Health Data Dashboard

In [ ]:
```
# Import Library
import streamlit as st
import plotly.express as px
import pandas as pd
import os
import warnings
warnings.filterwarnings('ignore')
```

In [ ]:
```
# Setting the title
st.set_page_config(page_title="Healthcare Data",  page_icon=":bar_chart", layout="wide")
st.title(":bar_chart: Healthcare Data Dashboard")
st.markdown('<style>div.block-container{padding-top:2rem;}</style>', unsafe_allow_html=True)
```

Load Data

In [ ]:
```
# Load the csv file
file_path = "healthcare_dataset.csv"
df = pd.read_csv(file_path)
```

Data Cleaning

In [ ]:
```
# Convert 'Date of Admission' to datetime
df['Date of Admission'] = pd.to_datetime(df['Date of Admission'], errors='coerce')

# Check for NaT values in 'Date of Admission' after conversion
nat_count = df['Date of Admission'].isna().sum()
if nat_count > 0:
    st.warning(f"There are {nat_count} invalid date entries in 'Date of Admission'.")

# Clean the DataFrame (remove rows where Name is NaN)
df_clean = df[df['Name'].notna()]
```

Filter by Year

In [ ]:
```
# Getting the unique years for the dropdown
df_clean['Year'] = df_clean['Date of Admission'].dt.year  # Extract the year
years = df_clean['Year'].unique()  # Get unique years
years.sort()  # Sort the years

# Include an "All Years" option
all_years_option = "All Years"
year_options = [all_years_option] + years.tolist()  # Add "All Years" to the list

# Create a dropdown to select a year
selected_year = st.selectbox("Select Year", year_options)

# Filter the DataFrame based on the selected year
if selected_year == all_years_option:
    df_clean_filtered = df_clean.copy()  # No filtering, show all data
else:
    df_clean_filtered = df_clean[df_clean['Year'] == int(selected_year)].copy()

# Display the filtered DataFrame
st.write(f"Filtered Data for Year: {selected_year}", df_clean_filtered)
```

Second Row

In [ ]:
```
# Add total number of Patient, Doctor, Hospital, Billing amount and Insurance Provider
st.header('Patient Overview')
total_patients = df_clean_filtered['Name'].nunique()
total_bill = df_clean_filtered['Billing Amount'].sum()
total_hospital = df_clean_filtered['Hospital'].nunique()
total_doctor = df_clean_filtered['Doctor'].nunique()
total_insurance = df_clean_filtered["Insurance Provider"].nunique()

col1, col2, col3, col4, col5 = st.columns(5)
col1.metric("Total Patients", total_patients)  # Corrected
col2.metric("Total Billing Amount", f"${total_bill:,.2f}")  # Formatted billing amount
col3.metric("Total Hospitals", total_hospital)
col4.metric("Total Doctors", total_doctor)
col5.metric("Total Insurance Providers", total_insurance)

col1, col2, col3 = st.columns(3)
# ---- Plot 1: Patient Distribution by Gender (Pie Chart) ----
with col1:
    st.subheader('Patient Distribution by Gender')
    gender_count = df_clean_filtered['Gender'].value_counts()
    fig_gender_pie = px.pie(values=gender_count.values,
                            names=gender_count.index,
                            hole=0.4,
                            color_discrete_sequence=px.colors.qualitative.Set2)
    st.plotly_chart(fig_gender_pie)

# ---- Plot 2: Total Patients by Age Group (Bar Chart) ----
with col2:
    st.subheader('Total Patients by Age Group')
    # Define age bins and group patients by age
    bins = [0, 18, 35, 50, 65, 80, 100]
    labels = ['0-18', '19-35', '36-50', '51-65', '66-80', '81-100']
    df_clean_filtered['Age Group'] = pd.cut(df_clean_filtered['Age'], bins=bins, labels=labels, right=False)

    # Count the number of patients in each age group
    age_group_count = df_clean_filtered['Age Group'].value_counts().sort_index()

    fig_age = px.bar(age_group_count,
                     x=age_group_count.index,
                     y=age_group_count.values,
                     labels={'x': 'Age Group', 'y': 'Count'},
                     title="Patients by Age Group",
                    color_discrete_sequence=px.colors.qualitative.Set1_r)
    st.plotly_chart(fig_age)

    # ---- Plot 3: Total Billing Amount by Year (Bar Chart) ----
with col3:
    st.subheader('Total Billing Amount by Year')

    # Extract the year from 'Date of Admission'
    df_clean_filtered['Year of Admission'] = pd.to_datetime(df_clean_filtered['Date of Admission']).dt.year

    # Group by year and sum the billing amounts
    billing_by_year = df_clean_filtered.groupby('Year of Admission')['Billing Amount'].sum().reset_index()

    fig_billing = px.line(billing_by_year,
                          x='Year of Admission',
                          y='Billing Amount',
                          labels={'Year of Admission': 'Year', 'Billing Amount': 'Total Billing Amount'},
                          title="Total Billing Amount by Year",
                          color_discrete_sequence=px.colors.qualitative.Set1)
    st.plotly_chart(fig_billing)
```

Third Row

In [ ]:
```
# Create three columns for the second set of plots
col1, col2, col3 = st.columns(3)

# ---- Plot 1: Total Patients by Medical Condition (Bar Chart) ----
with col1:
    # st.subheader('Total Patients by Medical Condition')
    condition_count = df_clean_filtered['Medical Condition'].value_counts()

    fig_condition = px.bar(condition_count,
                           x=condition_count.index,
                           y=condition_count.values,
                           labels={'x': 'Medical Condition', 'y': 'Count'},
                           title="Patients by Medical Condition",
                           color_discrete_sequence=px.colors.qualitative.Set2)
    st.plotly_chart(fig_condition)
    # ---- Plot 2: Total Patients by Admission Type (Bar Chart) ----
with col2:
    # st.subheader('Total Patients by Admission Type')
    admission_count = df_clean_filtered['Admission Type'].value_counts()

    fig_admission = px.bar(admission_count,
                           x=admission_count.index,
                           y=admission_count.values,
                           labels={'x': 'Admission Type', 'y': 'Count'},
                           title="Patients by Admission Type",
                           color_discrete_sequence=px.colors.qualitative.Set3)
    st.plotly_chart(fig_admission)

# ---- Plot 3: Total Billing Amount by Insurance Provider (Donut Chart) ----
with col3:
    # st.subheader('Total Billing Amount by Insurance Provider')

    # Group by Insurance Provider and sum the billing amounts
    billing_by_insurance = df_clean_filtered.groupby('Insurance Provider')['Billing Amount'].sum().reset_index()

    fig_donut = px.pie(billing_by_insurance,
                       names='Insurance Provider',
                       values='Billing Amount',
                       hole=0.4,  # This makes it a donut chart
                       title="Total Billing Amount by Insurance Provider",
                       color_discrete_sequence=px.colors.qualitative.Pastel)

    st.plotly_chart(fig_donut)
```