1

Examen Parcial 1

Heli Saul, Vásquez Gómez, 201700852
Escuela de Mecánica Eléctrica, Facultad de Ingeniería, Universidad de San Carlos de Guatemala

Resumen: a los resultados del examen parcial el cual consiste en una parte teórica y otra en una parte práctica, cumpliendo con todos los objetivos se logran visualizar los resultados requeridos en el documento final.

B. Objetivos

Objetivo General

1. Realiza el examen parcial

Objetivo especifico

1. Mostrar el código
2. Mostrar los resultados obtenidos

C. comandos utilizados

```
import psycopg2
import matplotlib.pyplot as plt
import pandas as pd
from prettytable import PrettyTable
import random
conn = psycopg2.connect(
   dbname='estudiantes',
   user='postgres',
   password='1234',
   host='localhost',
   port='5432'
)
cur = conn.cursor()
def opcion1():


   while True:

      print("Menu de Opciones:")
      print("1. Ingresar datos")
      print("2. Eliminar")
      print("3. Actualizar")
      print("4. Salir")

      op = int(input("ingres opcion "))

      if op == 1:

            conn = psycopg2.connect(
               dbname='estudiantes',
               user='postgres',
               password='1234',
               host='localhost',
               port='5432'
            )
            cur = conn.cursor()

            nombre = input("Ingrese su nombre : ")
            edad = int(input("Ingrese su edad : "))

            genero = input("ingrese su e¿genero : ")
            direccion = input("Ingrese su direccion: ")
            try:
                  # Preparar la instrucción SQL para la inserción
                     Ins1 = 'INSERT INTO datos (nombre,edad,genero,direccion) VALUES (%s,%s,%s,%s);' # %s es un marcador de posición para el valor


                     Instruccion = cur.mogrify(Ins1, (nombre,edad,genero,direccion))

                     # Ejecutar la instrucción SQL
                     cur.execute(Instruccion)

                     # Confirmar la transacción
                     conn.commit()

            except psycopg2.Error as e:
                  print(f"Error durante la conexión a la DB. Consulte el error: {e}")

            finally:
                  # Cerrar el cursor y la conexión
                     cur.close()
                     conn.close()

      elif op == 2:
         try:
            # Establecer la conexión a la base de datos
            conn = psycopg2.connect(
                  dbname='estudiantes',
                  user='postgres',
                  password='1234',
                  host='localhost',
                  port='5432'
               )
```

```
            cur = conn.cursor()

            # Obtener el nombre para identificar el registro a eliminar
            nombre_a_eliminar = input("Ingrese el nombre a eliminar: ")

            # Construir y ejecutar la sentencia DELETE
            sentencia_delete = "DELETE FROM datos WHERE nombre = %s;"
            cur.execute(sentencia_delete, (nombre_a_eliminar,))

            # Confirmar la transacción
            conn.commit()
            print("Registro eliminado exitosamente.")

        except psycopg2.Error as e:
            print(f"Error durante la conexión a la DB. Consulte el error: {e}")

        finally:
            # Cerrar el cursor y la conexión
            cur.close()
            conn.close()
    elif op == 3:

        conn = psycopg2.connect(
            dbname='estudiantes',
            user='postgres',
            password='1234',
            host='localhost',
            port='5432'
        )
        cur = conn.cursor()

        try:
            # Solicitar al usuario el nombre del producto
            nombre_estudiante = input("Ingrese el nombre del estuante que desea actualizar: ")

            # Verificar si el producto existe antes de continuar
            cur.execute("SELECT COUNT(*) FROM datos WHERE nombre = %s;", (nombre_estudiante,))
            cantidad_productos = cur.fetchone()[0]

            if cantidad_productos == 0:
                print(f"No se encontró ningún estudiante con el nombre '{nombre_estudiante}'.")
            else:
                # Solicitar al usuario los nuevos valores
                nuevo_edad = (input("Ingrese la nueva edad: "))
                nueva_genero = (input("Ingrese el nuevo genero: "))
                nuevo_direccion = (input("Ingrese una nueva direccion: "))

                # Sentencia SQL de actualización
                sql_actualizacion = """
                    UPDATE datos
                    SET edad = %s, genero = %s , direccion = %s
                    WHERE nombre = %s;
                """

                # Ejecutar la actualización
                cur.execute(sql_actualizacion, (nuevo_edad, nueva_genero,nuevo_direccion ,nombre_estudiante))

                # Confirmar la transacción
                conn.commit()

                print(f"Se ha actualizado la información del estudiante '{nombre_estudiante}'.")

        except psycopg2.Error as e:
            print(f"Error durante la conexión a la DB. Consulte el error: {e}")

        finally:
            # Cerrar la conexión
            cur.close()
            conn.close()
    elif op == 4:

        break




def opcion2():

  while True:


    print("Menu de Opciones:")
    print("1. Ingresar gasto")
    print("2. ver resumen")
    print("3. Actualizar")
    print("4. Salir")


    op = int(input("ingres opcion "))

    if op == 1:
        conn = psycopg2.connect(
            dbname='gasto',
            user='postgres',
            password='1234',
            host='localhost',
            port='5432'
        )
```

```
        cur = conn.cursor()
        nombre_gasto= input("nombre del gasto : ")
        gasto = float(input("ingrese cantidad de gasto: "))
        presupuesto = float(input("ingrese su presupuesto actual : "))
        try:
                # Preparar la instrucción SQL para la inserción
                    Ins1 = 'INSERT INTO gasto1 (nombregasto,cantidad,presupuesto) VALUES (%s,%s,%s);' # %s es un marcador de posición para el valor


                    Instruccion = cur.mogrify(Ins1, (nombre_gasto,gasto,presupuesto))

                    # Ejecutar la instrucción SQL
                    cur.execute(Instruccion)

                    # Confirmar la transacción
                    conn.commit()

        except psycopg2.Error as e:
                print(f"Error durante la conexión a la DB. Consulte el error: {e}")

        finally:
                # Cerrar el cursor y la conexión
                cur.close()
                conn.close()
    elif op == 2:
      # Parámetros de conexión a la base de datos
      conn = psycopg2.connect(
        dbname='gasto',
        user='postgres',
        password='1234',
        host='localhost',
        port='5432'
      )
      cur = conn.cursor()

      # Nombre de la tabla que deseas consultar
      nombre_tabla = "gasto1"

      # Consulta SQL para obtener todos los datos de la tabla
      consulta_sql = f"SELECT * FROM {nombre_tabla};"

      # Ejecutar la consulta
      cur.execute(consulta_sql)

      # Obtener todos los resultados
      resultados = cur.fetchall()

      # Mostrar los resultados

      for fila in resultados:
        print(" id "," nombre "," cantidad ", " presupuesto ")
        print("   ",fila)

      # Cerrar la conexión
      cur.close()
      conn.close()
    elif op == 3:

      conn = psycopg2.connect(
          dbname='gasto',
          user='postgres',
          password='1234',
          host='localhost',
          port='5432'
        )
      cur = conn.cursor()

      try:
        # Solicitar al usuario el nombre del producto
        nombre_gasto= input("Ingrese el nombre del gasto que desea actualizar: ")

        # Verificar si el producto existe antes de continuar
        cur.execute("SELECT COUNT(*) FROM gasto1 WHERE nombregasto = %s;", (nombre_gasto,))
        cantidad_productos = cur.fetchone()[0]

        if cantidad_productos == 0:
          print(f"No se encontró ningún gasto con el nombre '{nombre_gasto}'.")
        else:
          # Solicitar al usuario los nuevos valores
          nuevo_cantidad = int(input("Ingrese la nueva cantidad: "))
          nueva_presupuesto = int(input("Ingrese el nuevo presupesuto: "))


          # Sentencia SQL de actualización
          sql_actualizacion = """
            UPDATE gasto1
            SET cantidad = %s, presupuesto = %s
            WHERE nombregasto = %s;
          """

          # Ejecutar la actualización
          cur.execute(sql_actualizacion, (nuevo_cantidad, nueva_presupuesto,nombre_gasto))

          # Confirmar la transacción
          conn.commit()

          print(f"Se ha actualizado la información del gasto '{nombre_gasto}'.")

      except psycopg2.Error as e:
        print(f"Error durante la conexión a la DB. Consulte el error: {e}")

      finally:
        # Cerrar la conexión
```

```
            cur.close()
            conn.close()
        elif op == 4:

            break



def opcion3():


    while True:

        print("Menu de Opciones:")
        print("1. Ingresar producto")
        print("2. Eliminar")
        print("3. Actualizar")
        print("4. Salir")


        op = int(input("ingres opcion "))

        if op == 1:


            conn = psycopg2.connect(
                dbname='produc',
                user='postgres',
                password='1234',
                host='localhost',
                port='5432'
                )
            cur = conn.cursor()
            nombre = input("Ingrese nombre del producto : ")
            precio = input("Ingrese el pprecio: ")

            cantidad = input("ingrese la cantidad : ")

            try:
                    # Preparar la instrucción SQL para la inserción
                        Ins1 = 'INSERT INTO productos (nombre,precio,cantidad) VALUES (%s,%s,%s);'  # %s es un marcador de posición para el valor


                        Instruccion = cur.mogrify(Ins1, (nombre,precio,cantidad))

                        # Ejecutar la instrucción SQL
                        cur.execute(Instruccion)

                        # Confirmar la transacción
                        conn.commit()

            except psycopg2.Error as e:
                    print(f"Error durante la conexión a la DB. Consulte el error: {e}")

            finally:
                    # Cerrar el cursor y la conexión
                        cur.close()
                        conn.close()
        elif op == 2:
            try:
                # Establecer la conexión a la base de datos

                conn = psycopg2.connect(
                dbname='produc',
                user='postgres',
                password='1234',
                host='localhost',
                port='5432'
                )
                cur = conn.cursor()
                # Obtener el nombre para identificar el registro a eliminar
                nombre_a_eliminar = input("Ingrese el nombre del producto a eliminar: ")

                # Construir y ejecutar la sentencia DELETE
                sentencia_delete = "DELETE FROM productos WHERE nombre = %s;"
                cur.execute(sentencia_delete, (nombre_a_eliminar,))

                # Confirmar la transacción
                conn.commit()
                print("Registro eliminado exitosamente.")

            except psycopg2.Error as e:
                print(f"Error durante la conexión a la DB. Consulte el error: {e}")

            finally:
                # Cerrar el cursor y la conexión
                cur.close()
                conn.close()
        elif op == 3:


            conn = psycopg2.connect(
                    dbname='produc',
                    user='postgres',
                    password='1234',
                    host='localhost',
                    port='5432'
                )
            cur = conn.cursor()

            try:
                # Solicitar al usuario el nombre del producto
                nombre_producto= input("Ingrese el nombre del producto que desea actualizar: ")

                # Verificar si el producto existe antes de continuar
                cur.execute("SELECT COUNT(*) FROM productos WHERE nombre = %s;", (nombre_producto,))
```

```
            cantidad_productos = cur.fetchone()[0]

            if cantidad_productos == 0:
                print(f"No se encontró ningún gasto con el nombre '{nombre_producto}'.")
            else:
                # Solicitar al usuario los nuevos valores
                nuevo_precio = int(input("Ingrese el nuevo precio: "))
                nueva_cantidad = int(input("Ingrese la nueva cantidad: "))


                # Sentencia SQL de actualización
                sql_actualizacion = """
                    UPDATE productos
                    SET precio = %s, cantidad = %s
                    WHERE nombre = %s;
                """

                # Ejecutar la actualización
                cur.execute(sql_actualizacion, (nuevo_precio, nueva_cantidad,nombre_producto))

                # Confirmar la transacción
                conn.commit()

                print(f"Se ha actualizado la información del producto '{nombre_producto}'.")

        except psycopg2.Error as e:
            print(f"Error durante la conexión a la DB. Consulte el error: {e}")

        finally:
            # Cerrar la conexión
            cur.close()
            conn.close()
    elif op == 4:

        break

def opcion4():


  while True:

    print("Menu de Opciones:")
    print("1. Ingresar datos")
    print("2. Eliminar")
    print("3. Actualizar")
    print("4. Salir")


    op = int(input("ingres opcion "))

    if op == 1:
        conn = psycopg2.connect(
            dbname='pedidos',
            user='postgres',
            password='1234',
            host='localhost',
            port='5432'
            )
        cur = conn.cursor()
        nombre = input("Ingrese el nombre del solicitante : ")
        nombre_pedido = input("Ingrese nombre del pedido : ")

        precio = input("ingrese el precio del pedido: ")
        cantidad = input("Ingrese la cantidad del pedido: ")
        try:
                # Preparar la instrucción SQL para la inserción
                    Ins1 = 'INSERT INTO pedido (nombre,nombrepedido,precio,cantidad) VALUES (%s,%s,%s,%s);'  # %s es un marcador de posición para el valor


                    Instruccion = cur.mogrify(Ins1, (nombre,nombre_pedido,precio,cantidad))

                    # Ejecutar la instrucción SQL
                    cur.execute(Instruccion)

                    # Confirmar la transacción
                    conn.commit()

        except psycopg2.Error as e:
                print(f"Error durante la conexión a la DB. Consulte el error: {e}")

        finally:
            # Cerrar el cursor y la conexión
                cur.close()
                conn.close()
    elif op == 2:
        try:
            # Establecer la conexión a la base de datos

            conn = psycopg2.connect(
            dbname='pedidos',
            user='postgres',
            password='1234',
            host='localhost',
            port='5432'
            )
            cur = conn.cursor()
            # Obtener el nombre para identificar el registro a eliminar
            nombre_a_eliminar = input("Ingrese el nombre  a eliminar: ")

            # Construir y ejecutar la sentencia DELETE
            sentencia_delete = "DELETE FROM pedido WHERE nombre = %s;"
            cur.execute(sentencia_delete, (nombre_a_eliminar,))
```

```
            # Confirmar la transacción
            conn.commit()
            print("Registro eliminado exitosamente.")

        except psycopg2.Error as e:
            print(f"Error durante la conexión a la DB. Consulte el error: {e}")

        finally:
            # Cerrar el cursor y la conexión
            cur.close()
            conn.close()
    elif op == 3:

        conn = psycopg2.connect(
            dbname='pedidos',
            user='postgres',
            password='1234',
            host='localhost',
            port='5432'
        )
        cur = conn.cursor()

        try:
            # Solicitar al usuario el nombre del producto
            nombre_pedido= input("Ingrese el nombre  que desea actualizar: ")

            # Verificar si el producto existe antes de continuar
            cur.execute("SELECT COUNT(*) FROM pedido WHERE nombre = %s;", (nombre_pedido,))
            cantidad_productos = cur.fetchone()[0]

            if cantidad_productos == 0:
                print(f"No se encontró ningún gasto con el nombre '{nombre_pedido}'.")
            else:
                # Solicitar al usuario los nuevos valores
                nuevo_pedido = (input("Ingrese el nuevo nombre del pedido: "))
                nueva_precio = int(input("Ingrese el nuevo precio: "))
                nueva_cantidad = int(input("Ingrese la nueva cantidad: "))


                # Sentencia SQL de actualización
                sql_actualizacion = """
                    UPDATE pedido
                    SET nombrepedido = %s, precio = %s , cantidad = %s
                    WHERE nombre = %s;
                """

                # Ejecutar la actualización
                cur.execute(sql_actualizacion, (nuevo_pedido, nueva_precio,nueva_cantidad,nombre_pedido))

                # Confirmar la transacción
                conn.commit()

                print(f"Se ha actualizado la información del pedido '{nombre_pedido}'.")

        except psycopg2.Error as e:
            print(f"Error durante la conexión a la DB. Consulte el error: {e}")

        finally:
            # Cerrar la conexión
            cur.close()
            conn.close()
    elif op == 4:

        break

def opcion5():

    while True:


        print("Menu de Opciones:")
        print("1. Ingresar venta")
        print("2. Informe")
        print("3. Tendencia")
        print("4. Analisis")
        print("5. Salir")


        op = int(input("ingres opcion : "))

        if op == 1:
            conn = psycopg2.connect(
                dbname='ventas',
                user='postgres',
                password='1234',
                host='localhost',
                port='5432'
            )
            cur = conn.cursor()
            nombre_gasto= input("nombre venta : ")
            gasto = float(input("ingrese cantidad : "))

            try:
                    # Preparar la instrucción SQL para la inserción
                    Ins1 = 'INSERT INTO venta (nombre,cantidad) VALUES (%s,%s);'  # %s es un marcador de posición para el valor


                    Instruccion = cur.mogrify(Ins1, (nombre_gasto,gasto))

                    # Ejecutar la instrucción SQL
                    cur.execute(Instruccion)
```

```
                # Confirmar la transacción
                conn.commit()

        except psycopg2.Error as e:
                print(f"Error durante la conexión a la DB. Consulte el error: {e}")

        finally:
                # Cerrar el cursor y la conexión
                cur.close()
                conn.close()
    elif op == 2:
      # Parámetros de conexión a la base de datos
      conn = psycopg2.connect(
         dbname='ventas',
         user='postgres',
         password='1234',
         host='localhost',
         port='5432'
      )
      cur = conn.cursor()

      # Nombre de la tabla que deseas consultar
      nombre_tabla = "venta"

      # Consulta SQL para obtener todos los datos de la tabla
      consulta_sql = f"SELECT * FROM {nombre_tabla};"

      # Ejecutar la consulta
      cur.execute(consulta_sql)

      # Obtener todos los resultados
      resultados = cur.fetchall()

      # Mostrar los resultados

      for fila in resultados:
         print(" id "," nombre "," cantidad ")
         print("   ",fila)

      # Cerrar la conexión
      cur.close()
      conn.close()
    elif op == 3:


         conn = psycopg2.connect(
         dbname='ventas',
         user='postgres',
         password='1234',
         host='localhost',
         port='5432'
         )
         cur = conn.cursor()


         # Consulta SQL para obtener datos de la tabla
         consulta_sql = "SELECT nombre, cantidad FROM venta;"
         cur.execute(consulta_sql)

         # Obtener resultados
         resultados = cur.fetchall()

         # Extraer columnas para el gráfico
         columna_x, columna_y = zip(*resultados)

         # Graficar los datos
         plt.plot(columna_x, columna_y, marker='o', linestyle='-')
         plt.title("Tendencia de ventas")
         plt.xlabel("Nombre de la Venta")
         plt.ylabel("Cantidad de ventas")

         # Mostrar el gráfico
         plt.show()

         # Cerrar la conexión
         cur.close()
         conn.close()


    elif op ==4:

      conn = psycopg2.connect(
         dbname='ventas',
         user='postgres',
         password='1234',
         host='localhost',
         port='5432'
         )
      cur = conn.cursor()
      # Consulta SQL para obtener datos de la tabla
      consulta_sql = "SELECT * FROM venta;"
      df = pd.read_sql_query(consulta_sql, conn)

      # Convertir la columna 'cantidad' a tipo numérico y manejar NaN
      df['cantidad'] = pd.to_numeric(df['cantidad'], errors='coerce')

      # Eliminar filas con valores nulos
      df.dropna(subset=['cantidad'], inplace=True)

      # Realizar análisis de datos (ejemplo: obtener estadísticas descriptivas)
      analisis_descriptivo = df.describe()

      # Mostrar estadísticas descriptivas
      print("Estadísticas Descriptivas:")
      print(analisis_descriptivo)

      # Visualizar los datos (ejemplo: histograma)
      df['cantidad'].plot(kind='hist', bins=10, title="Histograma")
      plt.show()
```

```
        # Cerrar la conexión
        conn.close()


    elif op == 5:

        break



def opcion6():


    while True:

        print("Menu de Opciones:")
        print("1. Ingresar datos de los senores")
        print("2. PTC")
        print("3. PT100")
        print("4. PT100")
        print("5. Gratica y Tabla ")
        print("6. Salir")


        op = int(input("ingres opcion : "))

        if op == 1:

            conn = psycopg2.connect(
                dbname='sensores',
                user='postgres',
                password='1234',
                host='localhost',
                port='5432'
            )
            cur = conn.cursor()
            nombre_sensor= input("nombre : ")
            dato = float(input("datos : "))

            try:
                    # Preparar la instrucción SQL para la inserción
                    Ins1 = 'INSERT INTO sensor (ntc,datos) VALUES (%s,%s);'  # %s es un marcador de posición para el valor


                    Instruccion = cur.mogrify(Ins1, (nombre_sensor,dato))

                    # Ejecutar la instrucción SQL
                    cur.execute(Instruccion)

                    # Confirmar la transacción
                    conn.commit()


                    nombre_sensor2= input("nombre : ")
                    dato2 = float(input("datos : "))


                # Preparar la instrucción SQL para la inserción

                    Ins2 = 'INSERT INTO sensor2 (ptc,datos) VALUES (%s,%s);'  # %s es un marcador de posición para el valor


                    Instruccion2 = cur.mogrify(Ins2, (nombre_sensor2,dato2))


                    # Ejecutar la instrucción SQL
                    cur.execute(Instruccion2)

                    # Confirmar la transacción
                    conn.commit()


                    nombre_sensor3= input("nombre : ")
                    dato3 = float(input("datos : "))


                # Preparar la instrucción SQL para la inserción

                    Ins3 = 'INSERT INTO sensor3 (pt100,datos) VALUES (%s,%s);'  # %s es un marcador de posición para el valor


                    Instruccion3 = cur.mogrify(Ins3, (nombre_sensor3,dato3))


                    # Ejecutar la instrucción SQL
                    cur.execute(Instruccion3)

                    # Confirmar la transacción
                    conn.commit()

            except psycopg2.Error as e:
                print(f"Error durante la conexión a la DB. Consulte el error: {e}")
```

```
        finally:
            # Cerrar el cursor y la conexión
            cur.close()
            conn.close()


    elif op == 2:
        conn = psycopg2.connect(
            dbname='sensores',
            user='postgres',
            password='1234',
            host='localhost',
            port='5432'
        )
        cur = conn.cursor()

        consulta_sql = "SELECT * FROM sensor;"
        df = pd.read_sql_query(consulta_sql, conn)

        # Convertir la columna 'cantidad' a tipo numérico y manejar NaN
        df['datos'] = pd.to_numeric(df['datos'], errors='coerce')

        # Eliminar filas con valores nulos
        df.dropna(subset=['datos'], inplace=True)

        # Realizar análisis de datos (ejemplo: obtener estadísticas descriptivas)
        analisis_descriptivo = df.describe()

        # Mostrar estadísticas descriptivas
        print("Estadísticas Descriptivas:")
        print(analisis_descriptivo)

        # Visualizar los datos (ejemplo: gráfico de dispersión)
        plt.scatter(df.index, df['datos'])
        plt.title("Gráfico de Dispersión")
        plt.xlabel("Índice de Datos")
        plt.ylabel("Cantidad")
        plt.show()




        # Mostrar los datos como una tabla utilizando prettytable



        # Cerrar la conexión
        conn.close()


    elif op == 3:
        conn = psycopg2.connect(
            dbname='sensores',
            user='postgres',
            password='1234',
            host='localhost',
            port='5432'
        )
        cur = conn.cursor()

        consulta_sql = "SELECT * FROM sensor2;"
        df = pd.read_sql_query(consulta_sql, conn)

        # Convertir la columna 'cantidad' a tipo numérico y manejar NaN
        df['datos'] = pd.to_numeric(df['datos'], errors='coerce')

        # Eliminar filas con valores nulos
        df.dropna(subset=['datos'], inplace=True)

        # Realizar análisis de datos (ejemplo: obtener estadísticas descriptivas)
        analisis_descriptivo = df.describe()

        # Mostrar estadísticas descriptivas
        print("Estadísticas Descriptivas:")
        print(analisis_descriptivo)

        # Visualizar los datos (ejemplo: gráfico de dispersión)
        plt.scatter(df.index, df['datos'])
        plt.title("Gráfico de Dispersión")
        plt.xlabel("Índice de Datos")
        plt.ylabel("Cantidad")
        plt.show()

        # Cerrar la conexión
        conn.close()

    elif op ==4:


        conn = psycopg2.connect(
            dbname='sensores',
            user='postgres',
            password='1234',
            host='localhost',
            port='5432'
        )
        cur = conn.cursor()

        consulta_sql = "SELECT * FROM sensor3;"
        df = pd.read_sql_query(consulta_sql, conn)

        # Convertir la columna 'cantidad' a tipo numérico y manejar NaN
        df['datos'] = pd.to_numeric(df['datos'], errors='coerce')

        # Eliminar filas con valores nulos
        df.dropna(subset=['datos'], inplace=True)

        # Realizar análisis de datos (ejemplo: obtener estadísticas descriptivas)
        analisis_descriptivo = df.describe()
```

```
    # Mostrar estadísticas descriptivas
    print("Estadísticas Descriptivas:")
    print(analisis_descriptivo)

    # Visualizar los datos (ejemplo: gráfico de dispersión)
    plt.scatter(df.index, df['datos'])
    plt.title("Gráfico de Dispersión")
    plt.xlabel("Índice de Datos")
    plt.ylabel("Cantidad")
    plt.show()

    # Cerrar la conexión
    conn.close()


elif op == 5:

    conn = psycopg2.connect(
            dbname='sensores',
            user='postgres',
            password='1234',
            host='localhost',
            port='5432'
        )
    cur = conn.cursor()
    consulta_sql = "SELECT * FROM sensor;"
    df = pd.read_sql_query(consulta_sql, conn)

    consulta_sql2 = "SELECT * FROM sensor2;"
    df2 = pd.read_sql_query(consulta_sql2, conn)

    consulta_sql3 = "SELECT * FROM sensor3;"
    df3 = pd.read_sql_query(consulta_sql3, conn)

    # Mostrar los datos como una tabla utilizando pandas
    print("Datos de la Tabla NTC:")
    print(df)
    print("Datos de la Tabla PTC:")
    print(df2)
    print("Datos de la Tabla PT100:")
    print(df3)

    # Cerrar la conexión
    conn.close()



elif op == 6:

      break
```

```
def opcion7():


    while True:

      buscar = input("Buscar Gusto : ")


      if buscar == "accion":


        conn = psycopg2.connect(
          dbname='peliculas',
          user='postgres',
          password='1234',
          host='localhost',
          port='5432'
        )
        cur = conn.cursor()
        consulta_sql = "SELECT * FROM pelicula;"
        df = pd.read_sql_query(consulta_sql, conn)



        # Mostrar los datos como una tabla utilizando pandas
        print("Peliculas de accion:")
        print(df)


        # Cerrar la conexión
        conn.close()

      elif buscar == "comedia":

        conn = psycopg2.connect(
          dbname='peliculas',
          user='postgres',
          password='1234',
          host='localhost',
          port='5432'
        )
        cur = conn.cursor()
        consulta_sql = "SELECT * FROM pelicula3;"
        df = pd.read_sql_query(consulta_sql, conn)



        # Mostrar los datos como una tabla utilizando pandas
        print("peliculas de comedia:")
        print(df)


        # Cerrar la conexión
        conn.close()

      elif buscar == "miedo":
```

```
            conn = psycopg2.connect(
                dbname='peliculas',
                user='postgres',
                password='1234',
                host='localhost',
                port='5432'
            )
            cur = conn.cursor()
            consulta_sql = "SELECT * FROM pelicula2;"
            df = pd.read_sql_query(consulta_sql, conn)



            # Mostrar los datos como una tabla utilizando pandas
            print("peliculas de miedo:")
            print(df)


            # Cerrar la conexión
            conn.close()


        elif buscar == "salir":

            break




def opcion8():

    while True:

        print("Menu de Opciones:")

        print("1. Google")
        print("2. Cementos Progresos")
        print("3. Intel")
        print("4. Gratica y Tabla ")
        print("5. Salir")


        op = int(input("ingres opcion : "))




        if op == 1:
            conn = psycopg2.connect(
                dbname='empresas',
                user='postgres',
                password='1234',
                host='localhost',
                port='5432'
            )
            cur = conn.cursor()

            consulta_sql = "SELECT * FROM empresa;"
            df = pd.read_sql_query(consulta_sql, conn)

            # Convertir la columna 'cantidad' a tipo numérico y manejar NaN
            df['datos'] = pd.to_numeric(df['datos'], errors='coerce')

            # Eliminar filas con valores nulos
            df.dropna(subset=['datos'], inplace=True)

            # Realizar análisis de datos (ejemplo: obtener estadísticas descriptivas)
            analisis_descriptivo = df.describe()

            # Mostrar estadísticas descriptivas
            print("Estadísticas Descriptivas:")
            print(analisis_descriptivo)

            # Visualizar los datos (ejemplo: gráfico de dispersión)
            plt.scatter(df.index, df['datos'])
            plt.title("Gráfico de Dispersión")
            plt.xlabel("Índice de Datos")
            plt.ylabel("Cantidad")
            plt.show()



            # Mostrar los datos como una tabla utilizando prettytable

            # Cerrar la conexión
            conn.close()


        elif op == 2:
            conn = psycopg2.connect(
                dbname='empresas',
                user='postgres',
                password='1234',
                host='localhost',
                port='5432'
            )
            cur = conn.cursor()

            consulta_sql = "SELECT * FROM empresa2;"
            df = pd.read_sql_query(consulta_sql, conn)

            # Convertir la columna 'cantidad' a tipo numérico y manejar NaN
            df['datos'] = pd.to_numeric(df['datos'], errors='coerce')

            # Eliminar filas con valores nulos
            df.dropna(subset=['datos'], inplace=True)

            # Realizar análisis de datos (ejemplo: obtener estadísticas descriptivas)
```

```
        analisis_descriptivo = df.describe()

        # Mostrar estadísticas descriptivas
        print("Estadísticas Descriptivas:")
        print(analisis_descriptivo)

        # Visualizar los datos (ejemplo: gráfico de dispersión)
        plt.scatter(df.index, df['datos'])
        plt.title("Gráfico de Dispersión")
        plt.xlabel("Índice de Datos")
        plt.ylabel("Cantidad")
        plt.show()

        # Cerrar la conexión
        conn.close()

    elif op ==3:


        conn = psycopg2.connect(
            dbname='empresas',
            user='postgres',
            password='1234',
            host='localhost',
            port='5432'
          )
        cur = conn.cursor()

        consulta_sql = "SELECT * FROM empresa3;"
        df = pd.read_sql_query(consulta_sql, conn)

        # Convertir la columna 'cantidad' a tipo numérico y manejar NaN
        df['datos'] = pd.to_numeric(df['datos'], errors='coerce')

        # Eliminar filas con valores nulos
        df.dropna(subset=['datos'], inplace=True)

        # Realizar análisis de datos (ejemplo: obtener estadísticas descriptivas)
        analisis_descriptivo = df.describe()

        # Mostrar estadísticas descriptivas
        print("Estadísticas Descriptivas:")
        print(analisis_descriptivo)

        # Visualizar los datos (ejemplo: gráfico de dispersión)
        plt.scatter(df.index, df['datos'])
        plt.title("Gráfico de Dispersión")
        plt.xlabel("Índice de Datos")
        plt.ylabel("Cantidad")
        plt.show()

        # Cerrar la conexión
        conn.close()


    elif op == 4:

        conn = psycopg2.connect(
            dbname='empresas',
            user='postgres',
            password='1234',
            host='localhost',
            port='5432'
          )
        cur = conn.cursor()
        consulta_sql = "SELECT * FROM empresa;"
        df = pd.read_sql_query(consulta_sql, conn)

        consulta_sql2 = "SELECT * FROM empresa2;"
        df2 = pd.read_sql_query(consulta_sql2, conn)

        consulta_sql3 = "SELECT * FROM empresa3;"
        df3 = pd.read_sql_query(consulta_sql3, conn)

        # Mostrar los datos como una tabla utilizando pandas
        print("Datos de la Tabla Google:")
        print(df)
        print("Datos de la Tabla Cementos progresos:")
        print(df2)
        print("Datos de la Tabla INTEL:")
        print(df3)

        # Cerrar la conexión
        conn.close()


    elif op == 5:

        break

def opcion9():

    while True:

        print("Menu de Opciones:")
        print("1. Ingresar productos")
        print("2. actualizar")
        print("3. vender")
        print("4. Generar informe de ventas en tabla")
        print("5. Salir")


        op = int(input("ingres opcion "))

        if op == 1:
            nombre = input("Ingrese su producto : ")
            precio = input("Ingrese su precio : ")

            cantidad = input("ingrese su cantidad : ")

            conn = psycopg2.connect(
                dbname='problema9',
```

```
            user='postgres',
            password='1234',
            host='localhost',
            port='5432'
        )
    cur = conn.cursor()

    try:
            # Preparar la instrucción SQL para la inserción
            Ins1 = 'INSERT INTO productos (nombre,precio,cantidad) VALUES (%s,%s,%s);'  # %s es un marcador de posición para el valor


            Instruccion = cur.mogrify(Ins1, (nombre,precio,cantidad))

            # Ejecutar la instrucción SQL
            cur.execute(Instruccion)

            # Confirmar la transacción
            conn.commit()

    except psycopg2.Error as e:
            print(f"Error durante la conexión a la DB. Consulte el error: {e}")

    finally:
        # Cerrar el cursor y la conexión
            cur.close()
            conn.close()
elif op == 2:

    conn = psycopg2.connect(
        dbname='problema9',
        user='postgres',
        password='1234',
        host='localhost',
        port='5432'
        )
    cur = conn.cursor()

    try:
        # Solicitar al usuario el nombre del producto
        nombre_producto = input("Ingrese el nombre del producto que desea actualizar: ")

        # Verificar si el producto existe antes de continuar
        cur.execute("SELECT COUNT(*) FROM productos WHERE nombre = %s;", (nombre_producto,))
        cantidad_productos = cur.fetchone()[0]

        if cantidad_productos == 0:
            print(f"No se encontró ningún producto con el nombre '{nombre_producto}'.")
        else:
            # Solicitar al usuario los nuevos valores
            nuevo_precio = float(input("Ingrese el nuevo precio: "))
            nueva_cantidad = int(input("Ingrese la nueva cantidad: "))

            # Sentencia SQL de actualización
            sql_actualizacion = """
                UPDATE productos
                SET precio = %s, cantidad = %s
                WHERE nombre = %s;
            """

            # Ejecutar la actualización
            cur.execute(sql_actualizacion, (nuevo_precio, nueva_cantidad, nombre_producto))

            # Confirmar la transacción
            conn.commit()

            print(f"Se ha actualizado la información del producto '{nombre_producto}'.")

    except psycopg2.Error as e:
        print(f"Error durante la conexión a la DB. Consulte el error: {e}")

    finally:
        # Cerrar la conexión
        cur.close()
        conn.close()

elif op == 3:

    conn = psycopg2.connect(
        dbname='problema9',
        user='postgres',
        password='1234',
        host='localhost',
        port='5432'
        )
    cur = conn.cursor()

    try:
        # Solicitar al usuario el nombre del producto
        nombre_producto = input("Ingrese el nombre del producto que desea vender: ")

        # Verificar si el producto existe antes de continuar
        cur.execute("SELECT COUNT(*) FROM productos
```

```
WHERE nombre = %s;", (nombre_producto,))
            cantidad_productos = cur.fetchone()[0]

            if cantidad_productos == 0:
                print(f"No se encontró ningún producto con el nombre '{nombre_producto}'.")
            else:
                # Solicitar al usuario la cantidad que desea vender
                cantidad_a_vender = int(input("Ingrese la cantidad que desea vender: "))

                # Verificar si hay suficiente cantidad en el inventario
                cur.execute("SELECT cantidad FROM productos WHERE nombre = %s;", (nombre_producto,))
                cantidad_actual = int(cur.fetchone()[0])

                if cantidad_a_vender > cantidad_actual:
                    print(f"No hay suficiente cantidad en inventario para vender {cantidad_a_vender} unidades.")
                else:
                    # Actualizar la cantidad en el inventario
                    nueva_cantidad = cantidad_actual - cantidad_a_vender
                    cur.execute("UPDATE productos SET cantidad = %s WHERE nombre = %s;", (nueva_cantidad, nombre_producto))

                    # Confirmar la transacción
                    conn.commit()

                    print(f"Se han vendido {cantidad_a_vender} unidades del producto '{nombre_producto}'. Nuevo inventario: {nueva_cantidad} unidades.")

        except psycopg2.Error as e:
            print(f"Error durante la conexión a la DB. Consulte el error: {e}")

        finally:
            # Cerrar la conexión
            cur.close()
            conn.close()

    elif op == 4:

        conn = psycopg2.connect(
            dbname='problema9',
            user='postgres',
            password='1234',
            host='localhost',
            port='5432'
        )
        cur = conn.cursor()
        consulta_sql = "SELECT * FROM productos;"
        df = pd.read_sql_query(consulta_sql, conn)

        # Mostrar los datos como una tabla utilizando pandas
        print("Informes de ventas:")
        print(df)


        # Cerrar la conexión
        conn.close()
    elif op == 5:

        break

def opcion10():



    while True:

        buscar = input("Informacion del producto  : ")


        if buscar == "block":


            conn = psycopg2.connect(
                dbname='problema10',
                user='postgres',
                password='1234',
                host='localhost',
                port='5432'
            )
            cur = conn.cursor()
            try:
                # Consulta SQL para obtener la cantidad de arena
                consulta_sql = "SELECT cantidad FROM optimizar WHERE nombre = 'arena';"

                # Ejecutar la consulta
                cur.execute(consulta_sql)
                cantidad_arena_str = cur.fetchone()[0]

                # Convertir la cantidad de arena a entero
                cantidad_arena = int(cantidad_arena_str)

                # Calcular la cantidad de bloques que se pueden fabricar
                bloques_disponibles = cantidad_arena // 2  # Usamos "//" para la división entera

                # Mostrar el resultado
                print(f"Con {cantidad_arena} unidades de arena, puedes fabricar {bloques_disponibles} bloques.")

            except psycopg2.Error as e:
                print(f"Error durante la conexión a la base de datos. Consulte el error: {e}")

            finally:
```

```
            # Cerrar el cursor y la conexión
            cur.close()
            conn.close()



        elif buscar == "salir":

            break
def opcion11():
    while True:

        print("Menu de Opciones:")
        print("1. Desplegar lista de canciones")
        print("2. Buscar por artita ")
        print("3. Buscar por cancion")
        print("4. salir")



        op = int(input("ingres opcion : "))

        if op == 1:
            conn = psycopg2.connect(
                dbname='canciones',
                user='postgres',
                password='1234',
                host='localhost',
                port='5432'
            )
            cur = conn.cursor()
            consulta_sql = "SELECT * FROM tablacancion;"
            df = pd.read_sql_query(consulta_sql, conn)
            # Mostrar los datos como una tabla utilizando pandas
            print("Lista de canciones :")
            print(df)



            # Cerrar la conexión
            conn.close()
        elif op == 2:

            conn = psycopg2.connect(
                dbname='canciones',
                user='postgres',
                password='1234',
                host='localhost',
                port='5432'
            )
            cur = conn.cursor()
            artista_buscar = input('Ingrese el nombre del artista: ')

            try:
                # Consulta SQL para obtener información del artista
                consulta_sql = f"SELECT * FROM tablacancion WHERE artista = '{artista_buscar}';"

                # Ejecutar la consulta
                cur.execute(consulta_sql)

                # Obtener los resultados
                resultados = cur.fetchall()

                if resultados:
                    for resultado in resultados:
                        print(f"Artista: {resultado[1]}\nCanción: {resultado[2]}\nLetra:\n{resultado[3]}\n")
                else:
                    print(f"No se encontraron resultados para el artista {artista_buscar}.")

            except psycopg2.Error as e:
                print(f"Error durante la conexión a la base de datos. Consulte el error: {e}")

            finally:
                # Cerrar el cursor y la conexión
                cur.close()
                conn.close()









        elif op == 3:

            conn = psycopg2.connect(
                dbname='canciones',
                user='postgres',
                password='1234',
                host='localhost',
                port='5432'
            )
            cur = conn.cursor()
            # Solicitar al usuario que ingrese el nombre de la canción



            cancion_buscar = input('Ingrese el nombre de la canción: ')

            try:
                # Consulta SQL para obtener información de la canción
                consulta_sql = f"SELECT * FROM tablacancion WHERE cancion = '{cancion_buscar}';"

                # Ejecutar la consulta
                cur.execute(consulta_sql)
```

```
            # Obtener los resultados
            resultados = cur.fetchall()

            if resultados:
                for resultado in resultados:
                    print(f"Artista: {resultado[1]}\nCanción: {resultado[2]}\nLetra:\n{resultado[3]}\n")
            else:
                print(f"No se encontraron resultados para la canción {cancion_buscar}.")

        except psycopg2.Error as e:
            print(f"Error durante la conexión a la base de datos. Consulte el error: {e}")

        finally:
            # Cerrar el cursor y la conexión
            cur.close()
            conn.close()

    elif op == 4:

        break

def opcion12():

    while True:

        print("Menu de Opciones:")
        print("1. Jugar")
        print("2. Instruciones ")
        print("3. Ver Preguntas")
        print("4. salir")



        op = int(input("ingres opcion : "))

        if op == 1:


            conn = psycopg2.connect(
                    dbname='problema12',
                    user='postgres',
                    password='1234',
                    host='localhost',
                    port='5432'
                )
            cur = conn.cursor()
            # Realiza una consulta para obtener preguntas y respuestas
            cur.execute("SELECT pregunta, respuesta FROM juego;")
            preguntas_respuestas = cur.fetchall()

            # Inicializa el puntaje del usuario
            puntaje = 0
            puntaje_objetivo = 5

            # Baraja aleatoriamente las preguntas
            random.shuffle(preguntas_respuestas)

            # Presenta preguntas hasta que el usuario alcance el puntaje objetivo
            for pregunta, respuesta in preguntas_respuestas:
                print(f"Pregunta: {pregunta}")
                respuesta_usuario = input("Respuesta (Verdadero o Falso): ").lower()

                if respuesta_usuario == respuesta.lower():
                    print(" Ganaste 1 punto.")
                    puntaje += 1
                else:
                    print("Incorrecto. Perdistes puntos.")

                print(f"puntaje actual: {puntaje}")

                if puntaje == puntaje_objetivo:
                    print(f"Felicidades, tu puntaje es {puntaje_objetivo} puntos.")
                    break

            # Cierra la conexión a la base de datos
            cur.close()
            conn.close()
        elif op == 2:

            print(" ")
            print(" Las instrcciones son : \n se presentaran una serie de pregunatas \n cada pregunta si aciertas sumaras un punto \n de lo contrario perderas 1 punto tienes 3 vidas")
            print(" ")
        elif op == 3:
            conn = psycopg2.connect(
                    dbname='problema12',
                    user='postgres',
                    password='1234',
                    host='localhost',
                    port='5432'
                )
            cur = conn.cursor()

            consulta_sql = "SELECT * FROM juego;"
            df = pd.read_sql_query(consulta_sql, conn)
            # Mostrar los datos como una tabla utilizando pandas
            print("Lista de preguntas :")
            print(df)

        elif op == 4:
            break



while True:
```

```
exit()

    print("Menu de Opciones:")
    print("1.Programa de registro de estudiantes")
    print("2.Programa de seguimiento de presupuesto personal gastos")
    print("3.Programa de gestión de inventario")
    print("4.Programa de seguimiento de pedidos")
    print("5.Programa de monitoreo de ventas")
    print("6.Programa de análisis de datos de sensores")
    print("7.Sistema de recomendación de películas")
    print("8.Programa de análisis financiero")
    print("9.Sistema de gestión de inventario")
    print("10.Sistema de planificación de producción")
    print("11. Buscador de canciones")
    print("12. Concurso")
    print("13. Salir")


    seleccion = int(input("ingrese una opcion : "))

    if seleccion == 1:

        opcion1()

    elif seleccion == 2:

       opcion2()

    elif seleccion == 3:
        opcion3()

    elif seleccion == 4:
        opcion4()

    elif seleccion == 5:
        opcion5()

    elif seleccion == 6:
        opcion6()

    elif seleccion == 7:
        opcion7()

    elif seleccion == 8:
        opcion8()

    elif seleccion == 9:
        opcion9()

    elif seleccion == 10:
        opcion10()
    elif seleccion == 11:
        opcion11()

    elif seleccion == 12:
        opcion12()
    elif seleccion == 13:
```

C. Resultados imágenes

```
while True:


    print("Menu de Opciones:")
    print("1.Programa de registro de estudiantes")
    print("2.Programa de seguimiento de presupuesto personal gastos")
    print("3.Programa de gestión de inventario")
    print("4.Programa de seguimiento de pedidos")
    print("5.Programa de monitoreo de ventas")
    print("6.Programa de análisis de datos de sensores")
    print("7.Sistema de recomendación de películas")
    print("8.Programa de análisis financiero")
    print("9.Sistema de gestión de inventario")
    print("10.Sistema de planificación de producción")
    print("11. Buscador de canciones")
    print("12. Concurso")
    print("13. Salir")


    seleccion = int(input("ingrese una opcion : "))

    if seleccion == 1:

        opcion1()

    elif seleccion == 2:

       opcion2()

    elif seleccion == 3:
        opcion3()

    elif seleccion == 4:
        opcion4()

    elif seleccion == 5:
        opcion5()

    elif seleccion == 6:
        opcion6()

    elif seleccion == 7:
        opcion7()

    elif seleccion == 8:
```

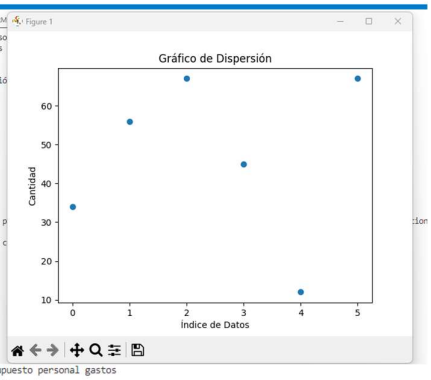
```
PS C:\Users\sergi\OneDrive\Desktop\examen parcial> & C:/Users/sergi/App
Menu de Opciones:
1.Programa de registro de estudiantes
2.Programa de seguimiento de presupuesto personal gastos
3.Programa de gestión de inventario
4.Programa de seguimiento de pedidos
5.Programa de monitoreo de ventas
6.Programa de análisis de datos de sensores
7.Sistema de recomendación de películas
8.Programa de análisis financiero
9.Sistema de gestión de inventario
10.Sistema de planificación de producción
11. Buscador de canciones
12. Concurso
13. Salir
ingrese una opcion : 1
Menu de Opciones:
1. Ingresar datos
2. Eliminar
3. Actualizar
4. Salir
ingres opcion 1
Ingrese su nombre : saul
Ingrese su edad : 45
ingrese su e¿genero : hombre
Ingrese su direccion: joyas
```

```
Ingrese su direccion: joyas
Menu de Opciones:
1. Ingresar datos
2. Eliminar
3. Actualizar
4. Salir
ingres opcion 4
Menu de Opciones:
1.Programa de registro de estudiantes
2.Programa de seguimiento de presupuesto personal gastos
3.Programa de gestión de inventario
4.Programa de seguimiento de pedidos
5.Programa de monitoreo de ventas
6.Programa de análisis de datos de sensores
7.Sistema de recomendación de películas
8.Programa de análisis financiero
9.Sistema de gestión de inventario
10.Sistema de planificación de producción
11. Buscador de canciones
12. Concurso
13. Salir
ingrese una opcion : 2
Menu de Opciones:
1. Ingresar gasto
2. ver resumen
3. Actualizar
4. Salir
ingres opcion 1
nombre del gasto : cemento
ingrese cantidad de gasto: 23
ingrese su presupuesto actual : 1234
Menu de Opciones:
1. Ingresar gasto
2. ver resumen
3. Actualizar
4. Salir
ingres opcion 4
Menu de Opciones:
1.Programa de registro de estudiantes
2.Programa de seguimiento de presupuesto personal gastos
3.Programa de gestión de inventario
4.Programa de seguimiento de pedidos
5.Programa de monitoreo de ventas
6.Programa de análisis de datos de sensores
7.Sistema de recomendación de películas
8.Programa de análisis financiero
9.Sistema de gestión de inventario
10.Sistema de planificación de producción
11. Buscador de canciones
12. Concurso
13. Salir
ingrese una opcion : 3
Menu de Opciones:
1. Ingresar producto
2. Eliminar
3. Actualizar
4. Salir
ingres opcion 1
Ingrese nombre del producto : carne
Ingrese el pprecio: 45
ingrese la cantidad : 67
```

```
Menu de Opciones:
1. Ingresar producto
2. Eliminar
3. Actualizar
4. Salir
ingres opcion 4
Menu de Opciones:
1.Programa de registro de estudiantes
2.Programa de seguimiento de presupuesto personal gastos
3.Programa de gestión de inventario
4.Programa de seguimiento de pedidos
5.Programa de monitoreo de ventas
6.Programa de análisis de datos de sensores
7.Sistema de recomendación de películas
8.Programa de análisis financiero
9.Sistema de gestión de inventario
10.Sistema de planificación de producción
11. Buscador de canciones
12. Concurso
13. Salir
ingrese una opcion : 4
Menu de Opciones:
1. Ingresar datos
2. Eliminar
3. Actualizar
4. Salir
ingres opcion 1
Ingrese el nombre del solicitante : heli
Ingrese nombre del pedido : rtx
ingrese el precio del pedido: 345
Ingrese la cantidad del pedido: 2
```

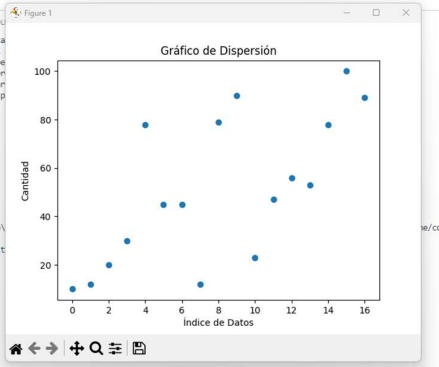
```
Menu de Opciones:
1.Programa de registro de estudiantes
2.Programa de seguimiento de presupuesto personal gastos
3.Programa de gestión de inventario
4.Programa de seguimiento de pedidos
5.Programa de monitoreo de ventas
6.Programa de análisis de datos de sensores
7.Sistema de recomendación de películas
8.Programa de análisis financiero
9.Sistema de gestión de inventario
10.Sistema de planificación de producción
11. Buscador de canciones
12. Concurso
13. Salir
ingrese una opcion : 5
Menu de Opciones:
1. Ingresar venta
2. Informe
3. Tendencia
4. Analisis
5. Salir
ingres opcion : 1
nombre venta : carner
ingrese cantidad : 3
Menu de Opciones:
1. Ingresar venta
2. Informe
3. Tendencia
4. Analisis
5. Salir
ingres opcion :
```

```
6.Programa de análisis de datos de senso
7.Sistema de recomendación de películas
8.Programa de análisis financiero
9.Sistema de gestión de inventario
10.Sistema de planificación de producció
11. Buscador de canciones
12. Concurso
13. Salir
ingrese una opcion : 6
Menu de Opciones:
1. Ingresar datos de los senores
2. PTC
3. PT100
4. PT100
5. Gratica y Tabla
6. Salir
ingres opcion : 2
c:\Users\sergi\OneDrive\Desktop\examen p
se consider using SQLAlchemy.
  df = pd.read_sql_query(consulta_sql, c
Estadísticas Descriptivas:
              id      datos
count  6.000000   6.000000
mean   3.500000  46.833333
std    1.870829  21.348692
min    1.000000  12.000000
25%    2.250000  36.750000
50%    3.500000  50.500000
75%    4.750000  64.250000
max    6.000000  67.000000
```




```
2.Programa de seguimiento de presupuesto personal gastos
3.Programa de gestión de inventario
4.Programa de seguimiento de pedidos
5.Programa de monitoreo de ventas
6.Programa de análisis de datos de sensores
7.Sistema de recomendación de películas
8.Programa de análisis financiero
9.Sistema de gestión de inventario
10.Sistema de planificación de producción
11. Buscador de canciones
12. Concurso
13. Salir
ingrese una opcion : 7
Buscar Gusto : accion
c:\Users\sergi\OneDrive\Desktop\examen parcial\main.py:1000: UserWarning: pandas only supports SQLAlchemy connectab
ase consider using SQLAlchemy.
  df = pd.read_sql_query(consulta_sql, conn)
Películas de accion:
   id          nombre  accion
0   1      El Padrino  accion
1   2  El Padrino 2  accion
2   3  El Padrino 3  accion
3   4  Durode matar  accion
4   5     Jonh Wick  accion
5   6   Jonh Wick 2  accion
6   7   Jonh Wick 3  accion
7   8   Jonh Wick 4  accion
8   9        matrix  accion
9  10      matrix 2  accion
10 11      matrix 3  accion
Buscar Gusto :
```

```
5.Programa de monitoreo de venta
6.Programa de análisis de datos
7.Sistema de recomendación de pe
8.Programa de análisis financier
9.Sistema de gestión de inventar
10.Sistema de planificación de p
11. Buscador de canciones
12. Concurso
13. Salir
ingrese una opcion : 8
Menu de Opciones:
1. Google
2. Cementos Progresos
3. Intel
4. Gratica y Tabla
5. Salir
ingres opcion : 1
c:\Users\sergi\OneDrive\Desktop\
ase consider using SQLAlchemy.
  df = pd.read_sql_query(consult
Estadísticas Descriptivas:
              id      datos
count  17.000000  17.000000
mean    9.000000  51.000000
std     5.049752  30.242768
min     1.000000  10.000000
25%     5.000000  23.000000
50%     9.000000  47.000000
75%    13.000000  78.000000
max    17.000000  100.000000
```



```
1.Programa de registro de estudiantes
2.Programa de seguimiento de presupuesto personal gastos
3.Programa de gestión de inventario
4.Programa de seguimiento de pedidos
5.Programa de monitoreo de ventas
6.Programa de análisis de datos de sensores
7.Sistema de recomendación de películas
8.Programa de análisis financiero
9.Sistema de gestión de inventario
10.Sistema de planificación de producción
11. Buscador de canciones
12. Concurso
13. Salir
ingrese una opcion : 9
Menu de Opciones:
1. Ingresar productos
2. actualizar
3. vender
4. Generar informe de ventas en tabla
5. Salir
ingres opcion 1
Ingrese su producto : pollo
Ingrese su precio : 34
ingrese su cantidad : 23
Menu de Opciones:
1. Ingresar productos
2. actualizar
3. vender
4. Generar informe de ventas en tabla
5. Salir
ingres opcion
```

```
4. Generar informe de ventas en tabla
5. Salir
ingres opcion 1
Ingrese su producto : pollo
Ingrese su precio : 34
ingrese su cantidad : 23
Menu de Opciones:
1. Ingresar productos
2. actualizar
3. vender
4. Generar informe de ventas en tabla
5. Salir
ingres opcion 5
Menu de Opciones:
1.Programa de registro de estudiantes
2.Programa de seguimiento de presupuesto personal gastos
3.Programa de gestión de inventario
4.Programa de seguimiento de pedidos
5.Programa de monitoreo de ventas
6.Programa de análisis de datos de sensores
7.Sistema de recomendación de películas
8.Programa de análisis financiero
9.Sistema de gestión de inventario
10.Sistema de planificación de producción
11. Buscador de canciones
12. Concurso
13. Salir
ingrese una opcion : 10
Informacion del producto  : block
Con 12 unidades de arena, puedes fabricar 6 bloques.
Informacion del producto  :
```

```
1. Desplegar lista de canciones
2. Buscar por artita
3. Buscar por cancion
4. salir
ingres opcion : 1
c:\Users\sergi\OneDrive\Desktop\examen parcial\main.py:1496: UserWarning: pandas only supports SQLAlchemy co
ase consider using SQLAlchemy.
  df = pd.read_sql_query(consulta_sql, conn)
Lista de canciones :
    id       artista                          cancion             letra
0    1  Ricardo Arjona                            amor  lettra de cancion
1    2  Ricardo Arjona                     Fuiste tu  lettra de cancion
2    3  Ricardo Arjona  Mi novia se me esta poniendo vieja  lettra de cancion
3    4  Ricardo Arjona                       Te quiero  lettra de cancion
4    5  Ricardo Arjona          Si tu no existieras  lettra de cancion
5    6  Alejandro Sanz                           Amiga  lettra de cancion
6    7  Alejandro Sanz                           Muero  lettra de cancion
7    8  Alejandro Sanz                           Desde  lettra de cancion
8    9  Alejandro Sanz                          Cuando  lettra de cancion
9   10  Alejandro Sanz                      Mi soledad  lettra de cancion
10  11   Laura Pausini                          Se fue  lettra de cancion
11  12   Laura Pausini                       Te amare  lettra de cancion
12  13   Laura Pausini                         Viveme  lettra de cancion
13  14   Laura Pausini                           Sino  lettra de cancion
14  15   Laura Pausini                             Yo  lettra de cancion
Menu de Opciones:
1. Desplegar lista de canciones
2. Buscar por artita
3. Buscar por cancion
4. salir
ingres opcion :
```

```
13  14   Laura Pausini                           Sino  lettra de cancion
14  15   Laura Pausini                             Yo  lettra de cancion
Menu de Opciones:
1. Desplegar lista de canciones
2. Buscar por artita
3. Buscar por cancion
4. salir
ingres opcion : 4
Menu de Opciones:
1.Programa de registro de estudiantes
2.Programa de seguimiento de presupuesto personal gastos
3.Programa de gestión de inventario
4.Programa de seguimiento de pedidos
5.Programa de monitoreo de ventas
6.Programa de análisis de datos de sensores
7.Sistema de recomendación de películas
8.Programa de análisis financiero
9.Sistema de gestión de inventario
10.Sistema de planificación de producción
11. Buscador de canciones
12. Concurso
13. Salir
ingrese una opcion : 12
Menu de Opciones:
1. Jugar
2. Instruciones
3. Ver Preguntas
4. salir
ingres opcion : 1
Pregunta: El mundo es redondo ?
Respuesta (Verdadero o Falso):
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

```
1. Jugar
2. Instruciones
3. Ver Preguntas
4. salir
ingres opcion : 1
Pregunta: El mar es salado?
Respuesta (Verdadero o Falso): verdadero
 Ganaste 1 punto.
puntaje actual: 1
Pregunta: las plantas necesitan agua?
Respuesta (Verdadero o Falso): falso
Incorrecto. Perdistes puntos.
puntaje actual: 1
Pregunta: El mundo es redondo ?
Respuesta (Verdadero o Falso): falso
Incorrecto. Perdistes puntos.
puntaje actual: 1
Pregunta: los gatos comen ratones?
Respuesta (Verdadero o Falso): falso
Incorrecto. Perdistes puntos.
puntaje actual: 1
Pregunta: los pericos vuelan?
Respuesta (Verdadero o Falso): falso
Incorrecto. Perdistes puntos.
puntaje actual: 1
Menu de Opciones:
1. Jugar
2. Instruciones
3. Ver Preguntas
4. salir
ingres opcion :
```

Nombre: Heli Saul Vasquez Gomez
Carne: 201700852
CUI: 2852625480106

Proyectos APicados

serie I

Pregunta 2
R// a) save()

Pregunta 4
R// a) sqrt()

Pregunta 6
R// a) round()

Pregunta 8
R// a) mean()

Pregunta 10
R// b) xcorr()

Pregunta 12
R// a) size()

Pregunta 14
R// a) mean()

Pregunta 16
R// a) round()

Pregunta 18
R// b) fft()

Pregunta 20
R// a) save()

Pregunta 22
R// c) ps9z

Pregunta 24
c) R// 11

Pregunta 26
a) CREATE TABEE

Pregunta 28
c) INSERT INTO

Pregunta 30
R// a) UPDATE

Pregunta 32
R// c) MATZ.conse

Pregunta 34
a) CREATE DATABASE

Pregunta 36
R// b)

Escaneado con Cam

Serie 2

Pregunta 2
R// Es un campo o conjunto de campos que posee dos propiedades.

Pregunta 4
R// se utiliza
DELETE FROM nombredelatabla;

Pregunta 6
R// se utiliza para filtrar los resultados de una Consulta.

Pregunta 8
R// Es una tabla virtual basada en los resultados de una Consulta.

Pregunta 10
R// Es un bloque de codigo que realiza una tarea

Pregunta 12
R// se tiene dos matrices
A y C
A = [1, 2]
C = [3, 4]
multiplica = A * C

Pregunta 14
R// se utiliza el Comando mean()

Pregunta 16
R// se calcula con el comando sqrt()

Pregunta 18
R// se realiza utilizando el comando max()

Pregunta 20
R// Con el comando rand()

Escaneado con Cam

Pregunta 38
R// a)

Pregunta 40
R// a)

Pregunta 42
R// a)

Pregunta 44
R// d)

Pregunta 46
R// b)

Pregunta 48
R// a)

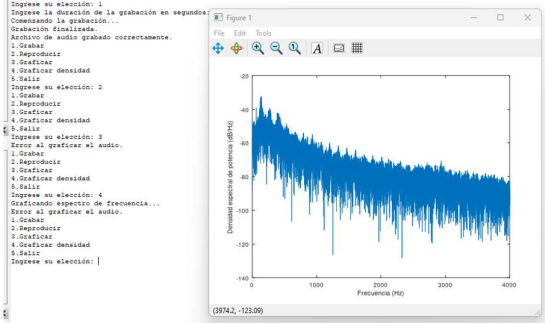
Pregunta 50
d) d)

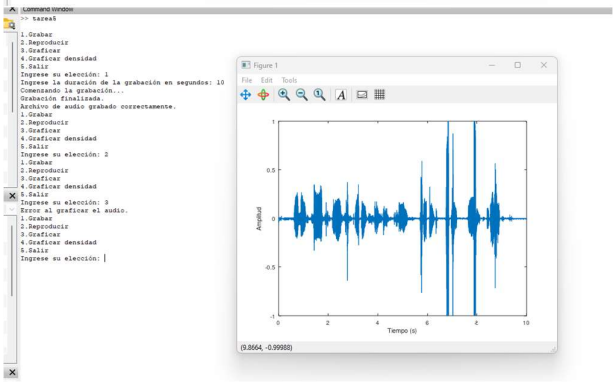
Escaneado con Cam

```
if(exist('OCTAVE_VERSION','builtin')~=0)
 pkg load signal;
end
 opcion = 0;
while opcion ~=5
  disp('1.Grabar')
  disp('2.Reproducir')
  disp('3.Graficar')
  disp('4.Graficar densidad')
  disp('5.Salir')
  opcion = input('Ingrese su elección: ');
switch opcion
 case 1
 % Grabación de audio
try
 duracion = input('Ingrese la duración de la grabación en segundos: ');
 disp('Comenzando la grabación...');
 recObj = audiorecorder;
 recordblocking(recObj, duracion);
 disp('Grabación finalizada.');
 data = getaudiodata(recObj);
 audiowrite('audio.wav', data, recObj.SampleRate);
 disp('Archivo de audio grabado correctamente.');
 catch
 disp('Error al grabar el audio.');
end
 case 2
try
   [data, fs] = audioread('audio.wav');
   sound(data, fs);
 catch
   disp('Error al reproducir el audio.');
end
 case 3
 % Gráfico de audio
try
 [data, fs] = audioread('audio.wav');
 tiempo = linspace(0, length(data)/fs, length(data));
 plot(tiempo, data);
 xlabel('Tiempo (s)');
 ylabel('Amplitud');
 tite('Audio');
 catch
 disp('Error al graficar el audio.');
end
 case 4
 % Graficando espectro de frecuencia
try
 disp('Graficando espectro de frecuencia...');
 [audio, Fs] = audioread('audio.wav'); % Lee la señal desde el archivo .wav
 N = length(audio); % Número de muestras de la señal
 f=linspace(0, Fs/2, N/2+1); % Vector de frecuencias
```

```
>> tarea5

1.Grabar
2.Reproducir
3.Graficar
4.Graficar densidad
5.Salir
Ingrese su elección: 1
Ingrese la duración de la grabación en segundos: 10
Comenzando la grabación...
Grabación finalizada.
Archivo de audio grabado correctamente.
1.Grabar
2.Reproducir
3.Graficar
4.Graficar densidad
5.Salir
Ingrese su elección: 2
1.Grabar
2.Reproducir
3.Graficar
4.Graficar densidad
5.Salir
Ingrese su elección:
```





conclusiones

D.

1. Se logro realizar todos los programas en el lenguaje python
2. Se logro ingresar toda la información en la base de datos postgres Pgadmin
3. Se ejecutaron de manera correcta todos los programas

E. *Código*

https://github.com/helisaul/Proyec_com_apli_AIE.git