MASTERS RESEARCH PROJECT – 02

TITLE - Breast Cancer Risk Analysis

GROUP - 14

Preliminary Data Generation and Reporting

1. **Synthea Script for Population Generation.**

The following script was used to generate a synthetic patient population for the state of California using Synthea:

- -s 12025: Specifies the seed (a five-digit number) for reproducibility.
- -p 5000: Generated 5000 patients.
- California: Specifies the population is limited to the state of California.
- Command Given **- ./run_synthea -s 12025 -p 5000 California.**

2. **Size of the Population**

The dataset contains **5000 patients**, as specified during the generation process.

3. **Descriptive Statistics of the Patient Population.**

Here are the descriptive statistics for the patient population:

- The age range of the patient population is from 0 to 110 years old.
- The gender distribution consists of 2,885 females and 2,879 males.
- Regarding race, there are 4,221 White patients, 852 Asian patients, 363 Black patients, 114 Native patients, 109 patients categorized as Other, and 105 Hawaiian patients.
- The city with the highest patient population is **Los Angeles**, while the city with the lowest patient population is **Rancho Mirage**.

4. **Data Integrity Checks**

We have run the referential integrity check and it confirms that there are no data integrity issues in terms of foreign key mismatches across the linked tables. The results indicate that all PATIENT IDs in the **conditions**, **procedures**, **observations**, and **imaging_studies** tables have valid references in the **patients** table, maintaining a consistent relationship between the primary key (Id in patients.csv) and the foreign keys (PATIENT in related tables). Since no missing patient references were detected, the database structure maintains proper integrity, ensuring accurate linkages between patient records and their associated medical data.

5. **Proposed Reports for Addressing the Problem Statement -**

1. High-Risk Patient Identification Report

This report identifies individual patients who are at high risk of developing breast cancer based on their medical history and procedures. The analysis includes:

- Identifying patients diagnosed with breast cancer.

- Checking if they have undergone breast-related procedures such as mastectomy or chemotherapy.
- Categorizing patients into High, Moderate, or Low Risk based on these factors.

Why This Report Is Important

- Helps primary Health care providers in identify patients who need urgent screening and monitoring.
- Supports early detection, which improves treatment outcomes.
- Provides a structured approach for prioritizing high-risk individuals.

Report Output

- A list of the **Top 10 High-Risk Patients** (sorted by age).
- A **Risk Distribution Report** showing how many patients fall into High, Moderate, and Low Risk categories.

2. Breast Cancer Risk Distribution Report

This report analyzes how breast cancer risk is distributed across different groups in the population. It does not focus on individual patients but instead looks at trends in risk levels across:

- Age groups to determine which age ranges have the highest number of high-risk patients.
- Geographic locations to identify cities with the most high-risk patients.

Why This Report Is Important

- Helps public health officials allocate resources to areas with higher risk.
- Identifies which age groups are most affected, aiding in targeted prevention strategies.
- Supports data-driven decision-making for screening programs and awareness campaigns.

Report Output

- A Risk Levels Report showing counts of High, Moderate, and Low Risk patients.
- An Age-Based Risk Report to see how risk is distributed among different age groups.
- A City-Based Risk Report highlighting the top 10 cities with the most high-risk patients.

The High-Risk Patient Identification Report is essential for **individual patient care**, allowing doctors to focus on those at greatest risk. The Breast Cancer Risk Distribution Report is valuable for **public health planning**, providing insights into trends that can guide resource allocation and early detection programs. Together, these reports contribute to better healthcare planning and improved breast cancer management.

6. **The script used to generate the report using Python.**

#Data Integrity check used codes

```
# Import necessary libraries
import pandas as pd
import matplotlib.pyplot as plt
```

```
from datetime import datetime

# Load datasets
patients = pd.read_csv("patients.csv")
conditions = pd.read_csv("conditions.csv")
procedures = pd.read_csv("procedures.csv")
observations = pd.read_csv("observations.csv")
imaging_studies = pd.read_csv("imaging_studies.csv")

# Convert birthdate to datetime format and calculate age
patients["BIRTHDATE"] = pd.to_datetime(patients["BIRTHDATE"], errors='coerce')
current_year = datetime.now().year
patients["AGE"] = current_year - patients["BIRTHDATE"].dt.year

# Compute age range
age_range = (patients["AGE"].min(), patients["AGE"].max())

# Get gender counts
gender_counts = patients["GENDER"].value_counts()

# Get race distribution
race_counts = patients["RACE"].value_counts()

# Get patient count per city
city_counts = patients["CITY"].value_counts()
highest_population_city = city_counts.idxmax()
lowest_population_city = city_counts.idxmin()

# Store statistics in a dictionary
statistics = {
    "Age Range": age_range,
    "Gender Counts": gender_counts.to_dict(),
    "Race Counts": race_counts.to_dict(),
    "City with Highest Population": highest_population_city,
```

```
    "City with Lowest Population": lowest_population_city,
}

# Display patient distribution per city
import ace_tools as tools
tools.display_dataframe_to_user(name="City Patient Distribution",
dataframe=city_counts.to_frame())

# Now we check for data integrity to ensure that all foreign keys are valid
patient_ids = set(patients["Id"])

# Checking if all linked tables reference valid patients
for table_name, df in [
    ("conditions.csv", conditions),
    ("procedures.csv", procedures),
    ("observations.csv", observations),
    ("imaging_studies.csv", imaging_studies)
]:
    missing_patients = df[~df["PATIENT"].isin(patient_ids)]["PATIENT"].unique()

    if len(missing_patients) > 0:
        print(f" ⚠️ Referential Integrity Issue in {table_name}: {len(missing_patients)} missing
PATIENT IDs")
    else:
        print(f"Referential Integrity OK for {table_name}")

# Checking for missing values in each dataset
for table_name, df in [
    ("patients.csv", patients),
    ("conditions.csv", conditions),
    ("procedures.csv", procedures),
    ("observations.csv", observations),
    ("imaging_studies.csv", imaging_studies)
]:
```

```
    missing_counts = df.isnull().sum()
    missing_counts = missing_counts[missing_counts > 0]  # Only show columns with missing values

    if not missing_counts.empty:
        print(f" Missing Values in {table_name}:")
        print(missing_counts, "\n")
    else:
        print(f" No missing values in {table_name}")

# Now we validate if the data types match the expected formats
expected_types = {
    "patients.csv": {"Id": "object", "BIRTHDATE": "object", "GENDER": "object", "INCOME":
"int64"},
    "conditions.csv": {"PATIENT": "object", "DESCRIPTION": "object"},
    "procedures.csv": {"PATIENT": "object", "DESCRIPTION": "object"},
    "observations.csv": {"PATIENT": "object", "DESCRIPTION": "object", "VALUE": "object"},
    "imaging_studies.csv": {"PATIENT": "object", "DATE": "object",
"BODYSITE_DESCRIPTION": "object"},
}

for table_name, df in [
    ("patients.csv", patients),
    ("conditions.csv", conditions),
    ("procedures.csv", procedures),
    ("observations.csv", observations),
    ("imaging_studies.csv", imaging_studies)
]:
    for col, expected_type in expected_types[table_name].items():
        actual_type = df[col].dtype
        if actual_type != expected_type:
            print(f" Data Type Mismatch in {table_name}: {col} (Expected: {expected_type}, Found:
{actual_type})")

# Now we check if there are any invalid values in categorical columns like gender, race, and ethnicity
```

```
valid_genders = {"F", "M", "Other"}
valid_races = {"white", "black", "asian", "other", "hawaiian", "native"}
valid_ethnicities = {"hispanic", "nonhispanic", "other"}

invalid_genders = patients[~patients["GENDER"].isin(valid_genders)]["GENDER"].unique()
invalid_races = patients[~patients["RACE"].isin(valid_races)]["RACE"].unique()
invalid_ethnicities =
patients[~patients["ETHNICITY"].isin(valid_ethnicities)]["ETHNICITY"].unique()

if len(invalid_genders) > 0:
   print(f" Invalid GENDER values found: {invalid_genders}")
if len(invalid_races) > 0:
   print(f" Invalid RACE values found: {invalid_races}")
if len(invalid_ethnicities) > 0:
   print(f" Invalid ETHNICITY values found: {invalid_ethnicities}")

# Finally, let's plot the age distribution to visualize patient demographics
plt.figure(figsize=(8, 5))
plt.hist(patients["AGE"].dropna(), bins=20, edgecolor="black", alpha=0.7)
plt.xlabel("Age")
plt.ylabel("Frequency")
plt.title("Patient Age Distribution")
plt.grid(axis="y", linestyle="--", alpha=0.6)
plt.show()
```

#Code Used to Generate Top 10 High-Risk Patients

```
import pandas as pd
from datetime import datetime

# Load datasets
conditions_df = pd.read_csv("conditions.csv")
patients_df = pd.read_csv("patients.csv")
procedures_df = pd.read_csv("procedures.csv")

# Identify patients diagnosed with breast cancer
```

```
breast_cancer_patients = conditions_df[
    conditions_df["DESCRIPTION"].str.contains("Malignant neoplasm of breast", case=False,
na=False)
]
breast_cancer_patient_ids = breast_cancer_patients["PATIENT"].unique()

# Identify patients who had breast-related procedures
breast_procedures = procedures_df[
    procedures_df["DESCRIPTION"].str.contains("breast", case=False, na=False)
]
procedure_patient_ids = breast_procedures["PATIENT"].unique()

# Merge with patient demographics
patients_df["BIRTHDATE"] = pd.to_datetime(patients_df["BIRTHDATE"], errors='coerce')
current_year = datetime.now().year
patients_df["AGE"] = current_year - patients_df["BIRTHDATE"].dt.year

# Filter breast cancer patients
breast_cancer_patients_df = patients_df[patients_df["Id"].isin(breast_cancer_patient_ids)][["Id",
"AGE", "GENDER", "CITY", "STATE"]]

# Function to classify risk
def classify_risk(patient_id):
    has_breast_cancer = patient_id in breast_cancer_patient_ids
    has_breast_procedure = patient_id in procedure_patient_ids

    if has_breast_cancer and has_breast_procedure:
        return "High"
    elif has_breast_cancer or has_breast_procedure:
        return "Moderate"
    else:
        return "Low"

# Assign risk level
```

```
breast_cancer_patients_df["Risk_Score"] = breast_cancer_patients_df["Id"].apply(classify_risk)

# Get only high-risk patients
high_risk_patients = breast_cancer_patients_df[breast_cancer_patients_df["Risk_Score"] == "High"]

# Sort by age (oldest first) and select top 10
top_10_high_risk = high_risk_patients.sort_values(by="AGE", ascending=False).head(10)

# Display results
print(top_10_high_risk.to_string(index=False))

# Code used for Breast Cancer Risk Distribution Report
# Count patients by risk level
#Risk Levels Distribution
risk_distribution = breast_cancer_patients_df["Risk_Score"].value_counts().reset_index()
risk_distribution.columns = ["Risk Level", "Patient Count"]

# Display the risk distribution
print(risk_distribution.to_string(index=False))
#High-Risk Patients by Age Group
# Define age bins and labels
age_bins = [20,30,40,50,60,70,80,90,100,120]
age_labels = ["20-29", "30-39", "40-49", "50-59", "60-69", "70-79", "80-89", "90-99", "100+"]
breast_cancer_patients_df["Age_Group"] = pd.cut(breast_cancer_patients_df["AGE"],
bins=age_bins, labels=age_labels, right=False)

# Count number of patients per age group
risk_by_age = breast_cancer_patients_df.groupby(["Age_Group",
"Risk_Score"]).size().unstack(fill_value=0)

# Display results
print(risk_by_age)
#Top 10 Cities with the Most High-Risk Patients
# Count number of high-risk patients per city
```

```
high_risk_by_city = breast_cancer_patients_df[breast_cancer_patients_df["Risk_Score"] ==
"High"].groupby("CITY").size().reset_index(name="High-Risk Count")

# Sort by highest risk count and select top 10
high_risk_by_city = high_risk_by_city.sort_values(by="High-Risk Count",
ascending=False).head(10)

# Display top 10 cities
print(high_risk_by_city.to_string(index=False)).
```

7. **Top 10 results from the report in the form of an appropriate table in the document.**

Table 1: Top 10 High-Risk Breast Cancer Patients

This table contains the top 10 high-risk patients based on age, gender, and location.

Id	AGE	GENDER	CITY	STATE	Risk_Score
a463fe93-cf66-e0c9-ccf6-c04e61626037	110	F	Modesto	California	High
52cfd59c-7777-c12b-2aa9-d25ff5e93789	105	F	San Jose	California	High
119ac92c-00b8-be08-7d78-f7500d01c406	90	F	Glendora	California	High
d7078623-34bb-44bb-a27e-a364abc6c240	81	F	Los Angeles	California	High
66cdeb6f-656a-ab64-0121-f1fc9758d53d	78	F	San Francisco	California	High
85dddd40-93eb-c1bf-2cd8-216c85e19343	78	F	Valle Vista	California	High
d4786bdf-3ff9-b24b-3a06-b9fd920fcfad	78	F	Valle Vista	California	High
5d441f2b-483c-d018-910a-e00d2729e733	78	F	Santa Ana	California	High

3ea2fac8-141f-8cf5-ab12-7997785864b4	78	F	Bell	California	High
80000032-c9a3-36a8-35cb-d2ac3f4bf39b	77	M	Desert Palms	California	High

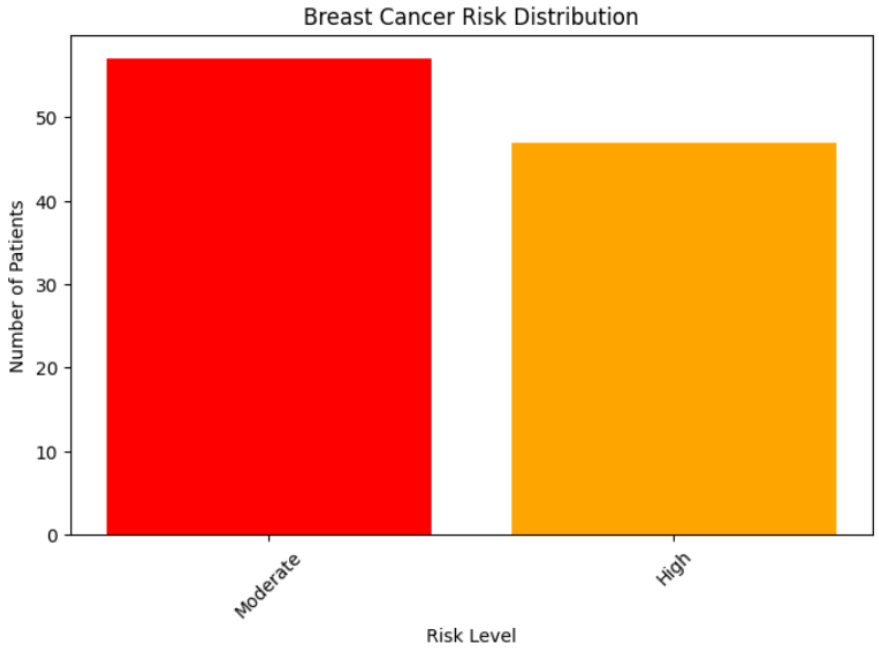
Table 2: Breast Cancer Risk Distribution by Age and Location

This table provides an overview of **risk levels (High, Moderate, Low) distributed by age groups and top high-risk cities**.

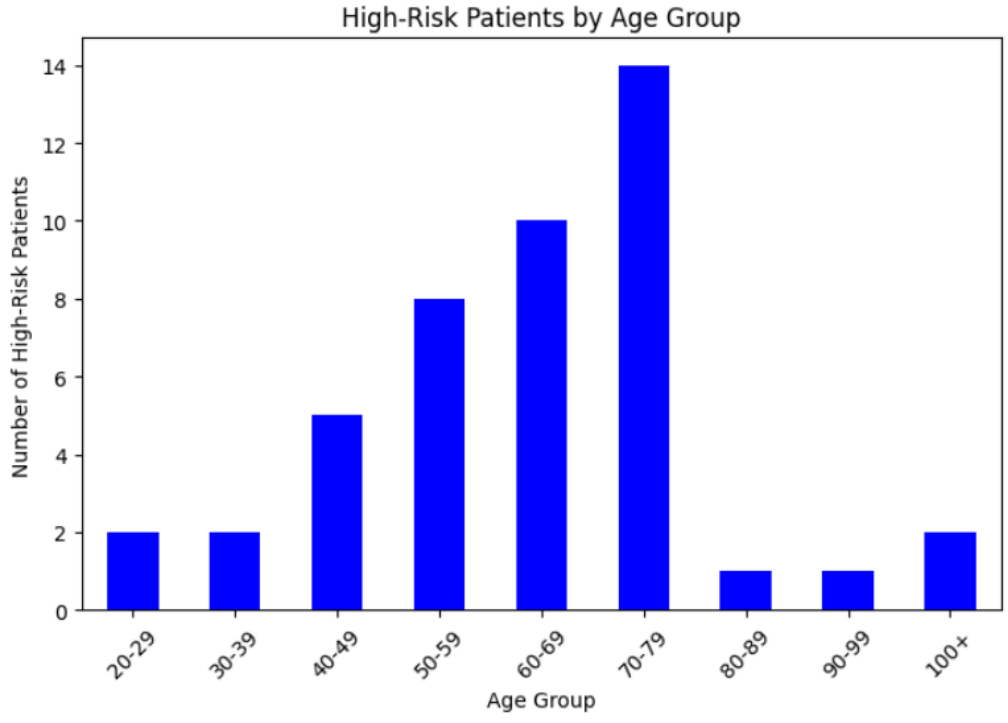
Age Group	**High-Risk Count**	**CITY**
70-79	14	Modesto
60-69	10	Bell
50-59	8	Los Angeles
40-49	5	Desert Palms
55-65	4	San Diego
57-58	3	San Jose
30-39	2	Santa Ana
20-29	2	Fresno
20-29	2	San Francisco
20-29	2	Valle Vista

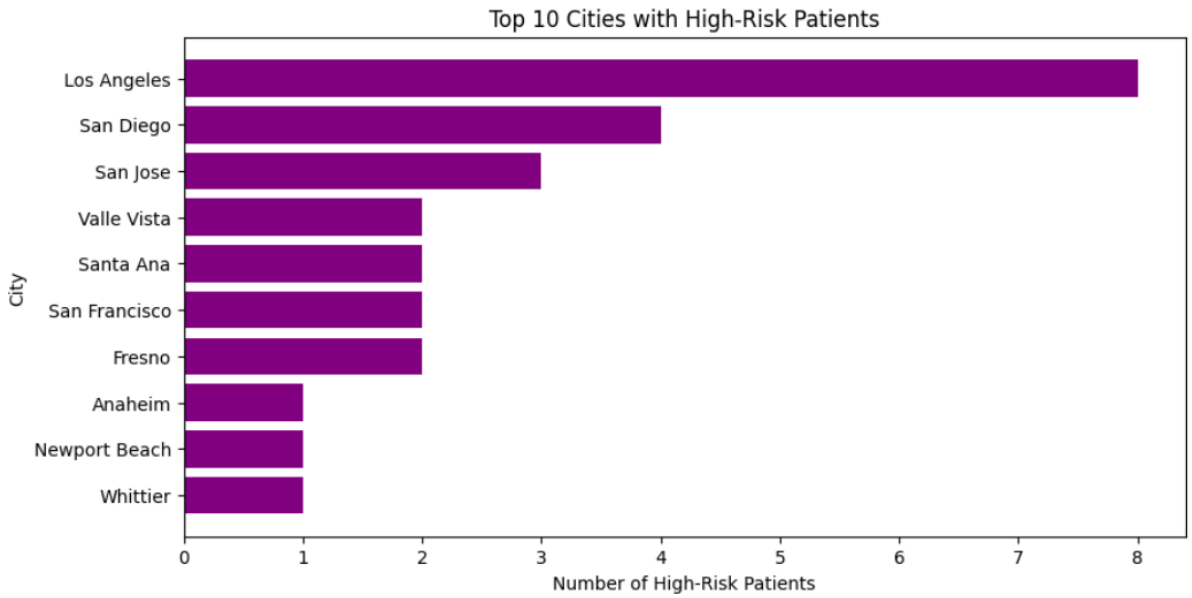
Additionally, we are attaching the plots which we generated for both our reports.

```
plt.show()
```




```
<ipython-input-28-791a06836a18>:7: FutureWarning: The default of observed=False is deprecated
  high_risk_age_distribution = breast_cancer_patients_df[breast_cancer_patients_df["Risk_Scor
```





8. Use of any AI tools.

Yes, we used AI tools, specifically ChatGPT, to assist with this assignment. ChatGPT helped in **generating Python code** for the two reports: **High-Risk Patient Identification** and **Breast Cancer Risk Distribution**. It provided structured code to extract data, classify risk levels, and display results.

We also used ChatGPT to **verify our outcomes**, ensuring the results were correct and troubleshooting when needed. At the end, ChatGPT helped in **aligning our assignment properly**, making sure all sections were correctly formatted and nothing was missing.

Prompts Used

- "Generate Python code for high-risk breast cancer patients."
- "Check if my code correctly identifies high-risk patients."
- "Generate risk distribution by age and city."
- "Help me align my assignment in the correct format."