Ally Hayden

Homework 2

DATA_ENG 300

Part 1

Question 1:

A)

```
conn.sql("""
SELECT
    a.ethnicity,
    p.drug_type,
    COUNT(*) AS total_prescriptions
FROM prescriptions p
JOIN admissions a ON p.hadm_id = a.hadm_id
GROUP BY a.ethnicity, p.drug_type
ORDER BY a.ethnicity, total_prescriptions DESC
""").df()
```

B) Relevant functions

a. JOIN connects prescriptions and admissions using the hadm_id column
b. COUNT(*) sums the amount of time the drug type is prescribed
c. SUM(...) AS total_dose adds up all dosages

C)

	ethnicity	drug_type	total_prescriptions
0	AMERICAN INDIAN/ALASKA NATIVE FEDERALLY RECOGN...	MAIN	200
1	AMERICAN INDIAN/ALASKA NATIVE FEDERALLY RECOGN...	BASE	80
2	AMERICAN INDIAN/ALASKA NATIVE FEDERALLY RECOGN...	ADDITIVE	2
3	ASIAN	MAIN	121
4	ASIAN	BASE	56

D) The three type of drugs are Main, Base, and Additive. The Main drug type was the most common for all of the ethnicities, followed by Base, and then Additive. White patients had by far the highest number of total prescriptions with 5,420 Main, 1,743 Base, and 66 Additive. In fact, this was the only ethnicity to have a notable number of additive presciptions. Hispanic/Latino – Puerto Rican patients had the second highest number of prescriptions and Asian had the lowest of the named ethnicities.

Question 2:

A)

```
conn.sql("""
WITH base AS (
    SELECT
        p.subject_id,
        a.hadm_id,
        FLOOR(DATE_DIFF('day', p.dob, a.admittime) / 365.25) AS age,
        d.long_title AS procedure_name
    FROM admissions a
    JOIN patients p ON a.subject_id = p.subject_id
    JOIN procedures_icd pi ON a.hadm_id = pi.hadm_id
    JOIN d_icd_procedures d ON pi.icd9_code = d.icd9_code
),
age_grouped AS (
    SELECT *,
        CASE
            WHEN age <= 19 THEN '0-19'
            WHEN age BETWEEN 20 AND 49 THEN '20-49'
            WHEN age BETWEEN 50 AND 79 THEN '50-79'
            ELSE '>80'
        END AS age_group
    FROM base
),
ranked AS (
    SELECT
        age_group,
        procedure_name,
        COUNT(*) AS count,
        ROW_NUMBER() OVER (PARTITION BY age_group ORDER BY COUNT(*) DESC) AS rank
    FROM age_grouped
    GROUP BY age_group, procedure_name
)
SELECT age_group, procedure_name, count
FROM ranked
WHERE rank <= 3
ORDER BY age_group, rank;
""").df()
```

B) Relevant Functions

a. CTE 1
   i. performs all of the joins (patients, admissions, procedures)
   ii. calculates age

b. CTE 2
   i. CASE WHEN assigns each row an age_group

c. CTE 3
   i. groups by age_group and procedure_name
   ii. counts procedures
   iii. ROW_NUMBER() ranks top 3 procedures

d. Final SELECT
   i. filters for rank <=3

	age_group	procedure_name	count
0	0-19	Venous catheterization, not elsewhere classified	3
1	0-19	Closure of skin and subcutaneous tissue of oth...	2
2	0-19	Other diagnostic procedures on brain and cereb...	1
3	20-49	Venous catheterization, not elsewhere classified	8
4	20-49	Enteral infusion of concentrated nutritional s...	7

C)

D) The analysis from question 2 identified the most common procedures among different age groups. I found that venous catheterization was the most common across all groups with it appearing the most number of times in the 50-79 age group with a count of 26. Enteral infusion of concentrated nutritional substances was the second most common in the 20-49 and 50-79 age groups. The most diversity of procedures was found in the 0-19 age group but this group also saw the lowest count of procedures in the data.

Question 3:

A)

```
conn.sql("""
SELECT
    p.gender,
    a.ethnicity,
    AVG(DATE_DIFF('day', i.intime, i.outtime)) AS avg_icu_stay_days,
    COUNT(*) AS num_stays
FROM icustays i
JOIN patients p ON i.subject_id = p.subject_id
JOIN admissions a ON i.hadm_id = a.hadm_id
WHERE i.intime IS NOT NULL AND i.outtime IS NOT NULL
GROUP BY p.gender, a.ethnicity
ORDER BY a.ethnicity,p.gender;
""").df()
```

B) Relevant functions

a. JOIN connects icustays, patients, and admissions
b. DATE_DIFF calculates the ICU length of stay in days for each patient by subtracting admission and discharge times
c. AVG(...) AS avg_icu_stay_days finds the average ICU stay length by group (gender and ethnicity)
d. COUNT(*) AS num_stays Counts the total number of ICU stays for each group

C)

	gender	ethnicity	avg_icu_stay_days	num_stays
0	M	AMERICAN INDIAN/ALASKA NATIVE FEDERALLY RECOGN...	11.500000	2
1	F	ASIAN	1.000000	1
2	M	ASIAN	7.000000	1
3	F	BLACK/AFRICAN AMERICAN	11.250000	4
4	M	BLACK/AFRICAN AMERICAN	3.000000	3

D) This question looked at average stay length across both gender and ethnicity. I found that white patients had the highest number of stays (48M and 44F) but their average durations were relatively short. Black/African American females had a notably long average duration of 11.25 days which is more than twice that of Black/African American males. In general, females tended to have slightly longer stays across multiple groups but that could have been influenced but smaller sample sizes in certain ethnicities.

Part 2

Question 1:

A)

```
session.execute("""
CREATE TABLE IF NOT EXISTS drug_usage_by_ethnicity (
    ethnicity TEXT,
    drug_type TEXT,
    id UUID,
    dose_val_rx TEXT,
    PRIMARY KEY ((ethnicity), drug_type, id)
)
""")
```

B)

```
#join data using pandas
import pandas as pd

# relevant fields
admissions = conn.sql("SELECT hadm_id, ethnicity FROM admissions").df()
prescriptions = conn.sql("SELECT hadm_id, drug_type, dose_val_rx FROM prescriptions").df()

merged = prescriptions.merge(admissions, on="hadm_id", how="left")
```

```
#insert sample rows into cassandra
import uuid

sample = merged.head(5000)

for _, row in sample.iterrows():
    session.execute("""
        INSERT INTO drug_usage_by_ethnicity (ethnicity, drug_type, id, dose_val_rx)
        VALUES (%s, %s, %s, %s)
    """, (
        str(row.ethnicity),
        str(row.drug_type),
        uuid.uuid4(),
        str(row.dose_val_rx)
    ))
```

```
rows = session.execute("""
SELECT ethnicity, drug_type FROM drug_usage_by_ethnicity
""")

df = pd.DataFrame(rows.all())
```

C)

```
summary = (
    df.groupby(["ethnicity", "drug_type"])
    .size()
    .reset_index(name="total_prescriptions")
    .sort_values(["ethnicity", "total_prescriptions"], ascending=[True, False])
)

top_usage = summary.groupby("ethnicity").head(1)
```

```
summary.head()
```

	ethnicity	drug_type	total_prescriptions
1	ASIAN	MAIN	657
0	ASIAN	BASE	285
3	BLACK/AFRICAN AMERICAN	MAIN	1542
2	BLACK/AFRICAN AMERICAN	BASE	410
5	HISPANIC OR LATINO	MAIN	163

D)

```
top_usage
```

	ethnicity	drug_type	total_prescriptions
1	ASIAN	MAIN	657
3	BLACK/AFRICAN AMERICAN	MAIN	1542
5	HISPANIC OR LATINO	MAIN	163
7	HISPANIC/LATINO - PUERTO RICAN	MAIN	860
9	OTHER	MAIN	98
12	UNABLE TO OBTAIN	MAIN	89
14	UNKNOWN/NOT SPECIFIED	MAIN	930
17	WHITE	MAIN	19300

Question 2:

A)

```
session.execute("""
CREATE TABLE IF NOT EXISTS procedures_by_age (
    age_group TEXT,
    procedure_name TEXT,
    id UUID,
    PRIMARY KEY ((age_group), procedure_name, id)
)
""")
```

B)

```
# relevant fields
admissions = conn.sql("SELECT subject_id, hadm_id, admittime FROM admissions").df()
patients = conn.sql("SELECT subject_id, dob FROM patients").df()
procedures = conn.sql("SELECT hadm_id, icd9_code FROM procedures_icd").df()
proc_titles = conn.sql("SELECT icd9_code, long_title FROM d_icd_procedures").df()

base = admissions.merge(patients, on="subject_id", how="left") \
                 .merge(procedures, on="hadm_id", how="inner") \
                 .merge(proc_titles, on="icd9_code", how="left")

# compute age
base["age"] = ((pd.to_datetime(base["admittime"]) - pd.to_datetime(base["dob"])) / pd.Timedelta(days=365.25)).astype(int)

# function to assign age
def assign_age_group(age):
    if age <= 19:
        return "0-19"
    elif age <= 49:
        return "20-49"
    elif age <= 79:
        return "50-79"
    else:
        return ">80"

base["age_group"] = base["age"].apply(assign_age_group)

sample = base[["age_group", "long_title"]].dropna().head(1000)

for _, row in sample.iterrows():
    session.execute("""
        INSERT INTO procedures_by_age (age_group, procedure_name, id)
        VALUES (%s, %s, %s)
    """, (
        row.age_group,
        row.long_title,
        uuid.uuid4()
    ))
```

C)

```
rows = session.execute("SELECT age_group, procedure_name FROM procedures_by_age")
df = pd.DataFrame(rows.all())

summary = (
    df.groupby(["age_group", "procedure_name"])
    .size()
    .reset_index(name="count")
    .sort_values(["age_group", "count"], ascending=[True, False])
)

top_3 = summary.groupby("age_group").head(3)
```

D)

```
top_3
```

	age_group	procedure_name	count
27	0-19	Venous catheterization, not elsewhere classified	28
5	0-19	Closure of skin and subcutaneous tissue of oth...	8
0	0-19	Application of external fixator device, femur	4
71	20-49	Venous catheterization, not elsewhere classified	32
40	20-49	Enteral infusion of concentrated nutritional s...	28
37	20-49	Continuous invasive mechanical ventilation for...	24

Question 3:

A)

```
session.execute("""
CREATE TABLE IF NOT EXISTS icu_stays_by_demo (
    gender TEXT,
    ethnicity TEXT,
    icu_stay_length DOUBLE,
    id UUID,
    PRIMARY KEY ((ethnicity), gender, id)
)
""")
```

B)

```
# relevant fields
icustays = conn.sql("SELECT subject_id, hadm_id, intime, outtime FROM icustays").df()
patients = conn.sql("SELECT subject_id, gender FROM patients").df()
admissions = conn.sql("SELECT hadm_id, ethnicity FROM admissions").df()

df = icustays.merge(patients, on="subject_id", how="left") \
             .merge(admissions, on="hadm_id", how="left")

df = df.dropna(subset=["intime", "outtime"])

# calculate ICU stay
df["icu_stay_length"] = (
    pd.to_datetime(df["outtime"]) - pd.to_datetime(df["intime"])
).dt.total_seconds() / (60 * 60 * 24)  # convert to days

sample = df[["gender", "ethnicity", "icu_stay_length"]].dropna().head(1000)

for _, row in sample.iterrows():
    session.execute("""
        INSERT INTO icu_stays_by_demo (ethnicity, gender, icu_stay_length, id)
        VALUES (%s, %s, %s, %s)
    """, (
        str(row.ethnicity),
        str(row.gender),
        float(row.icu_stay_length),
        uuid.uuid4()
    ))
```

C)

```
rows = session.execute("SELECT ethnicity, gender, icu_stay_length FROM icu_stays_by_demo")
icu_df = pd.DataFrame(rows.all())

summary = (
    icu_df.groupby(["gender", "ethnicity"])
    .agg(avg_icu_stay_days=("icu_stay_length", "mean"), num_stays=("icu_stay_length", "count"))
    .reset_index()
    .sort_values(["ethnicity", "gender"])
)
```

D)

```
summary
```

	gender	ethnicity	avg_icu_stay_days	num_stays
6	M	AMERICAN INDIAN/ALASKA NATIVE FEDERALLY RECOGN...	11.337159	6
0	F	ASIAN	0.662801	3
7	M	ASIAN	7.117303	3
1	F	BLACK/AFRICAN AMERICAN	11.201241	12
8	M	BLACK/AFRICAN AMERICAN	2.977245	9
2	F	HISPANIC OR LATINO	7.459637	9