MyTile
A Cloud-Native Storage Engine based on TileDB

Seth Shelnutt
Chief Technology Officer

9/14/2020
Outline

- What is TileDB
- MyTile Overview
- Use Cases of MyTile
- Internal Usage of MyTile At TileDB, Inc.
What is TileDB?
What is TileDB?

TileDB the universal data engine
It is built upon built on **multi-dimensional arrays**

This enables storing and accessing:

- Dense arrays (e.g., satellite images)
- Sparse arrays (e.g., LiDAR, genomics)
- Dataframes (any data in tabular form)
- Key-values (mappings between keys and values)
TileDB: Universal Data Engine

Efficient integration

**Open Source**
- **Store all your Data**
  Model any structured data and slice any segment or region directly from the cloud
- **Analyze with any tool**
  Seamlessly access data with any data science tool and perform ML & analytics at scale

**TileDB Cloud**
- **Share at planet-scale**
  Share data sets+ code within & beyond organizational boundaries. Manage access & audit activities.
- **Eliminate deployment hassles**
  Spin up a Jupyter notebook and scale out computations - all serverless

TileDB Cloud

Open Source
TileDB Core Features

- Cloud storage (AWS S3, Google Cloud Storage, Azure Blob Storage)
- Tiling (i.e., chunking) for fast slicing
- Columnar
- Multiple compression, encryption and checksum filters
- Fully multi-threaded implementation
- Parallel IO
- Data versioning (rapid updates, time traveling)
- Array metadata
- Array groups
- Embeddable C++ library
Immutability is important for cloud object-stores

- Time Traveling is native due to MVCC Design
- Every write is immutable
  - Updates are treated as writes
  - Crash safe with object immutability
- Consolidation mechanism supported
  - Combine fragments, purge history
  - Reduced disk space
  - Can improve performance
ACID support:
- All writes in TileDB are atomic
- Consistency is determined by the storage location
  - Eventual consistency of cloud object stores is natively handled
  - TileDB always returns valid results, never corrupt or invalid results
- Isolation and durability are inherent in the MVCC design
  - Every write gets its own fragment (folder) with a unique timestamp + uuid
  - Transactions and locking are not needed
    - Conflicting writes at the same timestamp are handled by the UUID of the write
MyTile
MyTile Overview

MyTile is a MariaDB Storage Engine based on TileDB Embedded
It inherits all functionality of TileDB

- Uses the TileDB Embedded C++ Library
- TileDB Arrays can be on remote cloud object store (S3, GCS, Azure)
- Fully dynamic discovery of existing arrays
  - SELECT * from `s3://my-bucket/my-array`. *It just works!*
  - Complete interoperability with existing arrays and other APIs
- Current maintained outside MariaDB source, hope to upstream soon
Query Optimizations

- Fully supports condition pushdown
  - Increased performance for large number of query types
  - Predicates on dimensions are intercepted and pushed to TileDB
- Multi-range read optimization built in
  - High performance joins between TileDB and any other tables!
  - Supports bulk key access for MRR optimization for joins
Transactions and Locking

- Completely lock free multi-reader/multi-writer design
  - TileDB storage engine and format natively handle conflicts
  - No need for any locking inside MariaDB
- ACID-like support without any locking or transactions
  - Eventual consistency of cloud object stores main limitation
- Transactions only supported for bulk insertion of data
  - No general transaction support at this time
  - TileDB itself does not have transactions
  - On roadmap for development to support only single mariadb server transactions
Embedded Usage

- MyTile is able to be built and bundled into a embedded MariaDB instance
- Customers use this for data exploration and manipulation
  - Works great to showcase sql in a jupyter notebook without a server
- Embedded MariaDB can be used by any client that links against mariadb c connector
  - Opens the door for anyone to run sql queries on remote datasets without needing a server setup
Backups

- TileDB arrays are self contained, simply copy the folder of the array
  - An array is always in a consistent state even in the middle of a write
- Hot backups and incremental backups are simple
  - Consistent and straightforward structure means it’s easy to copy only parts of an array after a given timestamp (last backup)
  - Rsync or aws s3 sync commands work great with TileDB arrays
What MyTile Is Not

- A general replacement for InnoDB
- Designed for highly transactional datasets
- Designed for full text searching and indexing (Mroonga)
Storage Engine Comparison - InnoDB

Advantages compared to InnoDB

- Cloud Native, store data on S3, Azure, GCS and more
- Many more compressors (zstd, lz4, RLE and more)
- Significantly reduced write amplification
- Supports time traveling
- Better performance at terabyte scale

Disadvantages:

- InnoDB is a very mature and battle tested storage engine
- InnoDB has great support for replication and backups through tools like mariabackup/xtrabackup
- InnoDB has significantly better single record insert performance
Storage Engine Comparison - MyRocks

Advantages compared to MyRocks

- Cloud Native, store data on S3, Azure, GCS and more
- Interops with any TileDB Integration or API
  - no special format or semantics
- Additional compressors such as run length encoding, double delta and more

Disadvantages:

- MyRocks is more mature, with additional MariaDB statistics and parameters
- MyRocks has support for MariaDB transactions and checkpoints
Use Cases
Use Case: #1 Geospatial - AIS

- AIS ship location data
- Data is stored as a Sparse TileDB Array
- Employees access data via:
  - Python
  - MariaDB
  - Embedded MariaDB
- Use the best tool for the job
- Efficient SQL pushdown of bounding box
Use Case: #2 Time Series

- Common datasets
  - Stock market, asset trading
  - Stored as sparse TileDB array
- Customer uses embedded mariadb
  - Allows fast and easy data exploration
- Uses spark, dask and other for large production jobs
  - Experimenting with distributed SQL via a "cluster" of independent MariaDB servers with custom query sharding.
Use of MyTile Internally
TileDB, Inc Offerings

Open-source
1. TileDB Embedded
A powerful open-source storage engine that works with all data science tools
- Captures all data via arrays
- Fast and cloud-optimized
- Data versioning built-in

Commercial product
2. TileDB Cloud
A scalable and easy-to-use collaboration and data analytics platform
- Serverless SQL and lambdas
- Jupyter notebooks
- Sharing with access control

Solve the **storage and data management pains** and enable **scalable compute**
MyTile is used to power TileDB Cloud's serverless SQL offering

- We run MariaDB servers in an autoscale cluster
- Allow users to dynamically access with their data with a simple query string

```python
tiledb.cloud.sql.exec("select avg(a) FROM `tiledb://TileDB-Inc/quickstart_dense`")
```
- Returns data directly as a pandas dataframe (python) or JSON (any language)