_index(drop=True)

df_top_procs
```

Verify same output:



	procedure_name	age_group	count
0	Venous catheterization, not elsewhere classified	<=19	2
1	Application of external fixator device, femur	<=19	1
2	Atlas-axis spinal fusion	<=19	1
3	Venous catheterization, not elsewhere classified	20-49	9
4	Enteral infusion of concentrated nutritional s...	20-49	7
5	Continuous invasive mechanical ventilation for...	20-49	6
6	Venous catheterization, not elsewhere classified	50-79	25
7	Enteral infusion of concentrated nutritional s...	50-79	22
8	Transfusion of packed cells	50-79	13
9	Venous catheterization, not elsewhere classified	80+	20
10	Transfusion of packed cells	80+	13
11	Insertion of endotracheal tube	80+	8

3. ICU Stays

Create Tables (Note: created 2 because wanted 2 different partitions, one on gender, one on age)

```
session.execute("""
CREATE TABLE IF NOT EXISTS icu_shit_GENDER (
   ETHNICITY TEXT,
   ICU_HOURS INT,
   GENDER TEXT,
   ICU_ID INT,
   ADM_ID INT,
   PAT_ID INT,
   PRIMARY KEY ((GENDER),ICU_ID,ADM_ID,PAT_ID)
);""")

session.execute("""
CREATE TABLE IF NOT EXISTS icu_shit_ETHNICITY (
   ETHNICITY TEXT,
   ICU_HOURS INT,
   GENDER TEXT,
   ICU_ID INT,
   ADM_ID INT,
   PAT_ID INT,
   PRIMARY KEY ((ETHNICITY),ICU_ID,ADM_ID,PAT_ID)
);""")
```

Inserting Data:

```
insert_query_gender = session.prepare("""
   INSERT INTO icu_shit_GENDER (ETHNICITY, ICU_HOURS, GENDER, ICU_ID,ADM_ID,PAT_ID)
   VALUES (?, ?, ?, ?, ?, ?)
""")

for _, row in icu_fuck.iterrows():
   session.execute(insert_query_gender, (
       str(row["ethnicity"]),
       int(row["ICU_HOURS"]),
       str(row["gender"]),
       int(row["ICU_ID"]),
       int(row["adm_id"]),
       int(row["pat_id"])
   ))

insert_query_ethnicity = session.prepare("""
   INSERT INTO icu_shit_ETHNICITY (ETHNICITY, ICU_HOURS, GENDER, ICU_ID,ADM_ID,PAT_ID)
   VALUES (?, ?, ?, ?, ?, ?)
""")

for _, row in icu_fuck.iterrows():
   session.execute(insert_query_ethnicity, (
       str(row["ethnicity"]),
       int(row["ICU_HOURS"]),
       str(row["gender"]),
       int(row["ICU_ID"]),
       int(row["adm_id"]),
       int(row["pat_id"])
   ))
```

Extracting Data:

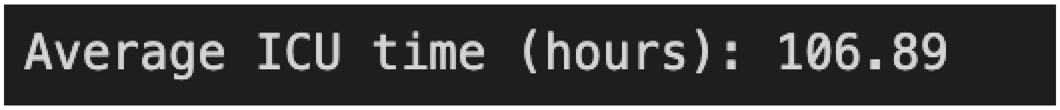
```
rows = session.execute("SELECT ICU_HOURS FROM icu_shit_ETHNICITY")
df = pd.DataFrame(rows)
avg_icu = df["icu_hours"].mean()
print(f"Average ICU time (hours): {round(avg_icu, 2)}")
```

```
genders = ["M","F"]
```

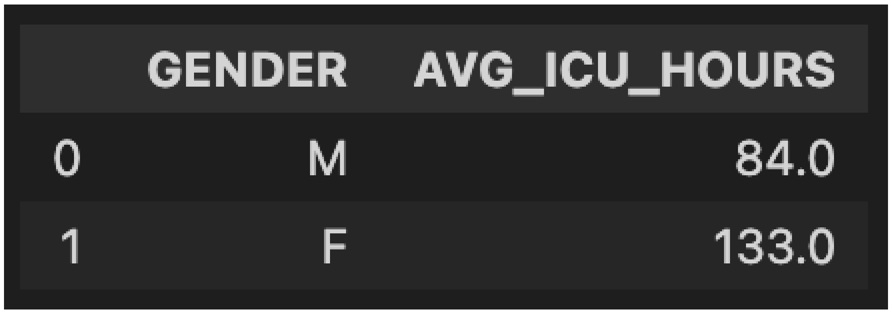
```
gender_stats = []

for gender in genders:
   rows = session.execute("SELECT ICU_HOURS FROM icu_shit_GENDER WHERE gender = %s",
[gender])
   df_gender = pd.DataFrame(rows)
   if not df_gender.empty:
       avg = df_gender["icu_hours"].mean()
       gender_stats.append({"GENDER": gender, "AVG_ICU_HOURS": round(avg, 0)})

df_gender_avg = pd.DataFrame(gender_stats)
df_gender_avg
```

```
ethnicities = icu_fuck["ethnicity"].unique()

ethnicity_stats = []

for eth in ethnicities:
   rows = session.execute("SELECT ICU_HOURS FROM icu_shit_ETHNICITY WHERE ethnicity =
%s", [eth])
   df_eth = pd.DataFrame(rows)
   if not df_eth.empty:
       avg = df_eth["icu_hours"].mean()
       ethnicity_stats.append({"ETHNICITY": eth, "AVG_ICU_HOURS": round(avg, 2)})

df_ethnicity_avg = df_ethnicity_avg.sort_values(by="AVG_ICU_HOURS",
ascending=False).reset_index(drop=True)
df_ethnicity_avg
```

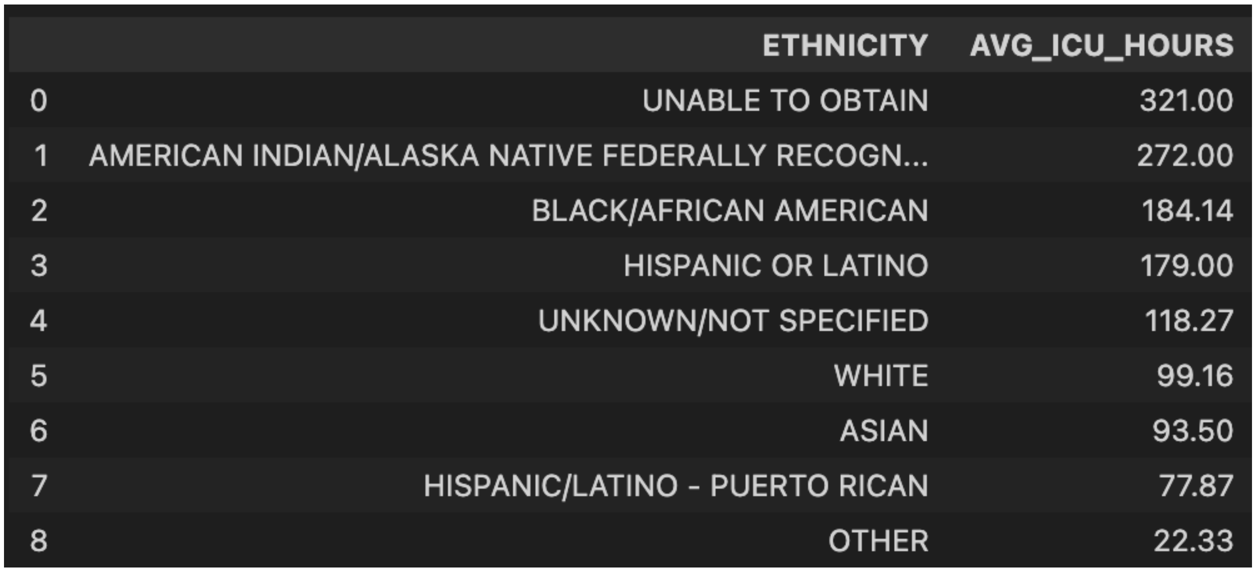
Verify same output:



```
Average ICU time (hours): 106.89
```



	GENDER	AVG_ICU_HOURS
0	M	84.0
1	F	133.0



	ETHNICITY	AVG_ICU_HOURS
0	UNABLE TO OBTAIN	321.00
1	AMERICAN INDIAN/ALASKA NATIVE FEDERALLY RECOGN...	272.00
2	BLACK/AFRICAN AMERICAN	184.14
3	HISPANIC OR LATINO	179.00
4	UNKNOWN/NOT SPECIFIED	118.27
5	WHITE	99.16
6	ASIAN	93.50
7	HISPANIC/LATINO - PUERTO RICAN	77.87
8	OTHER	22.33

Gen AI Disclosure

I used Chat GPT for a few things, namely on how to extract out the drug names for the first problem, debugging cassandra and why pandas' merge and SQL merge are different. I used several queries, namely:

1. I have this SQL query which effectively finds the amount of all drugs used by each ethnicity, how do I extract out the highest prescribed drug ethnicity from here?
2. Cassandra is not letting me upload the data, it is telling me that I have already created a table when I literally haven't. I am about to murder my laptop, please assist me, thank you :)
3. I am getting different fucking numbers between my SQL query and my cassandra query. Please tell me there is a goddamn difference between the way pandas merges data and how SQL merges data. If so, how do I correct it? You are the best Chat GPT, if you take over the world, know I appreciate your help!
4. Please help me debug (insert code here), it is not working.. again..