CCT - College of Computing Technology

**Database Project - Group Alohomora**

**Part 1**

13th December, 2017

|  |  |
| --- | --- |
| **Group Members** | **% of Work Done** |
| Bruno Santos | 12.5/100% |
| Ennio Vitor | 12.5/100% |
| Felipe Mello | 12.5/100% |
| Fernanda Palomino | 12.5/100% |
| Luciene Rodrigues | 12.5/100% |
| Talita Brizolla | 12.5/100% |
| Thieres Pereira | 12.5/100% |
| Vagner Costa | 12.5/100% |

**1) Carry out analysis of the proposed LEFT JOIN – note that there are 8640 records in the TimestampTable and 4619 records in the WindSpeedTable. Use SQL queries to verify these figures.**

Analysing the table timestamp we could verify and confirm that the records given are matching with the records in the file:

SELECT

(SELECT COUNT(\*) FROM timestamp) AS 'Timestamp Table',

(SELECT (6\*24\*(31+29))) AS Expected;



Analysing the table Windspeed we could verify and confirm that the records given are matching with the records in the file:

SELECT

(SELECT COUNT(\*) FROM windspeed

WHERE sampleTime

LIKE '2008%') AS 'Windspeed Table',

(SELECT 4619) AS Expected;



**2) Explain, based on your understanding of the data – why there are 8640 records in the TimestampTable and why there are 4619 records in the WindSpeedTable (give a written explanation showing calculations).**

In the TimeStamp Table there are 8640 records because each turbine is supposed to collect data at every 10 minutes which means 6 records per hour, 6\*24 = 144. There are 31 days in January and 29 days in February (2008 was a leap year), that's 60 days in total. 60\*144 = 8640 records.

SELECT 6\*24\*(31+29) AS 'Records in Timestamp Table';



In the Windspeed Table there are 4621. Upon inspecting the data we discovered that 2 of those records referred to data was collected in the year 2009, but we're interested in the year 2008, so if we disconsider these 2 records from 2009, we have 4619 records.

SELECT (COUNT(\*)-2) AS 'Records in WindSpeed Table'

FROM windspeed;



**3) Write SQL to verify that the TimestampTable contains data for Jan/Feb 2008. Each Day (31 in Jan and 29 in Feb) stores 24 hours with 6 samples per hr (i.e. each at 10min intervals)**

SQL query to verify that TimeStamp contains data for January of 2008.

SELECT j.\*, (j.HoursPerDay\*j.samples\*j.Days) AS 'Total Jan Samples'

FROM

(SELECT 24 AS HoursPerDay, 6 AS Samples, 32 AS Turbines, 31 AS Days, 4.464 AS Expected) as j;



SQL query to verify that TimeStamp contains data for February of 2008.

SELECT f.\*, (f.HoursPerDay\*f.samples\*f.Days) AS 'Total Feb Samples'

FROM

(SELECT 24 AS HoursPerDay, 6 AS Samples, 32 AS Turbines, 29 AS Days, 4.176 AS Expected) as f;



**4) Based on an understanding of how the LEFT JOIN operation will work, and before executing any query, write a short calculation of the number of records there should be in the output of the left join. Compare this to what you actually get after you execute a query.**

We expected 4608 records, however we got an output of 13104 because while carrying a left join query we got duplicated values of timestamps and these values were marked as null: 8496 null (duplicated values) + 4608 (correct values) = 13104 in the output.

SELECT turbineID, timestamp, val

FROM timestamp

LEFT JOIN w

ON w.sampleTime = timestamp.timestamp

WHERE timestamp LIKE '%2008-01-01%' -- **This line gives us 4.608 records. If we removed this line we get 13.104 records.**

GROUP BY turbineID, timestamp;

**5) Write SQL to find out where exactly the problem is. Give a brief written explanation to accompany your SQL.**

This query showed us the data we were given also contains samples collected on March of 2008 and February of 2009.

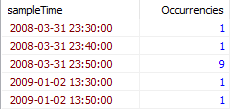
SELECT sampleTime,

COUNT(DISTINCT turbineID) AS Occurrencies

FROM windspeed

GROUP BY(sampleTime)

HAVING COUNT(\*) <> 32;



**6) Write SQL to solve the problem for Timestamp and Windspeed (i.e. a windspeed for each time-stamp) and give a brief description of how the overall problem of getting each variable (pitch, yaw, etc.) all lined up correctly (or “squarely” as it has been termed) would work.**

To have the tables timestamp and windspeed squared, we executed queries to fill each timestamp row with the correct windspeed value or a null value when the sample was not collected. The same procedure should be repeated in order to have all the other variables lined up correctly.

DROP TABLE IF EXISTS windspeed\_updated;

CREATE TABLE windspeed\_updated(

turbineID varchar(45) NULL,

sampleTime timestamp NULL,

val double NULL);

INSERT INTO windspeed\_updated

(SELECT \* FROM windspeed WHERE sampleTime LIKE '2008-01-01%');

The query below checks if there are any null record in the new table.

SELECT

turbineID, COUNT(\*)

FROM windspeed\_updated

GROUP BY turbineID

HAVING COUNT(\*) <> 144;

**7) Use the supplied alldata.sql to get each variable (pitch, yaw, etc) and create a fully joined (fully “square”) solution.**

DROP TABLE IF EXISTS windfarm;

CREATE TABLE windfarm (turbineID VARCHAR(10)

, sampleTime TIMESTAMP NULL

, p DOUBLE NULL

, r DOUBLE NULL

, w DOUBLE NULL

, y DOUBLE NULL

, pi DOUBLE NULL

);

INSERT INTO windfarm (turbineID, sampleTime, p, r, w,y,pi)

(SELECT turbineID, timestamp, p, r, w, y, pi

FROM inputdata

LEFT JOIN timestamp

ON timestamp.timestamp = inputdata.sampleTimeP

AND timestamp.timestamp = inputdata.sampleTimeR

AND timestamp.timestamp = inputdata.sampleTimeW

AND timestamp.timestamp = inputdata.sampleTimeY

AND timestamp.timestamp = inputdata.sampleTimePi

GROUP BY turbineID, timestamp);