What to expect from MySQL 8.0?

Norvald H. Ryeng
Software Engineer

FOSDEM 2017
Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle’s products remains at the sole discretion of Oracle.
Agenda

1. Goals
2. What will we do in 8.0?
3. Library
4. Standard compliance
5. Axis order
6. Data types
Goals

- Ease of use
  - Built-in GIS functionality
  - GIS data and functions as first class citizens
- Be the best DBMS for web maps
  - Global data
  - Data import/export
- Mobile devices
  - Tracking
  - Routing?
What will we do in 8.0?

- Geography
  - The framework to handle SRSs
  - Geographically enabling as many functions as possible
    - First add the functionality to Boost.Geometry
- Make the upgrade as easy as possible

8.0 is still in development — test it and give us feedback!
Library

- We don't want to maintain a GIS library alone
  - We're happy to contribute!
- C/C++
- Follow OGC standards
- Handle both Cartesian and geographic computations
- Started on Boost 1.55.0, now on 1.63.0
  - MySQL 5.7 requires Boost 1.59.0
  - Maintain our own patches/header files on top of Boost for bug fixing
Standard compliance

- Follow SQL/MM, OGC and other standards as closely as possible
  - Some things are not well-defined
  - Some things are just stupid
  - The MySQL SQL dialect is not object oriented
- The standards disagree
- Some things are not standardized
“Going forward, for new standards, coordinate values shall be listed in the axis order as specified by the referenced coordinate reference system (CRS).”

— Axis Order Policy and Recommendations, OGC 08-038r5
Axis order

- All geographic SRSs in the EPSG Dataset are latitude-longitude
- MySQL uses the EPSG Dataset
- MySQL follows the recommendation and uses the axis order defined by the SRS
  - But it can be overridden:
    ```sql
    ST_GeomFromText('POINT(50.8119483 4.3826169)', 4326, 'axis-order=lat-long')
    ST_GeomFromText('POINT(50.8119483 4.3826169)', 4326, 'axis-order=long-lat')
    ```
Same data types for Cartesian and geographic

```sql
SELECT ST_Distance(
    ST_GeomFromText('POINT(50.8119483 4.3826169)', 0),
    ST_GeomFromText('POINT(4.3826169 50.8119483)', 0)) AS distance;
```

**distance**
65.66099015779498

Unitless

```sql
SELECT ST_Distance(
    ST_GeomFromText('POINT(50.8119483 4.3826169)', 4326),
    ST_GeomFromText('POINT(4.3826169 50.8119483)', 4326)) AS distance;
```

**distance**
6712322.144680507

Meters
Same data types for Cartesian and geographic

```sql
SELECT ST_Distance_Sphere(
    ST_GeomFromText('POINT(50.8119483 4.3826169)', 0),
    ST_GeomFromText('POINT(4.3826169 50.8119483)', 0)) AS distance;
```

**distance**

6719621.730158467  Meters

```sql
SELECT ST_Distance(
    ST_GeomFromText('POINT(50.8119483 4.3826169)', 4326),
    ST_GeomFromText('POINT(4.3826169 50.8119483)', 4326)) AS distance;
```

**distance**

6712322.144680507  Meters
Prepare now for the upgrade from 5.7 to 8.0

- Think through your use of SRIDs
  - Use SRID 0 if you're unsure
  - May affect your query results after upgrade
- Use longitude-latitude ordering in 5.7
  - It matches the storage format (x=longitude, y=latitude)
- But remember that import and export functions follow SRS defined axis order in 8.0
Follow our progress on
www.mysqlserververteam.com
Integrated Cloud
Applications & Platform Services