import sqlite3

def connect\_db():

conn = sqlite3.connect("database.db")

cursor = conn.cursor()

cursor.execute('''

CREATE TABLE IF NOT EXISTS rooms (

id INTEGER PRIMARY KEY AUTOINCREMENT,

room\_no INTEGER UNIQUE NOT NULL,

capacity INTEGER NOT NULL,

occupied INTEGER DEFAULT 0

)''')

cursor.execute('''

CREATE TABLE IF NOT EXISTS students (

id INTEGER PRIMARY KEY AUTOINCREMENT,

name TEXT NOT NULL,

age INTEGER NOT NULL,

room\_no INTEGER,

FOREIGN KEY(room\_no) REFERENCES rooms(room\_no)

)''')

cursor.execute('''

CREATE TABLE IF NOT EXISTS payments (

id INTEGER PRIMARY KEY AUTOINCREMENT,

student\_id INTEGER,

amount REAL,

date TEXT,

FOREIGN KEY(student\_id) REFERENCES students(id)

)''')

conn.commit()

conn.close()

# Room Functions

def add\_room(room\_no, capacity):

try:

conn = sqlite3.connect("database.db")

cursor = conn.cursor()

cursor.execute("INSERT INTO rooms (room\_no, capacity) VALUES (?, ?)", (room\_no, capacity))

conn.commit()

except Exception as e:

print("Error adding room:", e)

finally:

conn.close()

def fetch\_rooms():

conn = sqlite3.connect("database.db")

cursor = conn.cursor()

cursor.execute("SELECT \* FROM rooms")

data = cursor.fetchall()

conn.close()

return data

# Student Functions

def add\_student(name, age, room\_no):

try:

conn = sqlite3.connect("database.db")

cursor = conn.cursor()

cursor.execute("INSERT INTO students (name, age, room\_no) VALUES (?, ?, ?)", (name, age, room\_no))

cursor.execute("UPDATE rooms SET occupied = occupied + 1 WHERE room\_no = ?", (room\_no,))

conn.commit()

except Exception as e:

print("Error adding student:", e)

finally:

conn.close()

def fetch\_students():

conn = sqlite3.connect("database.db")

cursor = conn.cursor()

cursor.execute("SELECT \* FROM students")

data = cursor.fetchall()

conn.close()

return data

def delete\_student(student\_id):

try:

conn = sqlite3.connect("database.db")

cursor = conn.cursor()

cursor.execute("SELECT room\_no FROM students WHERE id = ?", (student\_id,))

room = cursor.fetchone()

if room:

cursor.execute("UPDATE rooms SET occupied = occupied - 1 WHERE room\_no = ?", (room[0],))

cursor.execute("DELETE FROM students WHERE id = ?", (student\_id,))

conn.commit()

except Exception as e:

print("Error deleting student:", e)

finally:

conn.close()

def update\_student(student\_id, name, age, room\_no):

try:

conn = sqlite3.connect("database.db")

cursor = conn.cursor()

cursor.execute("UPDATE students SET name = ?, age = ?, room\_no = ? WHERE id = ?", (name, age, room\_no, student\_id))

conn.commit()

except Exception as e:

print("Error updating student:", e)

finally:

conn.close()

def search\_students(keyword):

conn = sqlite3.connect("database.db")

cursor = conn.cursor()

query = """

SELECT \* FROM students

WHERE

CAST(id AS TEXT) LIKE ? OR

name LIKE ? OR

CAST(age AS TEXT) LIKE ? OR

CAST(room\_no AS TEXT) LIKE ?

"""

keyword = f"%{keyword}%"

cursor.execute(query, (keyword, keyword, keyword, keyword))

results = cursor.fetchall()

conn.close()

return results

# Payment functions (optional, use if needed)

def add\_payment(student\_id, amount, date):

conn = sqlite3.connect("database.db")

cursor = conn.cursor()

cursor.execute("INSERT INTO payments (student\_id, amount, date) VALUES (?, ?, ?)", (student\_id, amount, date))

conn.commit()

conn.close()

def fetch\_payments():

conn = sqlite3.connect("database.db")

cursor = conn.cursor()

cursor.execute("SELECT payments.id, students.name, payments.amount, payments.date FROM payments JOIN students ON payments.student\_id = students.id")

data = cursor.fetchall()

conn.close()

return data