Python/Database Interaction

SQLite Example:

1. import SQLite3
2. Establish a connection to a new or existing database:

```
dbconn = sqlite3.connect('SQLite_Python.db')
```

   A database can be created in RAM instead of a physical file with:

```
dbconn = sqlite3.connect(':memory:')
```

3. Obtain a cursor object:

```
cursor = dbconn.cursor()
```

4. Execute a SQL statement with the execute method of the cursor object:

```
cursor.execute(‘SELECT * FROM users’)      #cursor.executemany is another option
```

   Alternatively, you can read SQL commands from a script (text file):

```
cursor.executescript(sql_script)
```

5. For SELECT queries, fetch and process record set:

```
rows = cursor.fetchall()    #other options are fetchone (1 row), fetchmany(numrows)
for row in rows:
    print('{0}, {1}, {2}, {3}'.format(row[0], row[1], row[2], row[3]))
```

6. Commit any changes (e.g., CREATE TABLE, INSERT, UPDATE, DELETE queries) to the database:

```
db.commit()
```

7. Very important: close the cursor and database connection:

```
cursor.close()
dbconn.close()
```

Best practice: Put all database operations in a try/except block. On error, rollback transactions to the last commit.

Examples of Connection Strings for Other Databases:

Database	**Import Module**	**Connection String**
T-SQL	`pymssql`	`dbconn = pymssql.connect(server=server, user=user, password=password, database=db)`
MySQL	`MySQLdb`	`dbconn = MySQLdb.connect(host=host, user=user, passwd=passwd, db=db)`
Postgres	`psycopg2`	`dbconn = psycopg2.connect(database=db, user=user, password=password, host=host, port="5432")`
ODBC databases such as MS SQL Server	`pyodbc`	`dbconn = pyodbc.connect("Driver={odbcdriver}; Server=servername; Database=dbname; Trusted_Connection=yes;”)`

Exercise:

Lookup the local forecast for your zipcode at the National Weather Service (forecast.weather.gov). Create a database (in your database type of choice). Add a table called, “Weather,” and define columns, “Id” (unique integer primary key), “Date (text)”, “Day” (text), “High” (integer), and “Low” (integer). Insert the number of days of temperature forecasts you have into the table in the appropriate fields. Select the data in the Weather table and process its records to print the records and calculate the average high and low temperatures for the days in the table. Optionally use Pandas for this purpose.