SQL Assignment
Data Mining and Discovery

Student Name: Saman Qayyum
Student ID: 23010599

Topic: Climate Database

1. Database Generation and Storage:

The UK Met Office website provides comprehensive data and research on climate and weather conditions. This climate data platform has some performance issues like how to manage large volumes of data, respecting user privacy, analyzing data effectively to extract actionable insights, handling real-time data efficiently, and managing the costs associated with data storage are all critical considerations. The reason for generating data from climate data platform, is that it often contains a wide variety of information useful for multiple scenarios, such as number of weather stations, and their recordings, temperature and other climate indicators information. This data is generated using python and stored into a SQLite database as shown below.

PythonApplication1.py

```
import pandas as pd
import numpy as np
from faker import Faker
import sqlite3
NUM_STATIONS = 50
NUM_OBSERVATIONS = 1000
NUM_CLIMATE_INDICATORS = 10
# Database setup
conn = sqlite3.connect('climate_data.db')
c = conn.cursor()
# **********************Create tables with SQL commands
c.execute('''
CREATE TABLE IF NOT EXISTS Stations (
    StationID INTEGER PRIMARY KEY AUTOINCREMENT,
    StationName TEXT,
    Latitude REAL,
    Longitude REAL,
    Elevation INTEGER
);
''')
c.execute('''
CREATE TABLE IF NOT EXISTS ClimateIndicators (
    IndicatorID INTEGER PRIMARY KEY AUTOINCREMENT,
    IndicatorName TEXT
);
''')
c.execute('''
```

```
CREATE TABLE IF NOT EXISTS Recordings (
    RecordingID INTEGER PRIMARY KEY AUTOINCREMENT,
    StationID INTEGER,
    ObservationID INTEGER,
    FOREIGN KEY (StationID) REFERENCES Stations(StationID),
    FOREIGN KEY (ObservationID) REFERENCES Observations(ObservationID)
);
''')
# ***************Insert synthetic data into the database
for _ in range(NUM_STATIONS):
    c.execute('INSERT INTO Stations (StationName, Latitude, Longitude, Elevation) VALUES (?, ?, ?, ?)',
              (fake.city(), float(fake.latitude()), float(fake.longitude()), fake.random_int(min=0, max=5000)))

indicator_names = ['Temperature', 'Humidity', 'Precipitation', 'Wind Speed', 'Air Pressure', 'Solar Radiation', 'Snowfall', 'Visibility', 'Frost Days',
for name in indicator_names:
    c.execute('INSERT INTO ClimateIndicators (IndicatorName) VALUES (?)', (name,))

for _ in range(NUM_OBSERVATIONS):
    c.execute('INSERT INTO Observations (IndicatorID, Value, ObservationDate) VALUES (?, ?, ?)',
              (fake.random_int(min=1, max=NUM_CLIMATE_INDICATORS), fake.random_number(), fake.date_between(start_date='-5y', end_date='today')))

for _ in range(NUM_OBSERVATIONS):
    c.execute('INSERT INTO Recordings (StationID, ObservationID) VALUES (?, ?)',
              (fake.random_int(min=1, max=NUM_STATIONS), _ + 1))
conn.commit()
conn.close()
'climate_data.db'
```

2. Database Schema:

- **Stations Table**: This table would store information about weather stations, including each station's ID, name, geographic coordinates (latitude and longitude), and elevation.
- **Climate Indicators Table**: This table would list different types of climate measurements, such as temperature, humidity, precipitation.
- **Observations Table**: This table store the climate observations.
- **Recordings Table**: This table would associate each observation with a particular station.
- **Nominal Attribute:** StationName
- **Ordinal Attribute:** Temperature
- **Interval Attribute:** ObservationDate
- **Ratio Attribute:** Elevation

We have created a complete database schemas based on SQL Queries as follows:

```
create table Stations
(
   StationID INTEGER PRIMARY KEY,
   StationName TEXT,
   Latitude REAL,
   Longitude REAL,
   Elevation INTEGER
);

create table ClimateIndicators
(
   IndicatorID INTEGER PRIMARY KEY,
   IndicatorName TEXT
);

create table if not exists Observations
(
   ObservationID INTEGER PRIMARY KEY AUTOINCREMENT,
   IndicatorID INTEGER,
   Value REAL,
   ObservationDate DATE,
   FOREIGN KEY (IndicatorID) REFERENCES ClimateIndicators(IndicatorID)
```

```
);

create table if not exists Recordings
(
    RecordingID INTEGER PRIMARY KEY,
    StationID INTEGER,
    ObservationID INTEGER,
    FOREIGN KEY (StationID) REFERENCES Stations(StationID),
    FOREIGN KEY (ObservationID) REFERENCES Observations(ObservationID)
);
```

3. Justification and Ethical Discussion about Climate Database:

The ethical justification for using a climate database schema involves promoting transparency, supporting informed decision-making, and enhancing public understanding of climate trends and impacts. It ensures data integrity and facilitates accurate analysis. Ethical use requires respecting data privacy, ensuring accuracy, and using data responsibly to avoid misinformation.

- **Transparency**: Making data accessible and understandable to climate database, supporting public and scientific scrutiny, and encouraging trust.
- **Accuracy**: Ensuring data integrity for reliable analysis and decision-making using climate data.
- **Privacy**: Respecting the confidentiality of sensitive information and communities.
- **Responsibility**: Using data to inform and educate without causing misinformation.
- **Collaboration**: Facilitating shared research and global responses to climate challenges.

4. SQL Queries:

4.1 List all weather stations and their locations:

Query	**SELECT StationID, StationName, Latitude, Longitude, Elevation FROM Stations;**
Description	This query retrieves all details from the Stations table, including each station's name and geographical location.
Output	

StationID	StationName	Latitude	Longitude	Elevation
1	Dixonburgh	-9.6422075	-122.68654	3043
2	West Benjamin	-32.0476195	-12.332847	4239
3	West Jasonton	26.95506	-124.228693	3320
4	East Elizabethchester	-82.2700655	-30.031661	2083
5	Port Luismouth	-76.9510095	114.637038	4563
6	Lake Katherineshire	9.069548	-98.452831	1889
7	Lake Jenniferport	-61.7218855	-140.072853	2596
8	South Edward	-1.67726	-129.460476	4630
9	Elizabethtown	84.904402	-139.403681	2985
10	New Josefort	52.173253	-116.215128	3811

4.2 Find the total number of observations recorded by each station:

Query	SELECT Stations.StationName, COUNT(Recordings.StationID) AS TotalObservations FROM Stations JOIN Recordings ON Stations.StationID = Recordings.StationID GROUP BY Stations.StationName;
Description	This query shows how many weather observations each station has recorded.
Output	

StationName	TotalObservations
Alextown	23
Angelachester	22
Brandonhaven	15
Charlesmouth	17
Colliershire	12
Dixonburgh	15
East Craig	23
East Elizabethchester	14
East Sarahburv	20

4.3 Average temperature recorded at each station:

Query	SELECT Stations.StationName, AVG(Observations.Value) AS AverageTemperature FROM Observations JOIN Recordings ON Observations.ObservationID = Recordings.ObservationID JOIN Stations ON Recordings.StationID = Stations.StationID JOIN ClimateIndicators ON Observations.IndicatorID = ClimateIndicators.IndicatorID WHERE ClimateIndicators.IndicatorName = 'Temperature' GROUP BY Stations.StationName;
Description	Calculates the average temperature for each weather station
Output	

StationName	AverageTemperature
Alextown	187569759.25
Angelachester	3157075
Brandonhaven	11403
Charlesmouth	391279.5
Colliershire	642206
Dixonburgh	35841003.5
East Craig	59990815
East Elizabethchester	2
East Sarahburv	132977422.75

4.4 Count the number of days with precipitation for each station:

Query	SELECT Stations.StationName, COUNT(Observations.ObservationID) AS RainyDays FROM Observations JOIN Recordings ON Observations.ObservationID = Recordings.ObservationID JOIN Stations ON Recordings.StationID = Stations.StationID JOIN ClimateIndicators ON Observations.IndicatorID = ClimateIndicators.IndicatorID WHERE ClimateIndicators.IndicatorName = 'Precipitation' AND Observations.Value > 0 GROUP BY Stations.StationName;
Description	Counts how many days each station recorded any amount of precipitation.
Output	

StationName	RainyDays
Alextown	2
Angelachester	4
Brandonhaven	1
Charlesmouth	4
Colliershire	2
Dixonburgh	2
East Craig	2
East Elizabethchester	1
East Sarahbury	4

4.5 Identify the highest temperature ever recorded at each station:

Query	SELECT Stations.StationName, MAX(Observations.Value) AS MaxTemperature FROM Observations JOIN Recordings ON Observations.ObservationID = Recordings.ObservationID JOIN Stations ON Recordings.StationID = Stations.StationID JOIN ClimateIndicators ON Observations.IndicatorID = ClimateIndicators.IndicatorID WHERE ClimateIndicators.IndicatorName = 'Temperature' GROUP BY Stations.StationName;
Description	Finds the highest temperature recorded at each station
Output	

StationName	MaxTemperature
Alextown	741170656
Angelachester	6314142
Brandonhaven	22780
Charlesmouth	782102
Colliershire	642206
Dixonburgh	71681912
East Craig	119981622
East Elizabethchester	2
East Sarahbury	531008174

4.6 Retrieve all data for a specific month and year across all stations:

Query	SELECT * FROM Observations JOIN Recordings ON Observations.ObservationID = Recordings.ObservationID JOIN Stations ON Recordings.StationID = Stations.StationID WHERE strftime('%m-%Y', Observations.ObservationDate) = '06-2021'; -- Example for June 2021
Description	Fetches all observations for June 2021 from all stations
Output	

Obs...	Indicat...	Value	Observatio...	Record...	StationID	Observa...	StationID	Station...	Latitude	Longit...	Elevation
16	7	200711...	2021-06-28	16	16	16	16	East Sar...	-59.86...	-160.6...	1595
36	8	79125	2021-06-08	36	12	36	12	Port Bra...	-10.98...	-155.1...	3543
494	10	1166	2021-06-11	494	37	494	37	Lake Pe...	-14.63...	107.27...	867
614	6	14104	2021-06-19	614	7	614	7	Lake Je...	-61.72...	-140.0...	2596
730	8	1847	2021-06-06	730	39	730	39	Zacharyl...	71.988...	-20.31...	235
861	10	62	2021-06-12	861	30	861	30	Hansons...	34.335...	-113.9...	3554
875	8	487815...	2021-06-29	875	18	875	18	South C...	15.733...	-122.2...	3040
890	1	3753	2021-06-07	890	42	890	42	West Ro...	-29.00...	-83.42...	199
905	1	534066...	2021-06-19	905	6	905	6	Lake Kat...	9.0695...	-98.45...	1889

4.7 Selection:

Query	SELECT * FROM Recordings
Description	Retrieve all columns and rows from Recordings Table
Output	

RecordingID	StationID	ObservationID
1	32	1
2	3	2
3	23	3
4	21	4
5	46	5
6	43	6
7	21	7
8	49	8
9	38	9
10	9	10

4.8 Display Table Structure:

Query	PRAGMA table_info(DataRecords)
Description	Give the table structure
Output	

cid	name	type	notnull	dflt_value	pk
0	DataRecordID	INTEGER	0	NULL	1
1	LocationID	INTEGER	0	NULL	0
2	MeasurementID	INTEGER	0	NULL	0