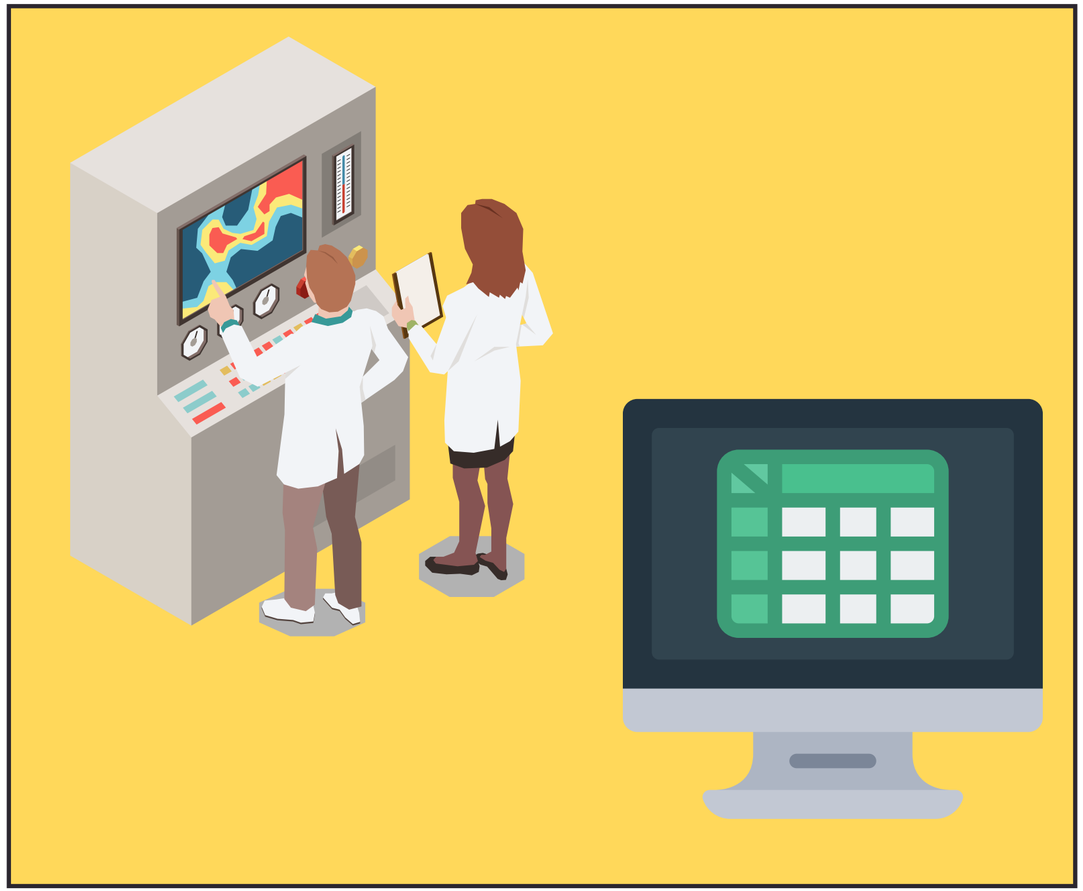


SOEN 363: Project Phase 2

Team 18:

Marita Brichan - 40138194

Sami Merhi - 40136648

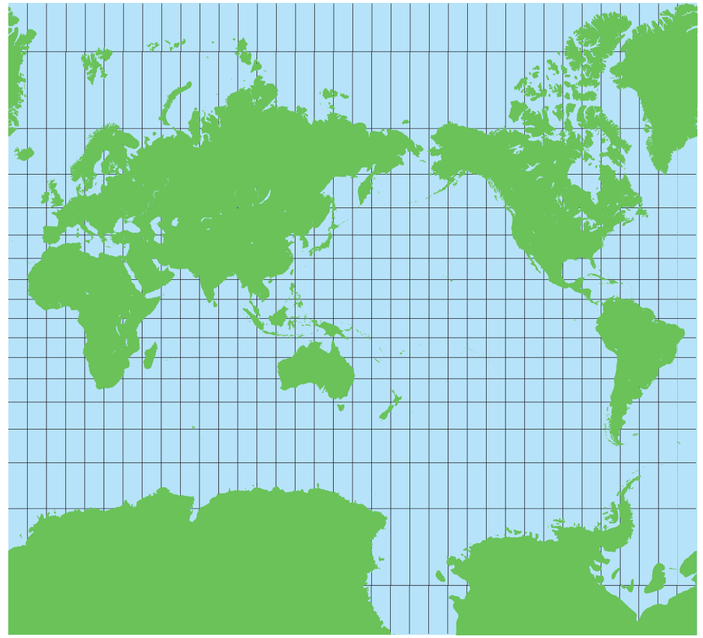
Dionisia Poulios - 40131986

Mohammad Ali Zahir - 40077619

Our dataset: Weather

Why?

Weather describes the world around us and guides our day-to-day, our future and our most important moments



Combination of 2 datasets

Global Historical Climatology Network (GHCN)

Public database giving data on maximum and minimum temperatures, precipitation, snow fall, wind strength, etc. which are organized by date and weather station

World Cities Database

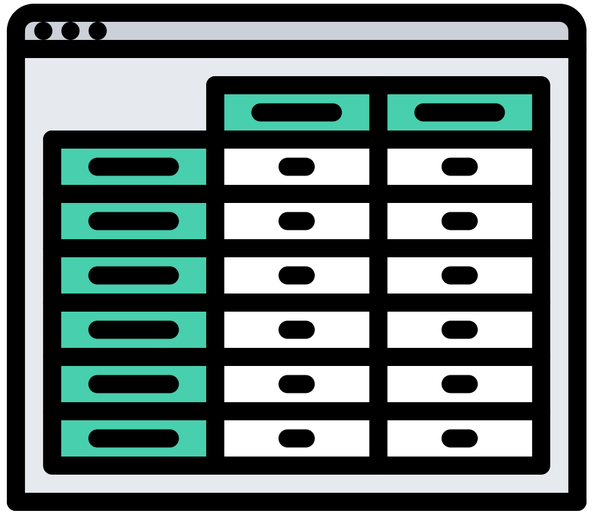
Database containing city names, geographical coordinates, population, country names, etc.

Dataset Information

Size: 1.24 GB

Format: .csv

Number of files: 9 files



File descriptions

1. 6 files for weather data from 2015-2020
2. 1 file with weather station ID and their coordinates
3. 1 file with city information and stationID
4. 1 file with information about each weather station

Formatting and Pre-processing

1. Combined and filtered, and split into 2 categories
2. Discarded
3. Unchanged
4. Discarded

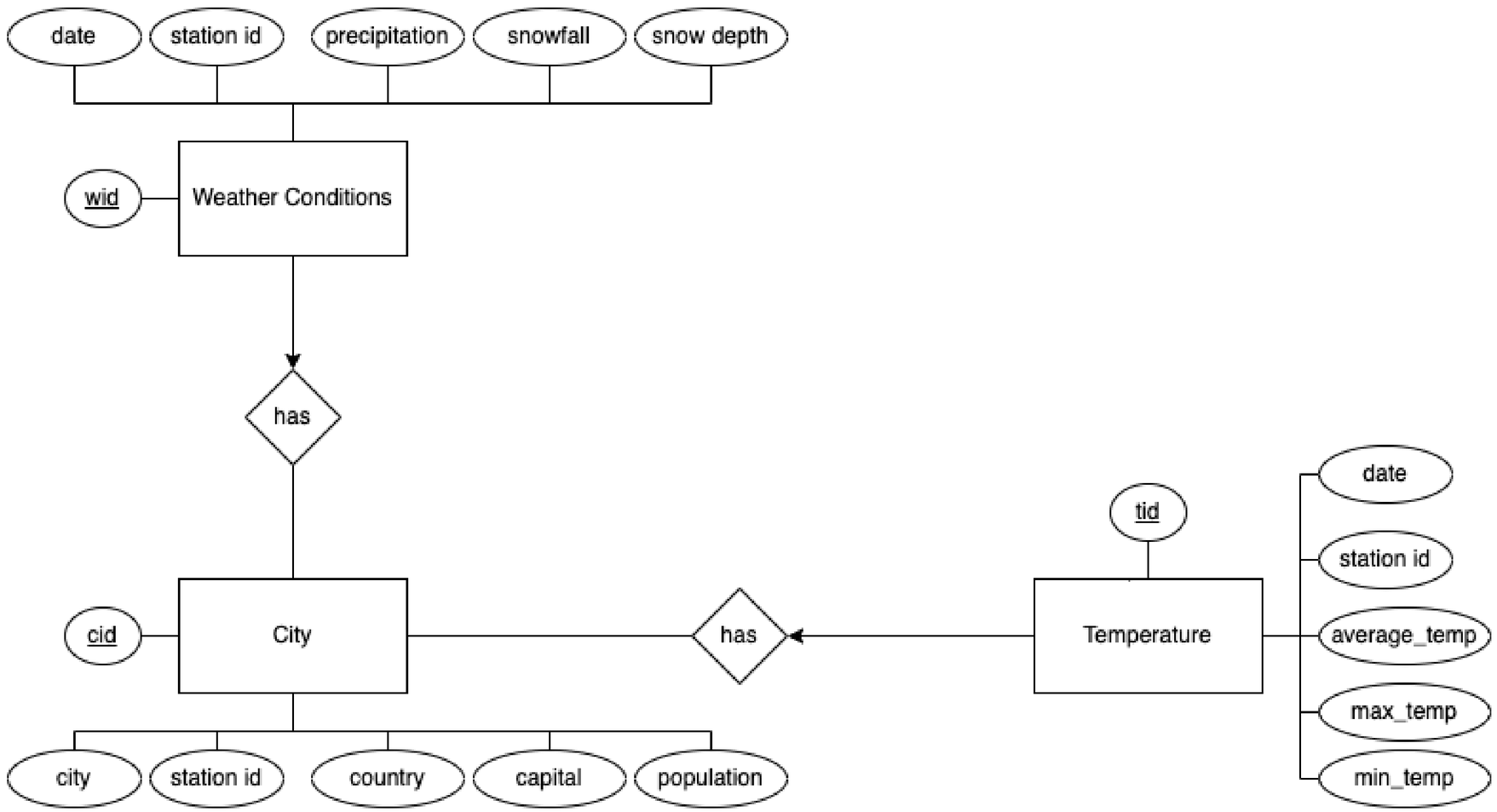
Creation of 2 new files
Number of files: 5 files
Total size: 1.66 GB

SQL

Data System:

PostegreSQL

ER Diagram (SQL)



Loading dataset in database (SQL-PostgreSQL)

```
create table Cities
(
cid INTEGER,
city CHAR(50),
country CHAR(50),
capital CHAR(8),
population DECIMAL,
stationID CHAR(20),
PRIMARY KEY (cid)
);
```

```
create table WeatherConditions
(
wid INTEGER,
stationID CHAR(20),
date DATE,
precipitation DECIMAL,
snowfall DECIMAL,
snow_depth DECIMAL,
PRIMARY KEY (wid)
);
```

```
create table Temperature
(
tid INTEGER,
stationID CHAR(20),
date DATE,
average_temp DECIMAL,
max_temp DECIMAL,
min_temp DECIMAL,
PRIMARY KEY (tid)
);
```

```
create table City_Weather
(
cid INTEGER,
wid INTEGER,
FOREIGN KEY (cid) references Cities,
FOREIGN KEY (wid) references
WeatherConditions,
PRIMARY KEY (cid, wid)
);
```

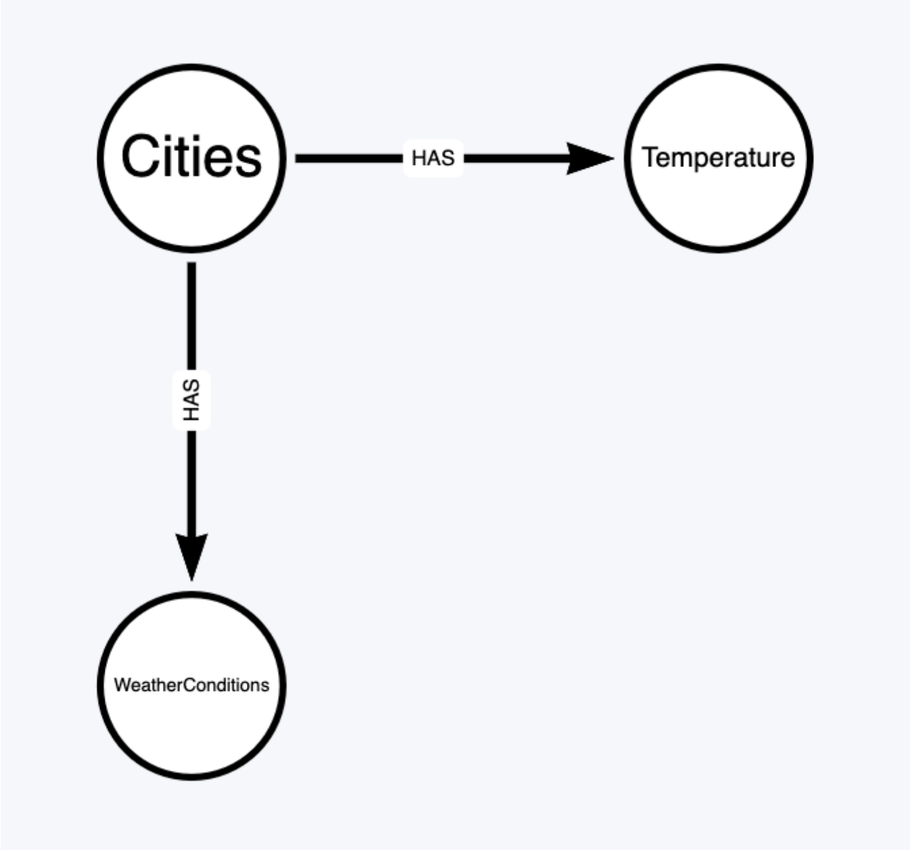
```
create table City_Temperature
(
cid INTEGER,
tid INTEGER,
FOREIGN KEY (cid) references Cities,
FOREIGN KEY (tid) references
Temperature,
PRIMARY KEY (cid, tid)
);
```

NoSQL

Data System:

Neo4j with Cypher

Data Model (NoSQL)



Loading dataset in database (NoSQL)

```
LOAD CSV WITH HEADERS
FROM "file:///all_cities.csv"
as cities
create (c1:City {cid: cities.cid,
city: cities.city, country:
cities.country, capital:
cities.capital, population:
cities.population, stationID:
cities.stationID})
```

```
:auto USING PERIODIC COMMIT 500 LOAD
CSV WITH HEADERS FROM
"file:///temperature.csv" as temperature
create (t1:Temperature {tid:
temperature.tid, stationID:
temperature.stationID, date:
temperature.Date, average_temp:
temperature.average_temp, max_temp:
temperature.max_temp, min_temp:
temperature.min_temp})
```

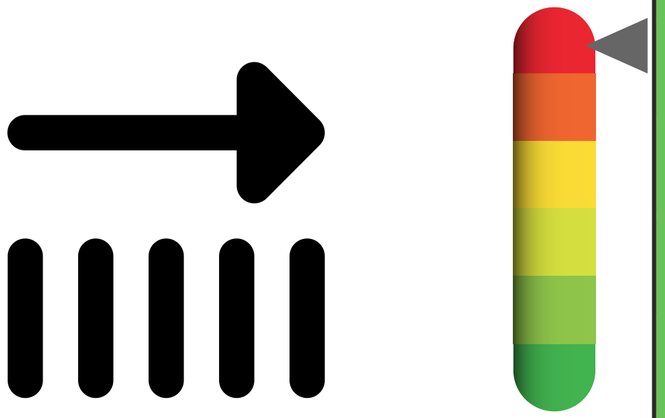
```
:auto USING PERIODIC COMMIT 500
LOAD CSV WITH HEADERS FROM
"file:///weather_conditions.csv" as
weather
create (w1:Weather {wid:
weather.wid, stationID:
weather.stationID, date:
weather.Date, precipitation:
weather.precipitation, snowfall:
weather.snowfall, snow_depth:
weather.snow_depth})
```

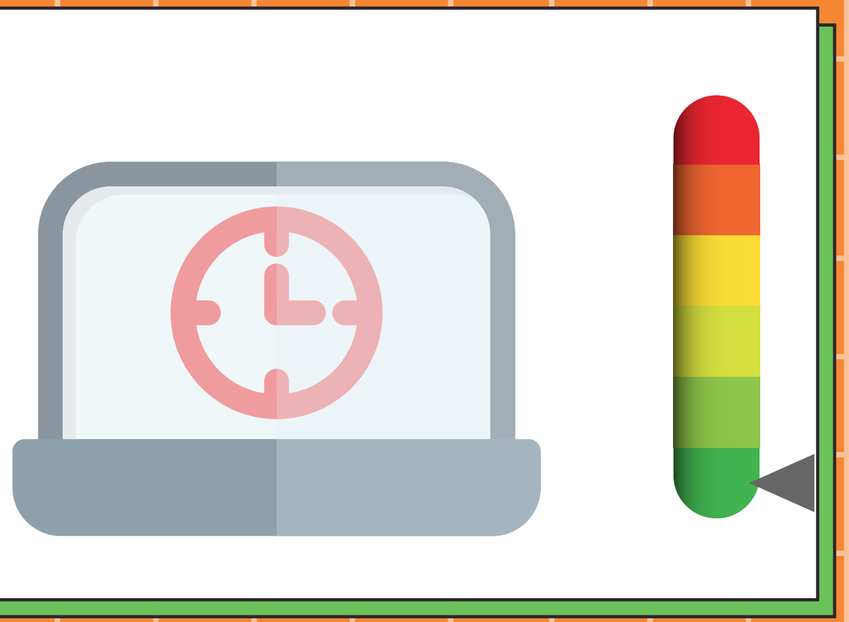
```
CALL apoc.periodic.iterate("
MATCH (c1:City)
MATCH (t1:Temperature)
WHERE c1.stationID=t1.stationID
RETURN c1, t1
", "
CREATE (c1)-[:HAS]->(t1)
", {batchSize: 10000, parallel:true}
)
```

```
CALL apoc.periodic.iterate("
MATCH (c1:City)
MATCH (w1:Weather)
WHERE c1.stationID=w1.stationID
RETURN c1, w1
", "
CREATE (c1)-[:HAS]->(w1)
", {batchSize: 10000, parallel:true}
)
```

Availability & Consistency

Neo4J is an ACID system, meaning that the system has high consistency while losing availability.





Availability & Consistency

For a system to have high availability, it will need to adhere to the CAP theorem.

However, the CAP theorem only applies to distributed databases while Neo4J **does not allow partitioning or sharding**, so it cannot be a distributed database.

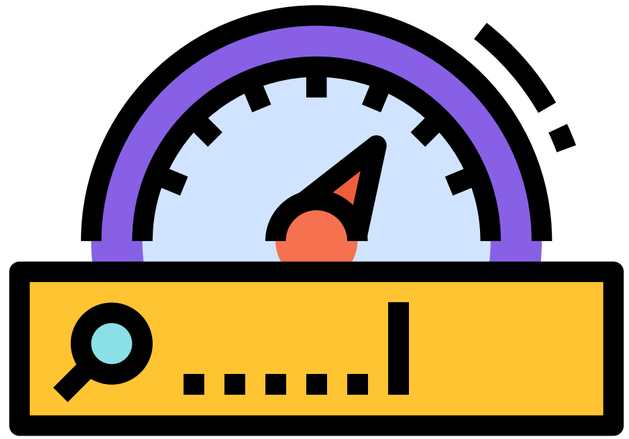
Indexes in Neo4j

Discussion

Indexing in Neo4J is done through nodes and relationships

There are 3 types of indexes that we could have in Neo4J

- TEXT Indexes - involve queries that have the keywords CONTAINS, STARTS WITH, AND ENDS WITH
- B-TREE Indexes - the default way to search for queries, usually for "normal" queries
- TOKEN Indexes - looking up relationships with specific labels

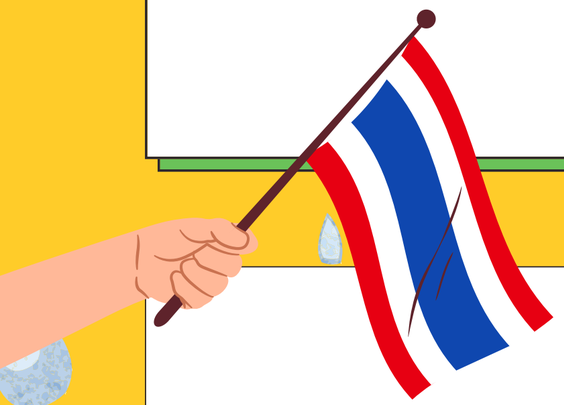


Indexes in Neo4j (continued)

In our case, we would only need to use the B-Tree Indexes since we do not have any specific queries :

```
CREATE INDEX city_index FOR (c:Cities) ON (c.cid)

CREATE INDEX weatherCondition_index FOR (w:WeatherConditions) ON (w.wid)

CREATE INDEX temperature_index FOR (t:Temperature) ON (t.tid)

CREATE INDEX cityWeather_index FOR (cw:City_Weather) ON (cw.cid, cw.wid)

CREATE INDEX cityTempeature_index FOR (ct:City_Temperature) ON (ct.cid, ct.tid)
```

I want to travel to Thailand but I want to avoid the rainy season. Show me the average precipitation in Thailand for each month of the year in ascending order.



SQL

execution time

without indexes:

3 secs 740 msec

with indexes:

4 secs 75 msec

NoSQL

execution time

without indexes:

4489 msec

with indexes:

1600 msec

I want to move to another city. I love the cold and I am a very anti-social, so I want to live in a city with less than 10, 000 people. Show me the top 10 cities along with their countries that have the lowest average minimum temperatures and their population.

SQL

execution time

without indexes:

6 secs 580 msec

with indexes:

1 secs 973 msec.

NoSQL

execution time

without indexes:

161197 ms

with indexes:

155365 ms

I want to get married in Montreal in July and want to avoid a rainy day. Which top days in July have the least precipitation on average. Show the day and its average precipitation throughout the years in ascending order.

SQL

execution time

without indexes:

901 msec

with indexes:

895 msec

NoSQL

execution time

without indexes:

413 ms

with indexes:

97 ms

I am a diplomat and have to travel to capital cities often. June is a particularly busy month for me. I enjoy temperatures between 22 and 25 degrees. Show me the capital cities with an average temperature between 22 and 25 degrees in June and their average temperature.

SQL

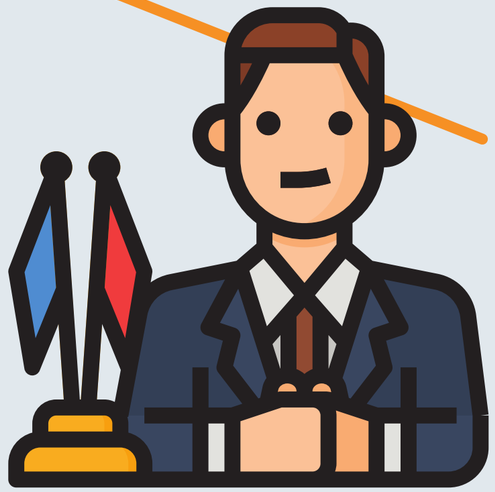
execution time

without indexes:

2 secs 278 msec

with indexes:

2 secs 197 msec



I want to relocate outside of Canada, it has become too cold for me. What are the top 10 hottest countries in terms of average temperature? Show the country and its average temperature in descending order.

NoSQL

execution time

without indexes:

71924 ms

with indexes:

70764 ms

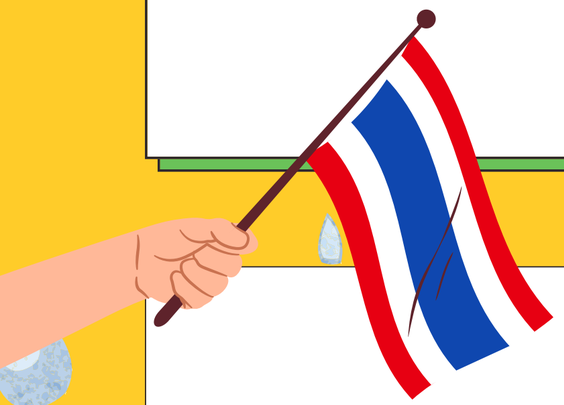
Queries-Live Demo

Data Systems:

PostegreSQL

Neo4j

I want to travel to Thailand but I want to avoid the rainy season. Show me the average precipitation in Thailand for each month of the year in ascending order.



SQL

```
SELECT EXTRACT(MONTH FROM W.date),
ROUND((AVG(W.precipitation)/10), 2) as
avg_precipitation_mm
FROM cities C, weatherconditions W
WHERE C.stationid=W.stationid and
C.country='Thailand' and W.precipitation IS
NOT NULL
GROUP BY EXTRACT(MONTH FROM W.date)
ORDER BY avg_precipitation_mm asc;
```

NoSQL

```
MATCH (c:City)-[h:HAS]->(w:Weather)
WHERE c.country='Thailand' and
w.precipitation IS NOT NULL
RETURN date(w.date).month,
ROUND(AVG(toInteger(w.precipitation))/10,
2) as avg_precipitation_mm
ORDER BY avg_precipitation_mm asc
```

I want to move to another city. I love the cold and I am a very anti-social, so I want to live in a city with less than 10, 000 people. Show me the top 10 cities along with their countries that have the lowest average minimum temperatures and their population.

SQL

```
SELECT C.city, C.country, C.population,
ROUND(AVG(T.min_temp/10), 2)
AS avg_min_temp
FROM Cities C, Temperature T
WHERE C.stationid = T.stationid AND C.population <
10000 AND T.min_temp IS NOT NULL
GROUP BY C.city, C.country, C.population
ORDER BY avg_min_temp asc
LIMIT 10;
```

NoSQL

```
MATCH (c:City)-[h:HAS]->(t:Temperature)
WHERE t.min_temp IS NOT NULL and c.population
IS NOT NULL and c.country IS NOT NULL and
(toInteger(c.population) < 10000) = true
RETURN c.city AS City, c.country AS
Country,c.population AS Population ,
ROUND(AVG(toInteger(t.min_temp))/10,2) as
avg_min_temp
ORDER BY avg_min_temp asc
LIMIT 10
```

I want to get married in Montreal in July and want to avoid a rainy day. Which top days in July have the least precipitation on average. Show the day and its average precipitation throughout the years in ascending order.

SQL

```
SELECT EXTRACT(MONTH FROM W.date) as month,
EXTRACT(DAY FROM W.date) as day ,
ROUND((AVG(W.precipitation)/10), 2) as avg_prec
FROM cities C, weatherconditions W
WHERE C.stationid=w.stationid and C.city='Montreal'
and w.precipitation IS NOT NULL and
EXTRACT(MONTH FROM W.date)=7
GROUP BY month, day
ORDER BY avg_prec asc, day;
```

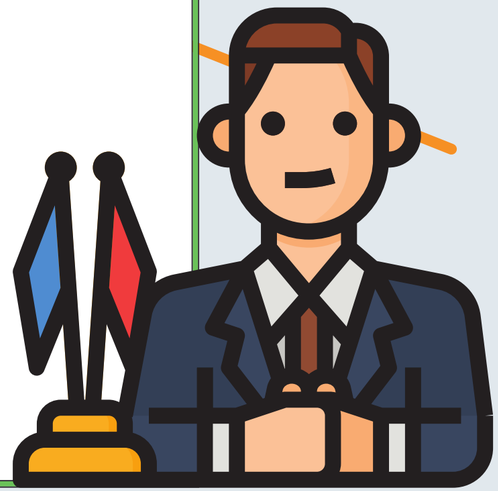
NoSQL

```
MATCH (c:City)-[h:HAS]->(w:Weather)
WHERE c.city='Montreal'and w.precipitation IS NOT
NULL and date(w.date).month=7
RETURN date(w.date).month, date(w.date).day,
ROUND(AVG(toInteger(w.precipitation))/10,2)
ORDER BY
ROUND(AVG(toInteger(w.precipitation))/10,2) asc,
date(w.date).day
```

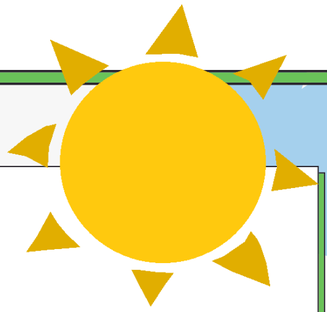
I am a diplomat and have to travel to capital cities often. June is a particularly busy month for me. I enjoy temperatures between 22 and 25 degrees. Show me the capital cities with an average temperature between 22 and 25 degrees in June and their average temperature.

SQL

```
SELECT C.city, ROUND((AVG(T.average_temp)/10), 2) as avg_temp
FROM Cities C, Temperature T
WHERE C.stationid=T.stationid and C.capital='primary' and
EXTRACT(MONTH FROM T.date)=6
GROUP BY C.city
HAVING ROUND((AVG(T.average_temp)/10), 2) >= 22 and
ROUND((AVG(T.average_temp)/10), 2) <=25;
```



I want to relocate outside of Canada, it has become too cold for me. What are the top 10 hottest countries in terms of average temperature? Show the country and its average temperature in descending order.



NoSQL

```
MATCH (c:City)-[h:HAS]->(t:Temperature)
WHERE t.average_temp IS NOT NULL
RETURN c.country AS Country,
ROUND(AVG(toInteger(t.average_temp))/10,2) as avg_temp
ORDER BY avg_temp desc
LIMIT 10
```

Queries-Results

Data Systems:

PostegreSQL

Neo4j

Results - SQL

Thailand Query

	extract	avg_precipitation_mm
1	2	4.2
2	3	7
3	4	7.62
4	7	8.11
5	1	8.27
6	5	8.79
7	6	8.84
8	8	8.89
9	11	9.53
10	12	10.12
11	10	10.31
12	9	11

Cold City Query

	city	country	population	avg_min_temp
1	Bukachacha	Russia	3372	-23.08
2	Batagay	Russia	4266	-17.97
3	Verkhoyansk	Russia	1122	-17.97
4	Susuman	Russia	4760	-16.74
5	Ust'-Nera	Russia	9148	-16.36
6	Menkerya	Russia	10	-15.96
7	Zhilinda	Russia	10	-15.75
8	Saskylakh	Russia	1920	-15.6
9	Pokrovsk	Russia	9256	-15.11
10	Khandyga	Russia	6796	-15.01

Results - SQL

Wedding Query

	month	day	avg_prec
1	7	2	0
2	7	4	0
3	7	5	0
4	7	10	0
5	7	16	0
6	7	22	0
7	7	23	0
8	7	24	0
9	7	25	0
10	7	26	0
11	7	29	0
12	7	3	0.5
13	7	1	0.55
14	7	15	0.75
15	7	19	0.75
16	7	6	1.15
17	7	30	1.25
18	7	11	1.4
19	7	31	1.4
20	7	14	1.5
21	7	21	1.65
22	7	7	2.05
23	7	9	2.4
24	7	13	2.8
25	7	8	3.7
26	7	28	3.7
27	7	20	4.05
28	7	18	5.2
29	7	27	8.25
30	7	17	9
31	7	12	21.95

Capital Query

	city	avg_temp
1	Amman	24.36
2	Belgrade	22.4
3	Bishkek	23.38
4	Brazzaville	24.24
5	Kathmandu	23.73
6	Luanda	23.6
7	Madrid	22.51
8	Monaco	22.37
9	Port-Vila	22.98
10	Pristina	22.85
11	Pyongyang	22.18
12	Rome	23.02
13	San Jose	22.92
14	San Marino	22.92
15	Seoul	23.37
16	Skopje	22.85
17	Suva	24.07
18	Tegucigalpa	23.15
19	Tokyo	22.35
20	Valletta	24.04
21	Zagreb	23.25

Results - NoSQL

Thailand Query

"date(w.date).month"	"avg_precipitation_mm"
2	4.21
3	7.02
4	7.6
7	8.08
1	8.21
5	8.77
6	8.8
8	8.87
11	9.51
12	9.92
10	10.28
9	10.95

Cold City Query

"City"	"Country"	"Population"	"avg_min_temp"
"Bukachacha"	"Russia"	"3372.0"	-23.08
"Batagay"	"Russia"	"4266.0"	-17.97
"Verkhoyansk"	"Russia"	"1122.0"	-17.97
"Susuman"	"Russia"	"4760.0"	-16.74
"Ust'-Nera"	"Russia"	"9148.0"	-16.36
"Menkerya"	"Russia"	"10.0"	-15.96
"Zhilinda"	"Russia"	"10.0"	-15.75
"Saskylakh"	"Russia"	"1920.0"	-15.6
"Pokrovsk"	"Russia"	"9256.0"	-15.11
"Khandyga"	"Russia"	"6796.0"	-15.01

Results - NoSQL

Wedding Query

"date(w.date).month"	"date(w.date).day"	"ROUND(AVG(toInteger(w.precipitation))/10,2)"
7	2	0.0
7	4	0.0
7	5	0.0
7	10	0.0
7	16	0.0
7	22	0.0
7	23	0.0
7	24	0.0
7	25	0.0
7	26	0.0
7	29	0.0
7	3	0.5
7	1	0.55
7	15	0.75
7	19	0.75
7	30	1.25
7	11	1.4
7	31	1.4
7	14	1.5
7	21	1.65
7	7	2.05
7	6	2.3
7	9	2.4
7	13	2.8
7	8	3.7
7	28	3.7
7	20	4.05
7	18	5.2
7	27	8.25
7	17	9.0
7	12	21.95

Hot Country Query

"Country"	"avg_temp"
"Chad"	30.57
"Niger"	29.64
"Sudan"	29.53
"Yemen"	29.27
"Somalia"	29.13
"United Arab Emirates"	29.13
"Maldives"	29.12
"Tuvalu"	29.11
"Qatar"	29.09
"South Sudan"	29.03

References

Data Systems:

PostegreSQL

Neo4j

References

Slides 12-13: https://www.infoq.com/articles/graph-nosql-neo4j/
Slides 14-15: https://neo4j.com/docs/cypher-manual/current/query-tuning/indexes/

Dataset used:
https://drive.google.com/drive/folders/1lettO8xSJz4LAsgOh9gU81lKtx9DgJKr?usp=sharing

https://www.kaggle.com/datasets/abhijitk/ghcn-daily-weather-data-by-city-20152020