```
#!/usr/bin/env python3
import psycopg2

##########################################################
##  Database Connection
##########################################################

'''
Connect to the database using the connection string
'''
def openConnection():
    # connection parameters - ENTER YOUR LOGIN AND PASSWORD HERE
    userid = "y24s2c9120_abho0299"
    passwd = "Sawankdaa@5603"
    myHost = "awsprddbs4836.shared.sydney.edu.au"

    # Create a connection to the database
    conn = None
    try:
        # Parses the config file and connects using the connect string
        conn = psycopg2.connect(database=userid,
                                user=userid,
                                password=passwd,
                                host=myHost)

    except psycopg2.Error as sqle:
        print("psycopg2.Error : " + sqle.pgerror)

    # return the connection to use
    return conn

'''
Validate staff based on username and password
'''
def checkLogin(login, password):
    # Call the checkLogin function
    conn = None
    try:
        # Establish connection to your PostgreSQL database
        conn = openConnection()
        cursor = conn.cursor()

        # Call the stored function using SELECT
        cursor.execute("SELECT * FROM checkLogin(%s, %s)", (login, password))
        result = cursor.fetchone()  # Fetch the result

        if result:
            # If the result is not empty, unpack the returned values
            username, first_name, last_name, email = result
            return username, first_name, last_name, email
        else:
            # If no result is found, return None
            return None

    except psycopg2.Error as sqle:
        print("psycopg2.Error : " + sqle.pgerror)

    finally:
        if conn is not None:
```

```
            cursor.close()
            conn.close()


'''
List all the associated admissions records in the database by staff
'''
def findAdmissionsByAdmin(login):
    conn = None
    try:
        conn = openConnection()
        cursor = conn.cursor()

        # Call the stored function using SELECT
        cursor.execute("SELECT * FROM findAdmissionsByAdmin(%s)", (login,))
        result = cursor.fetchall()  # Fetch all matching rows

        # Get column names from cursor.description
        column_names = [desc[0] for desc in cursor.description]

        # Convert each row in the result to a dictionary
        result_dicts = [dict(zip(column_names, row)) for row in result]

        return result_dicts if result_dicts else None

    except psycopg2.Error as sqle:
        print("psycopg2.Error : " + sqle.pgerror)

    finally:
        if conn:
            cursor.close()
            conn.close()


'''
Find a list of admissions based on the searchString provided as parameter
See assignment description for search specification
'''
def findAdmissionsByCriteria(searchString):
    conn = None
    try:
        conn = openConnection()
        cursor = conn.cursor()

        # Call the function using SELECT
        cursor.execute("SELECT * FROM findAdmissionsByCriteria(%s)",
(searchString,))
        result = cursor.fetchall()  # Fetch all matching rows

        # Get column names from cursor.description
        column_names = [desc[0] for desc in cursor.description]

        # Convert each row in the result to a dictionary
        result_dicts = [dict(zip(column_names, row)) for row in result]

        return result_dicts if result_dicts else None

    except psycopg2.Error as sqle:
```

```
        print("psycopg2.Error : " + sqle.pgerror)

    finally:
        if conn:
            cursor.close()
            conn.close()


'''
Add a new addmission
'''
def addAdmission(type, department, patient, condition, admin):
    conn = None
    try:
        conn = openConnection()
        cursor = conn.cursor()

        # Call the stored function
        cursor.execute("SELECT addAdmission(%s, %s, %s, %s, %s)",
                       (type, department, patient, condition, admin))
        conn.commit()  # Commit the transaction

        return True

    except psycopg2.Error as sqle:
        print("psycopg2.Error : " + sqle.pgerror)

    finally:
        if conn:
            cursor.close()
            conn.close()


'''
Update an existing admission
'''
def updateAdmission(id, type, department, dischargeDate, fee, patient, condition):
    conn = None
    try:
        conn = openConnection()
        cursor = conn.cursor()

        # Call the updateAdmission function
        cursor.execute("SELECT updateAdmission(%s, %s, %s, %s, %s, %s, %s)",
                       (id, type, department, dischargeDate, fee, patient,
condition))
        conn.commit()  # Commit the transaction

        return True

    except psycopg2.Error as sqle:
        print("psycopg2.Error : " + sqle.pgerror)

    finally:
        if conn:
            cursor.close()
            conn.close()
```