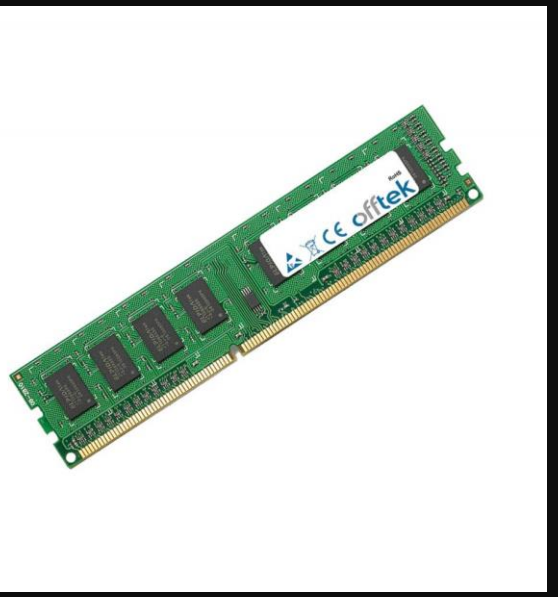
Databases

Jan Tyc, Bartłomiej Tarcholik

Technology used during project

- Ryzen 7 4800HS 35W
- 16 GB RAM 3200MHz
- Nvidia RTX2060 Max-Q



Tech stack

-Connecting to the database (via python-Arango), initial quering for relationships and some queries

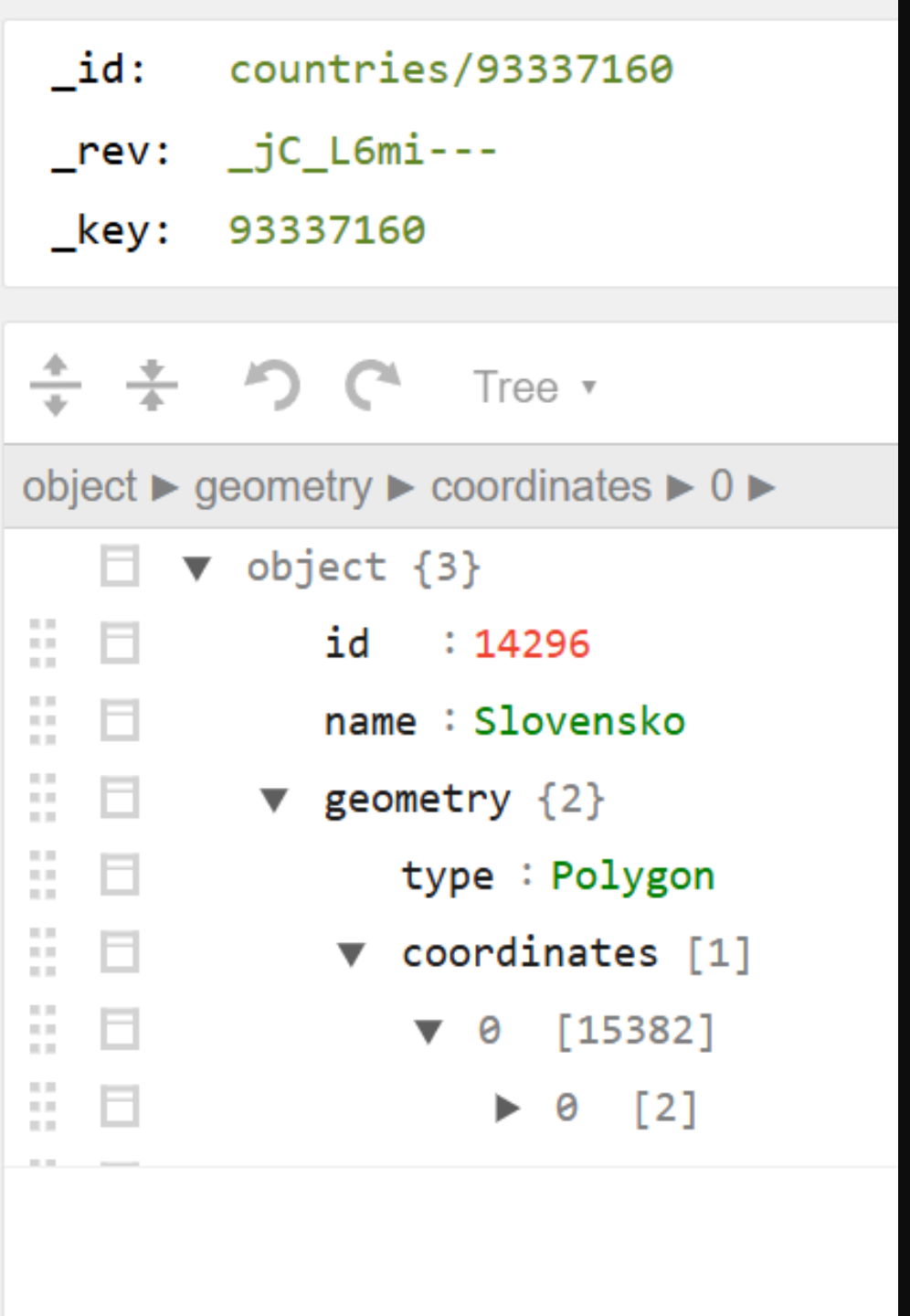
- Creating relationships, data import help

- More advanced, graph-based queries

Data model

- Data storage – Document Type
- (faster quering, we highly relied on python so it was most useful)
- Geometry as GeoJson
- Other columns as string or int

- Edges storage – Edge Type
- (ensures existence of indexes in collections, connects them with an custom attribute)



Data import

Have all the data been imported? Yes

What have we used? Python-Arango function „insert_many()”

Have we tested other things? Shell arangoimport function and Python-Arango „import_bulk()” but where were harder to implement or required prior conversion to json so it was more time consuming overall

Biggest issue: Converting geospatial data to geojson properly (good function with lots of ifs was needed)

Longest import time: Buildings ~ 40 minutes

Total import time: about 55 minutes

Relationships

What we used? Geopandas, cKDTree, and custom functions

Are all relationship detected? Except for relationship 6 is done but truncated to 100_000 buildings limit

Relationships times

- Relation 1: 112.21s
- Relation 2: 50.28s
- Relation 3: 20.49s
- Relation 4: 7.83s
- Relation 5: 23.41s
- Relation 6: 658.43s
- Relation 7: 813.46s
- Relation 8: 1878.11s
- Relation 9: 2203.87s
- Relation 10: 90.78s
- Total: ~ 1:20h

Relationship storage

Collection type: Edges

Schema:

- _from (id from one collection with collection name)(eg. „railways/8988738”
- _to (id from other collection with collection name)
- atribute: (distance, angle etc.)

Import time of 10,000 edges (from 0.25 to 1s depending on mode of processing) - tqdm used measuring

For trees edges (8 milion) it took 7 minutes.

Queries

Queries 1-3, 5 used purely AQL for quering

Query 6 is python based as necessery information are not in the relationships

Query 4,7 imports the data from relationships to python and creates a graph to more efficiently look for concave hulls and transform them

Query 10 works similarly to the 7th

Queries 8,9 are also lacking

Queries times

- Query 1 – 10s
- Query 2 – 23s
- Query 3 – 70s
- Query 4 – 34s
- Query 5 – 10s
- Query 6 – 17 minutes
- Query 7 - 90s
- Query 10 – 64s

Thank you.