SQL CODE

# Calculating First Indices

## Average Temperature, average daily and average night temperature

1. DROP VIEW IF EXISTS indices CASCADE;
2. CREATE VIEW indices AS
3. SELECT
4. mean\_total.id, mean\_total.device\_id, mean\_total.t\_avg,
5. mean\_d.t\_d,
6. mean\_n.t\_n
7. FROM
8. (
9. SELECT
10. id,
11. device\_id,
12. avg(d.value) as t\_avg
13. FROM data d
14. JOIN metadata m ON m.id=d.meta\_id
15. WHERE m.term\_id = 11
16. GROUP BY m.id
17. ) as mean\_total
19. JOIN
21. (
22. SELECT
23. id,
24. avg(d.value) as t\_d
25. FROM data d
26. JOIN metadata m ON m.id=d.meta\_id
27. WHERE d.variable\_id=1 AND m.term\_id = 11
28. AND EXTRACT(HOUR FROM d.tstamp) >= 6
29. AND EXTRACT(HOUR FROM d.tstamp) < 18
30. GROUP BY m.id
31. ) as mean\_d
33. ON mean\_total.id=mean\_d.id
35. JOIN
37. (
38. SELECT
39. id,
40. avg(d.value) as t\_n
41. FROM data d
42. JOIN metadata m ON m.id=d.meta\_id
43. WHERE d.variable\_id=1 AND m.term\_id = 11
44. AND (EXTRACT(HOUR FROM d.tstamp) < 6
45. OR EXTRACT(HOUR FROM d.tstamp) >= 18)
46. GROUP BY m.id
47. ) as mean\_n
49. ON mean\_d.id=mean\_n.id
51. SELECT \* FROM indices;

## Adding temperature difference of day and night

1. DROP VIEW IF EXISTS hobo\_indices CASCADE;
2. CREATE VIEW hobo\_indices AS
3. SELECT \*,
4. (i.t\_d - i.t\_n) as t\_nd
5. FROM indices i;
6. SELECT \* FROM hobo\_indices;

# Final Table of Indices

## Prequesitions

Determination of closest neighbour of each HOBO in 2021

1. DROP VIEW IF EXISTS meta21;
2. CREATE VIEW meta21 AS
3. -- From here, it's only a single SQL statement
4. WITH meta21 AS (
5. SELECT \*,
6. (SELECT id FROM metadata ly WHERE term\_id=9 ORDER BY st\_distance(m.location, ly.location) ASC LIMIT 1) as close\_meta20\_id,
7. (SELECT id FROM metadata ly WHERE term\_id=7 ORDER BY st\_distance(m.location, ly.location) ASC LIMIT 1) as close\_meta19\_id
8. FROM metadata m
9. WHERE term\_id=11 AND sensor\_id=1
10. )
11. -- This is your main SELECT. You can adapt and add joins etc.
12. SELECT \* FROM meta21;

Normalizing data

1. DROP VIEW IF EXISTS data\_norm;
2. CREATE VIEW data\_norm AS
3. -- From here, it's only a single SQL statement
4. SELECT
5. row\_number() OVER (PARTITION BY meta\_id, variable\_id ORDER BY tstamp ASC) as measurement\_index,
6. \*,
7. value - avg(value) OVER (PARTITION BY meta\_id, variable\_id) AS norm,
8. avg(value) OVER (PARTITION BY meta\_id, variable\_id) AS group\_avg
9. FROM data;

Calculating Correlation to HOBOs in 2019

1. DROP VIEW IF EXISTS indices19 CASCADE;
2. CREATE VIEW indices19 AS
3. SELECT
4. meta21.id,
5. avg(d.value) AS "mean",
6. corr(d.norm, d20.norm) AS "Tcorr1Y19"
7. FROM data\_norm d
8. JOIN meta21 on meta21.id = d.meta\_id
9. JOIN metadata m20 on meta21.close\_meta19\_id=m20.id
10. JOIN data\_norm d20 on m20.id=d20.meta\_id AND d.measurement\_index=d20.measurement\_index
11. GROUP BY meta21.id;
12. SELECT \* FROM indices19;

Calculating Correlation to HOBOs in 2020

1. DROP VIEW IF EXISTS indices20;
2. CREATE VIEW indices20 AS
3. SELECT
4. meta21.id,
5. avg(d.value) AS "mean",
6. corr(d.norm, d20.norm) AS "Tcorr1Y20"
7. FROM data\_norm d
8. JOIN meta21 on meta21.id = d.meta\_id
9. JOIN metadata m20 on meta21.close\_meta20\_id=m20.id
10. JOIN data\_norm d20 on m20.id=d20.meta\_id AND d.measurement\_index=d20.measurement\_index
11. GROUP BY meta21.id;
12. SELECT \* FROM indices20;

Joining Correlations together

1. DROP VIEW IF EXISTS indices\_rosien CASCADE;
2. CREATE VIEW indices\_rosien AS
3. SELECT i19."mean", i19."Tcorr1Y19", i20."Tcorr1Y20", m.device\_id, m.location FROM indices19 i19
4. JOIN metadata m on i19.id=m.id
5. JOIN indices20 i20 on i19.id=i20.id;
6. SELECT \* FROM indices\_rosien;

Joining all VIEWS together: Final VIEW of all indices

1. DROP VIEW IF EXISTS HOBO\_corr;
2. CREATE VIEW HOBO\_corr AS
3. SELECT
4. id, r.device\_id, t\_avg, t\_d, t\_n, t\_nd, r."Tcorr1Y19", r."Tcorr1Y20"
5. FROM hobo\_indices
6. JOIN indices\_rosien r ON r.device\_id=hobo\_indices.device\_id;
7. SELECT \* FROM HOBO\_corr;