Spatial Data

Use the `metadata` and `raw_data` table for this introduction. To follow along, you need to upload the HOBO metadata and the the data recorded by the HOBOs into the database. It is recommended to load at least one of the past years as well. The hydenv CLI can dramatically simplify this process:

```
python -m hydenv examples hobo --terms=[WT18,WT19]
```

Below, you will find the SQL queries used in the video, followed by a summary of the lessons learned.

SQL commands in the video

Spatial Data 01

2:22:

```sql
-- create a Geometry in Postgis
SELECT ST_GeomFromText('POINT (7.852618 47.995554)') as geom
```

2:46:

```sql
SELECT ST_GeomFromText('POINT (7.852618 47.995554)') as geom
UNION
SELECT ST_GeomFromText('POINT (7.853141 48.003268)') as geom
```

4:19

```sql
-- calculate the distance between two points
SELECT
  ST_Distance(
      ST_GeomFromText('POINT (7.852618 47.995554)'),
      ST_GeomFromText('POINT (7.853141 48.003268)')
  )
as distance
```

8:44:

```sql
-- Transform coordinates
SELECT
  ST_Distance(
    ST_Transform(
        ST_GeomFromEWKT('SRID=4326;POINT (7.852618 47.995554)'),
        25832
    ),
    ST_Transform(
        ST_SetSRID(ST_GeomFromText('POINT (7.853141 48.003268)'), 4326),
        25832
    )
```

```
  )
as distance
```

9:38:

```
-- Transform coordinates into insufficient CRS
SELECT
  ST_Distance(
      ST_Transform(
        ST_GeomFromEWKT('SRID=4326;POINT (7.852618 47.995554)'),
        3857
      ),
      ST_Transform(
        ST_SetSRID(ST_GeomFromText('POINT (7.853141 48.003268)'), 4326),
        3857
      )
  )
as distance
```

Spatial Data 02

3:04:

```
-- HOBO distance to Freiburger Muenster
SELECT
ST_Distance(
    ST_Transform(location, 25832),
    ST_Transform(
        ST_GeomFromEWKT('SRID=4326;POINT (7.852618 47.995554)'),
        25832
    )
)
FROM metadata
```

3:40:

```
SELECT
ST_Distance(
    ST_Transform(location, 25832),
    ST_Transform(
        ST_GeomFromEWKT('SRID=4326;POINT (7.852618 47.995554)'),
        25832
    )
) AS distance
FROM metadata
ORDER BY distance ASC
```

5:21:

```
-- combining information
SELECT
  device_id,
  ST_AsEWKT(location) as "WKT",
ST_Distance(
    ST_Transform(location, 25832),
    ST_Transform(
        ST_GeomFromEWKT('SRID=4326;POINT (7.852618 47.995554)'),
        25832
    )
) AS distance
FROM metadata
ORDER BY distance ASC
```

Spatial Data 03

4:05:

```
-- aggregate the HOBOs and calculate distances
SELECT
  device_id,
  avg(r.value) as temperature,
  ST_AsEWKT(location) as "WKT",
  ST_Distance(
      ST_Transform(location, 25832),
      ST_Transform(
        ST_GeomFromEWKT('SRID=4326;POINT (7.852618 47.995554)'),
        25832
      )
  ) AS distance
FROM raw_data r
JOIN metadata m ON m.id=r.meta_id
WHERE r.variable_id=1
GROUP BY m.device_id, "WKT", distance
```

Spatial Data 04

5:12

```
-- select from a sub-query - sub-query JOIN
SELECT
  sub1.meta_id, sub1.temperature,
  sub2.distance, sub2."WKT"
FROM
(
SELECT
  meta_id,
```

```
  avg(r.value) as temperature
FROM raw_data
WHERE variable_id=1
GROUP BY meta_id
) as sub1

JOIN

(
SELECT
  id,
  ST_AsEWKT(location) as "WKT",
  ST_Distance(
    ST_Transform(location, 25832),
    ST_Transform(
        ST_GeomFromEWKT('SRID=4326;POINT (7.852618 47.995554)'),
        25832
    )
  ) AS distance
FROM metadata
) as sub2
ON sub1.meta_id=sub2.id
```

Summary

- PostGIS is a full featured GIS system
- (Almost) anything you can do with QGis, can be done with PostGIS
- On large datasets, PostGIS is often the only option. In the cloud, RAM and disk space is cheap, available and can be scaled in seconds.
- Keep in mind that PostGIS won't do any on-the-fly reprojection, thus forcing you to program cleaner code
- PostGIS is often favorable over local solutions as you are forced to implement your analysis as code. This is a huge advantage in terms of reproducability and maintenance
- technically, PostGIS in PostgreSQL are just a set of functions and data types. Anything you learned about PostGIS functions can be applied to PostGIS. You can use them to create attributes, Order, filter and Join data.