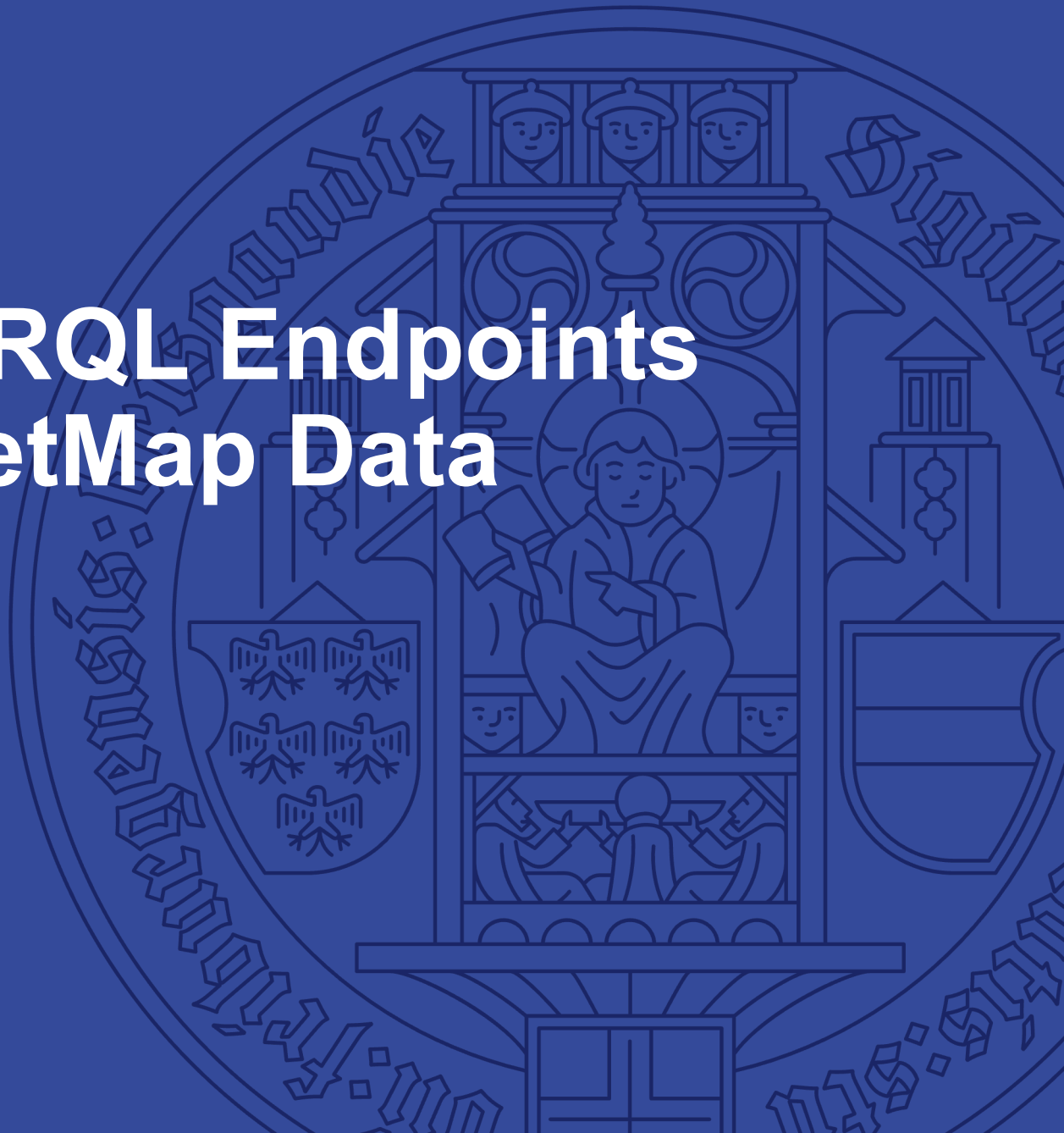
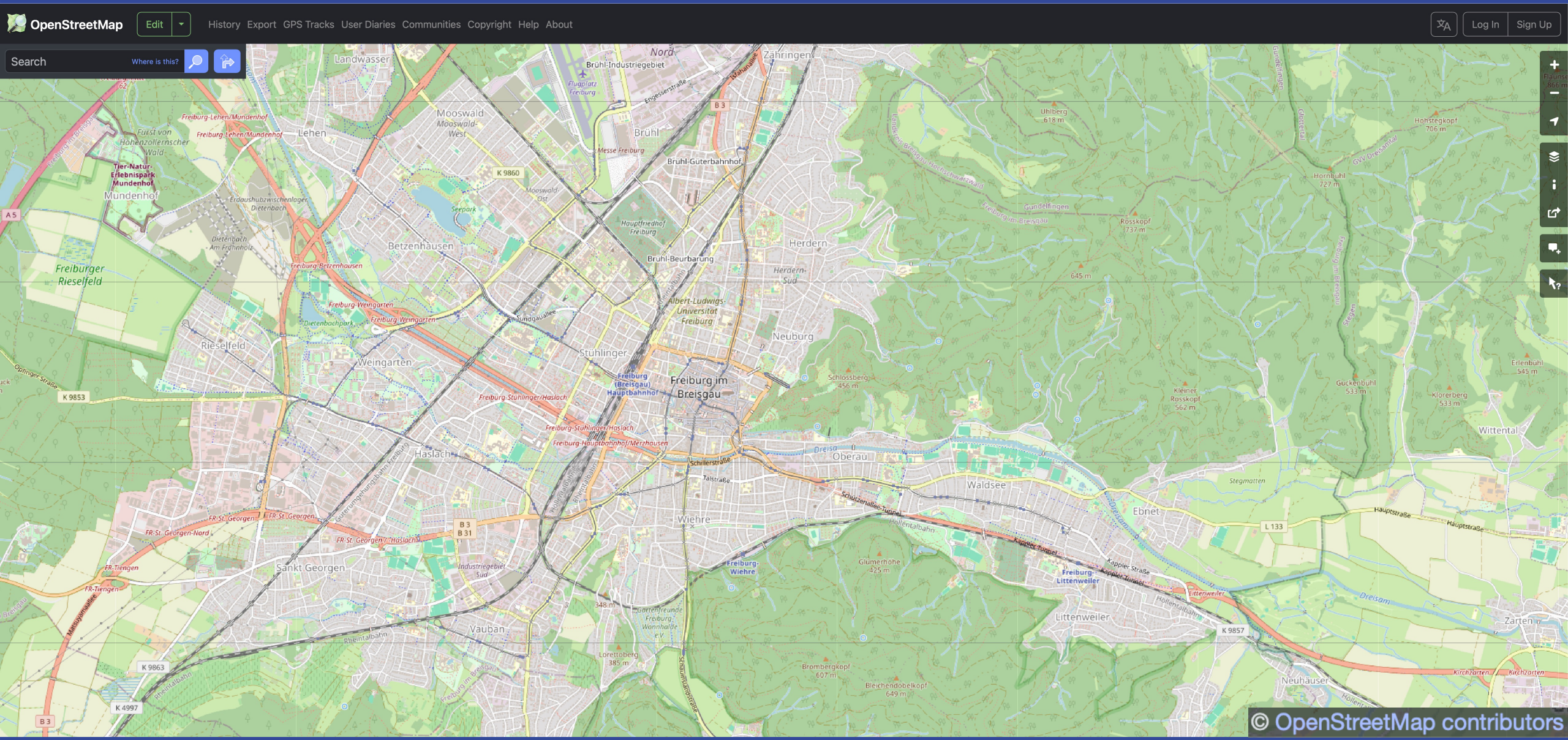
universität freiburg

Live Updates for SPARQL Endpoints Containing OpenStreetMap Data

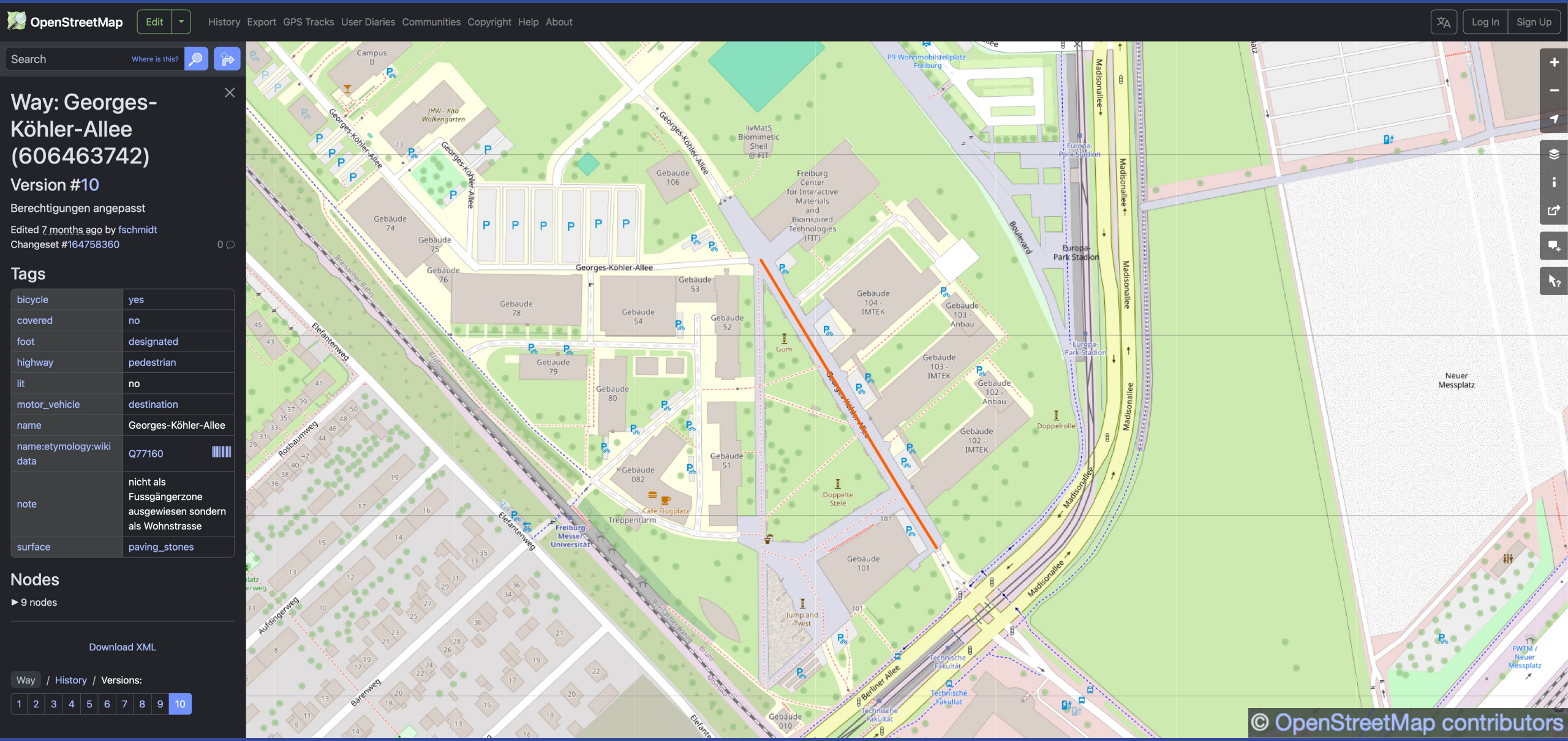


Nicolas von Trott
Chair of Algorithms and Data Structures
Faculty of Engineering
University of Freiburg

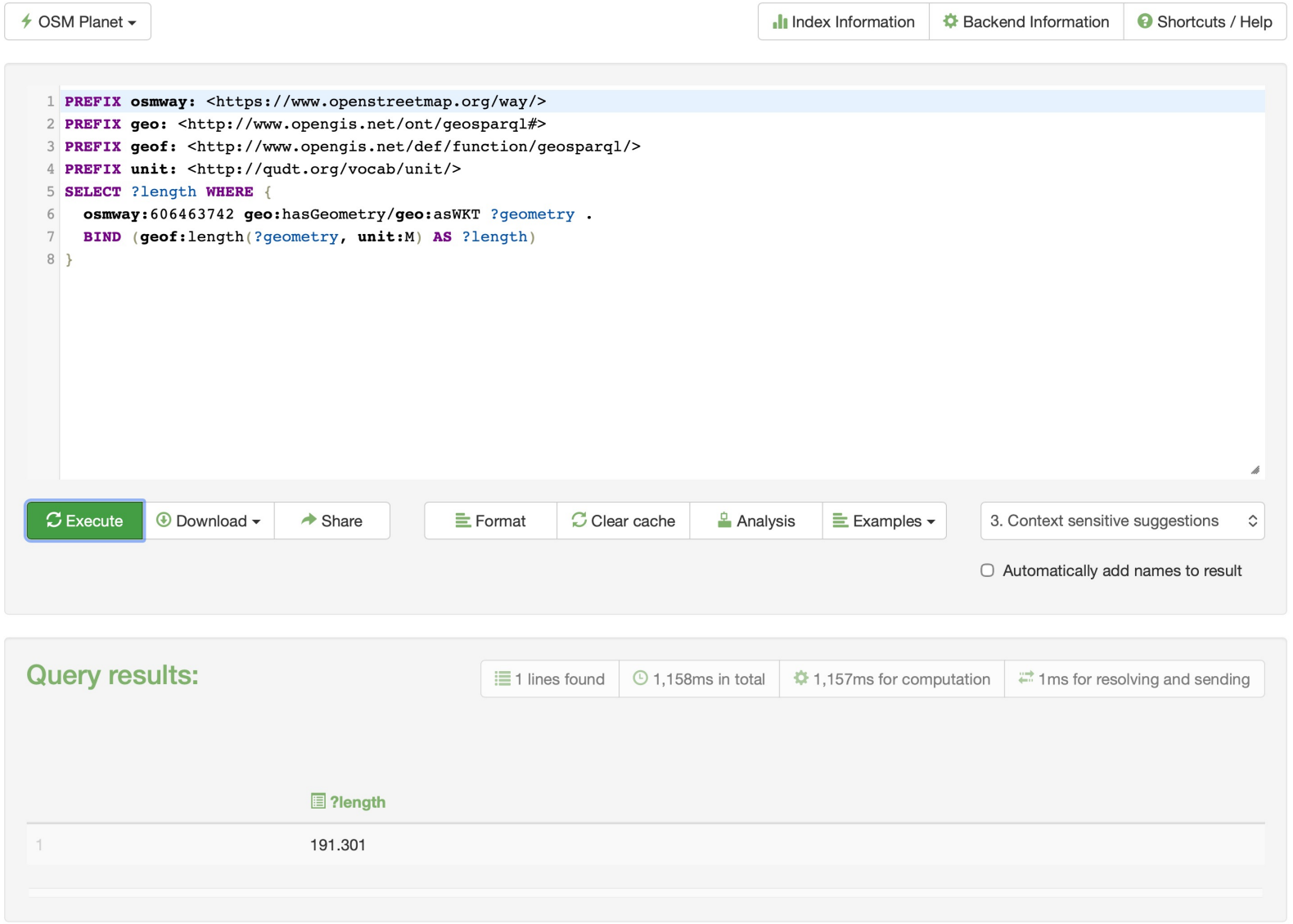
Introduction – OpenStreetMap



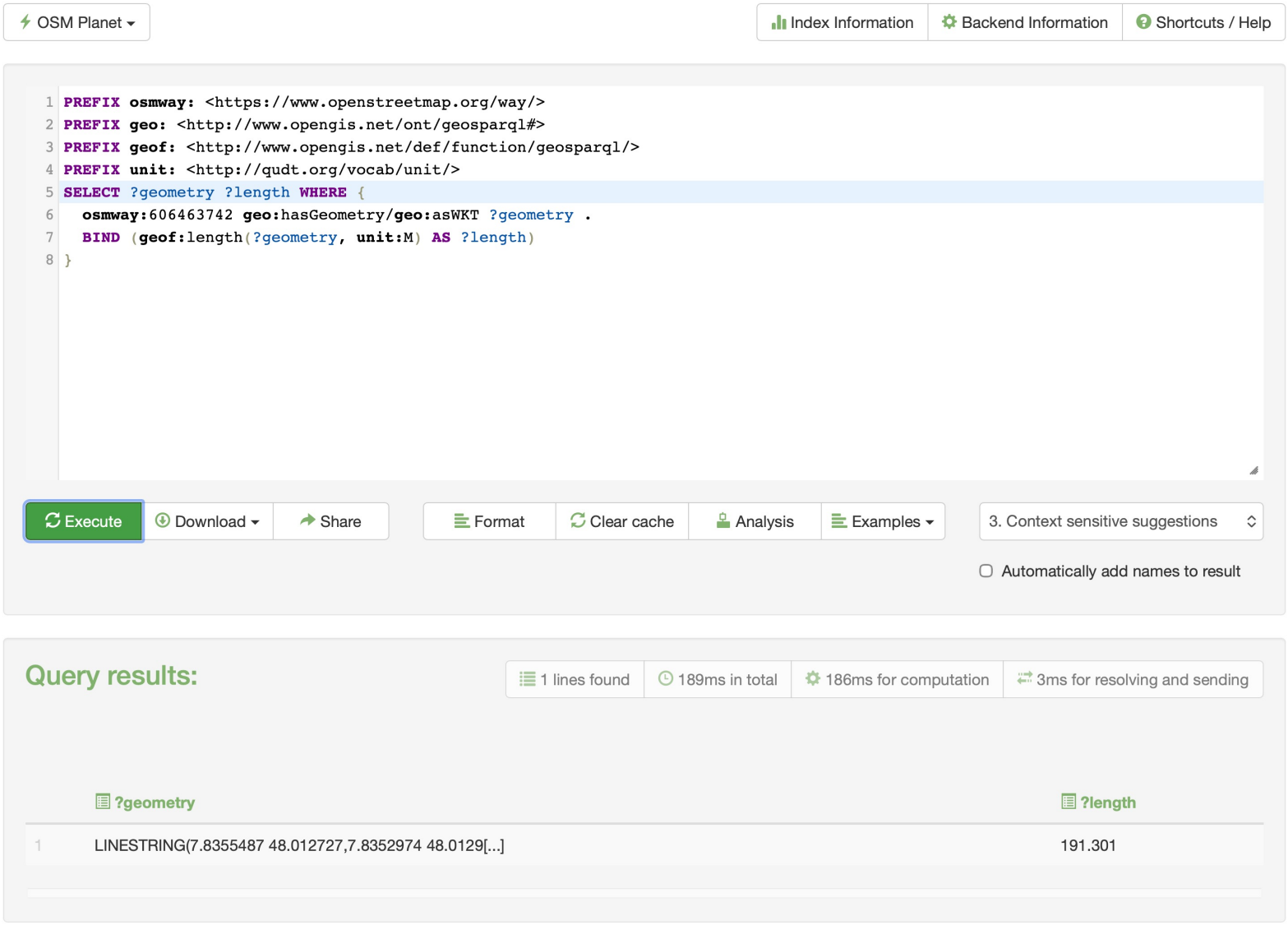
Introduction – OpenStreetMap



Introduction - QLever



Introduction - QLever



Introduction – Set up SPARQL endpoint

- Setting up a SPARQL endpoint can be time- and resource-intensive:
  - OSM Planet data consists of [1]:
    - ➢ 10 billion Nodes, 1 billion Ways, and 13 million Relations
  - Conversion with *osm2rdf* [2] takes around 17 hours
    - ➢ Results in 134.8 billion triples
  - Indexing with *QLever* [3] takes around 30 hours

➢ It takes 1-2 days to set up an OSM Planet instance

Introduction – Set up SPARQL endpoint

- Setting up a SPARQL endpoint can be time- and resource-intensive:
  - OSM Planet data consists of [1]:
    - ➢ 10 billion Nodes, 1 billion Ways, and 13 million Relations
  - Conversion with *osm2rdf* [2] takes around 17 hours
    - ➢ Results in 134.8 billion triples
  - Indexing with *QLever* [3] takes around 30 hours

  ➢ It takes 1-2 days to set up an OSM Planet instance

- OSM database is highly dynamic:
  - ➢ Around 3.2 million OSM elements are created, modified, or deleted each day in the OSM database

Introduction – Updating OSM data

- OSM change files provide a way to track changes to the OSM database

```
<osmChange version="0.6">
  <delete>
    <node id="17870963” lat="61.0513771" lon="29.0674509"/>
  </delete>
  <modify>
    <node id="31413949” lat="48.0141735" lon="7.8342196"/>
  </modify>
  <create>
    <way id="1347584463”>
      <nd ref="12465374441"/>
      <nd ref="12465374443"/>
      <tag k="highway" v="footway"/>
    </way>
  </create>
</osmChange>
```

Introduction – SPARQL Update Operations

- SPARQL update operations provide a way to modify triples

```
PREFIX osmmeta: <https://www.openstreetmap.org/meta/>
PREFIX osmnode: <https://www.openstreetmap.org/node/>

INSERT DATA {
  osmnode:12345 osmmeta:timestamp “2025-08-30T13:36:28”
}
```

```
PREFIX osmmeta: <https://www.openstreetmap.org/meta/>
PREFIX osmnode: <https://www.openstreetmap.org/node/>

DELETE DATA {
  osmnode:12345 osmmeta:timestamp “2025-08-30T13:36:28”
}
```

Introduction – Problem Statement

- **Problem:** Update a SPARQL endpoint containing OSM data

Introduction – Problem Statement

- **Problem:** Update a SPARQL endpoint containing OSM data
- **Solution:** Generate SPARQL update operations from OSM change files

Introduction – Problem Statement

- **Problem:** Update a SPARQL endpoint containing OSM data
- **Solution:** Generate SPARQL update operations from OSM change files
- Implemented ***osm-live-updates*** (***olu***), a tool that:
  - ➢ Downloads needed change files automatically
  - ➢ Generates SPARQL update operations from OSM change files
  - ➢ Updates SPARQL endpoint with generated update operations

Introduction - *olu* integration in *qlever-control*

- Set up SPARQL endpoint:

```
qlever get-data
qlever index
qlever start
```

- Start continuous updates in the desired interval:

```
qlever update-osm --granularity day/hour/minute
```

Implementation – Generate Update Operations

- 3 categories of OSM elements in the OSM change file:

  1. Deleted elements
  2. Created elements
  3. Modified elements

➢ Generate SPARQL update operations for these elements

Implementation - 1. Deleted OSM Elements

```
<delete>
  <node id="178709637” lat="61.0513771" lon="29.0674509"/>
</delete>
```

Implementation - 1. Deleted OSM Elements

```
<delete>
  <node id="178709637" lat="61.0513771" lon="29.0674509"/>
</delete>
```

- Delete all triples on the SPARQL endpoint for deleted OSM elements

```
PREFIX osmnode: <https://www.openstreetmap.org/node/>

DELETE WHERE {
  osmnode:178709637 ?predicate ?object .
  ?object ?hasGeometry ?geometry .
}
```

Implementation - 2. Created OSM Elements

```
<create>
  <way id="1347584463”>
    <nd ref="12465374441"/>
    <nd ref="12465374443"/>
    <tag k="highway" v="footway"/>
  </way>
</create>
```

Implementation - 2. Created OSM Elements

```
<create>
  <way id="1347584463”>
    <nd ref="12465374441"/>
    <nd ref="12465374443"/>
    <tag k="highway" v="footway"/>
  </way>
</create>
```

- Insert RDF triples into the SPARQL endpoint for newly created OSM elements
- Generate these triples using ***osm2rdf***

Implementation - 2. Created OSM Elements

```
<create>
  <way id="1347584463”>
    <nd ref="12465374441"/>
    <nd ref="12465374443"/>
    <tag k="highway" v="footway"/>
  </way>
</create>
```

- Insert RDF triples into the SPARQL endpoint for newly created OSM elements
- Generate these triples using ***osm2rdf***

➢ **Problem:** Member elements are not necessarily included in the change file

Implementation - 2. Created OSM Elements

- **Solution:** Fetch relevant information for member elements from the SPARQL endpoint:
  - ➢For nodes, fetch the location
  - ➢For ways and relations, fetch the ordered list of members

Implementation - 2. Created OSM Elements

- **Solution:** Fetch relevant information for member elements from SPARQL endpoint:
  - ➢For nodes, fetch the location
  - ➢For ways and relations, fetch the ordered list of members

```
<create>
  <node id="12465374441" lat="48.0141735" lon="7.8342196"/>
  <node id="12465374443" lat="48.0141764" lon="7.8342193"/>
  <way id="1347584463">
    <nd ref="12465374441"/>
    <nd ref="12465374443"/>
    <tag k="highway" v="footway"/>
  </way>
</create>
```

Implementation - 2. Created OSM Elements

- Insert **newly** generated triples into the SPARQL endpoint

```
INSERT DATA {
  osmway:238831394 rdf:type osm:way .
  osmway:238831394 osmway:member _:3_0 .
  _:3_0 osmway:member_id osmnode:296992601 .
  _:3_0 osmway:member_pos "0"^^xsd:integer .
  osmway:238831394 osmway:member _:3_1 .
  _:3_1 osmway:member_id osmnode:296992622 .
  _:3_1 osmway:member_pos "1"^^xsd:integer .
  osmway:238831394 geo:hasGeometry osm2rdfgeom:osmway_238831394 .
  osm2rdfgeom:osmway_238831394 geo:asWKT "LINESTRING(7.7322638 48.2683591,
      7.7323233 48.2685646)"^^geo:wktLiteral .
}
```

Implementation - 3. Modified Elements

```
<modify>
  <node id="31413949” lat="48.0141735" lon="7.8342196"/>
</modify>
```

Implementation - 3. Modified Elements

```
<modify>
  <node id="31413949” lat="48.0141735" lon="7.8342196"/>
</modify>
```

- **First:** Delete all triples on the SPARQL endpoint for the modified OSM element
- **Second:** Insert newly generated triples into the SPARQL endpoint

Implementation - 3. Modified Elements

```
<modify>
  <node id="31413949" lat="48.0141735" lon="7.8342196"/>
</modify>
```

- **First:** Delete all triples on the SPARQL endpoint for the modified OSM element
- **Second:** Insert newly generated triples into the SPARQL endpoint

➢ **Problem:** The modification of an OSM element can ***change*** the geometries of other OSM elements

Implementation - 3. Modified Elements

- **Solution:** Fetch all elements from the SPARQL endpoint that have a modified element as a member

➢ Add *indirectly* modified elements to the change file

Implementation - 3. Modified Elements

- **Solution:** Fetch all elements from the SPARQL endpoint that have a modified element as a member

➢ Add *indirectly* modified elements to the change file

```
<modify>
  <node id="31413949" lat="48.0141735" lon="7.8342196"/>
  <way id="606463742">
    <nd ref="1315201886"/>
     ...
    <nd ref="31413949"/>
  </way>
</modify>
```

➢ Delete and insert ***only*** geometry triples of the *indirectly* modified elements

Evaluation – Correctness

- Verify that all triples are correctly inserted and deleted at the SPARQL endpoint:

Evaluation – Correctness

- Verify that all triples are correctly inserted and deleted at the SPARQL endpoint:
  - Experiment for the OSM Germany dataset (6.3 Billion Triples)
  - Build one graph containing the latest OSM data (G_{latest})
  - Build another graph containing OSM Data that is one week old ($G_{updated}$)

Evaluation – Correctness

- Verify that all triples are correctly inserted and deleted at the SPARQL endpoint:
  - Experiment for the OSM Germany dataset (6.3 Billion Triples)
  - Build one graph containing the latest OSM data (G_{latest})
  - Build another graph containing OSM Data that is one week old ($G_{updated}$)

```
SELECT ?s ?p ?o WHERE {
  {
    { GRAPH <http://example.com/updated> { ?s ?p ?o . } }
    MINUS
    { GRAPH <http://example.com/latest> { ?s ?p ?o . } }
  }
  UNION {
    { GRAPH <http://example.com/latest> {?s ?p ?o . } }
    MINUS
    { GRAPH <http://example.com/updated> { ?s ?p ?o . } }
  }
}
```

Evaluation – Performance

- Publication interval of OSM change files is a natural benchmark
  - ➢ Changes files must be processed faster than their publication interval

Evaluation – Performance

- Publication interval of OSM change files is a natural benchmark
  - ➢ Changes files must be processed faster than their publication interval

- Mean time required to generate SPARQL update operations and mean time required to apply them on a SPARQL endpoint

		Generating Updates			Applying Updates		
Dataset	N_{triple}	t_{min}	t_{hour}	t_{day}	t_{min}	t_{hour}	t_{day}
P	134.8 B	6.7 s	73.0 s	21.6 min	48.5 s	180.0 s	-
EU	37.7 B	1.3 s	19.1 s	5.4 min	10.2 s	41.5 s	26.1 min
DE	4.8 B	0.3 s	5.5 s	74.9 s	0.8 s	4.5 s	92.2 s
BW	0.6 B	0.1 s	2.1 s	33.0 s	40 ms	0.5 s	9.6 s
FR	0.2 B	68 ms	1.5 s	27.9 s	7 ms	88 ms	1.4 s

Conclusion

- ***osm-live-updates (olu)*** is the first tool capable of synchronizing a SPARQL endpoint with the OSM database, while preserving the geometries and information of all OSM elements on the endpoint
- ***olu*** generates SPARQL update operations from OSM change files
- ***olu*** can **correctly** and **efficiently** update a SPARQL endpoint
- ***olu*** is open-source and publicly available on GitHub (https://github.com/ad-freiburg/osm-live-updates)

Sources

[1] OpenStreetMap stats – OpenStreetMaps; https://planet.openstreetmap.org/statistics/data_stats.html

[2] Bast, H., Brosi, P., Kalmbach, J., & Lehmann, A. (2021). An Efficient RDF Converter and SPARQL Endpoint for the Complete OpenStreetMap Data. Proceedings of the 29th International Conference on Advances in Geographic Information Systems, 536–539. Presented at the Beijing, China. doi:10.1145/3474717.3484256

[3] Bast, H., & Buchhold, B. (2017). QLever: A Query Engine for Efficient SPARQL+Text Search. Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, 647–656. Presented at the Singapore, Singapore. doi:10.1145/3132847.3132921