CRATEDB: A SEARCH ENGINE OR A DATABASE? BOTH!

HOW WE BUILT A SQL DATABASE ON TOP OF ELASTICSEARCH AND LUCENE

Maximilian Michels
@stadtlegende
max@crate.io
mxm@apache.org
WHY ARE WE TALKING ABOUT THIS?

• Traditional databases are well-researched and there are plenty of them (Postgres, MySQL, Oracle…)
  • Scalable search using these can be tricky

• Search engines are databases optimized for search and scale (Lucene, Solr, Elasticsearch)
  • You can’t typically use SQL with Search Engines

• Why not stick with an mature query language standard which everybody knows?
“A scalable SQL database optimized for search without the NoSQL bullshit.”

CrateDB
Cratedb in a Nutshell

- Since 2014: https://github.com/crate/crate
- Apache 2.0 licensed (community edition)
- Built using Elasticsearch, Lucene, Netty, Antlr, ...
- SQL-99 compatible
- REST / Postgres Wire Protocol / JDBC / Python ...

CrateDB
WHAT TO EXPECT

• What is great about CrateDB
  • Easy to setup
  • No funny APIs / SQL
  • Great scale out - Massive reads / writes
  • Container aware

• Not so great
  • Transactions
  • Foreign keys
USING CRATEDB

CrateDB
CRATEDB IS JUST LIKE A SQL DB

- SQL is the only query API

- **CREATE TABLE** fosdem.speakers (id int PRIMARY KEY, name string)

- **CREATE TABLE** fosdem.talks (id INT PRIMARY KEY, title STRING, abstract STRING, speaker INT);

- **INSERT INTO** fosdem.speakers (id, name) VALUES (1, 'max')

- **INSERT INTO** fosdem.talks (id, title, abstract, speaker) VALUES (1, 'Talk about CrateDB', 'bla', 1)

- **SELECT * FROM** fosdem.talks t1 **LEFT JOIN** fosdem.speakers t2 **ON** t1.id = t2.id
BUT THERE IS MORE

• CrateDB denormalized (no joins necessary)

• CREATE TABLE fosdem.speakers (name STRING, talk OBJECT AS (title STRING, abstract STRING))

• INSERT INTO fosdem.speakers (name, talk) VALUES ('max', {title = 'CrateDB', abstract = 'Lorem ipsum'})

• SELECT talk['title'] as title FROM fosdem.speakers ORDER BY title
• CREATE TABLE fosdem.speakers (name STRING, talk OBJECT AS (title STRING, abstract STRING))

• CLUSTERED BY name into 4 shards
CLUSTERING / REPLICATION

- CREATE TABLE fosdem.speakers (name STRING, talk OBJECT AS (title STRING, abstract STRING))
- CLUSTERED BY name into 4 shards
- WITH (number_of_replicas = 1)
PARTITIONED TABLES

- **CREATE TABLE** fosdem.speakers (name STRING, talk OBJECT as (title = STRING, abstract = STRING), year INT)

- **CLUSTERED BY** name into 4 shards

- **PARTITIONED BY** (year, …)

- **WITH** (number_of_replicas = 1)
MORE FEATURES

• Aggregations
• Geo search
• Text Analyzers
• UDFs
• Snapshots
• User management
• Schema / Table privileges
• SSL encryption
• MQTT Ingestion
ARCHITECTURE

+ CrateDB
ON THE SHOULDERS OF GIANTS

• **CrateDB**: Distributed SQL Execution Engine
• **Antlr**: Parsing of SQL statements
• **Netty**: REST, Postgres Wire Protocol, Web interface
• **Lucene**: Storage, Indexing, Queries
• **Elasticsearch**: Transport, Routing, Replication
INTRODUCTION TO Lucene

• Lucene stores documents which are CrateDB’s rows

• Documents have fields

  • { _id : ‘123’,
    name : ‘Bob’,
    title : ‘How I Learned to Stop Worrying and Love the Bomb’,
    text  : ‘Lorem ipsum…’
  }

• Fields are indexed for efficient lookup

• Fields have column store for efficient aggregation
INTRODUCTION TO ELASTICSEARCH

• Elasticsearch core concepts revolve around indices, shards, and replicas

• An index is a document store with n parts, called shards

• Each shard has 0 or more replicas which hold copies of the shard data

• Replicas are not only useful for fault tolerance but also increase the search performance
HOW TABLES RELATE TO INDICES AND SHARDS

- Each table in CrateDB is represented by an ES index with a mapping.
- Each partition in a partitioned table is represented by an ES index.
- Partition indices are created by encoding the partition value in the index name.

```json
"properties": {
  "name": {"type": "keyword"},
  "talks": {"dynamic": "true",
    "properties": {
      "abstract": {"type": "keyword"},
      "title": {"type": "keyword"}
    }
  }
}
```
FROM QUERY TO EXECUTION

- SELECT name, count(*) as talks FROM fosdem.speakers
  WHERE room = 'hpc' AND year = 2018 GROUP BY name ORDER BY name
ARCHITECTURE HIGHLIGHTS

• Distributed storage / Distributed query execution

• Masterless

• Replication

• Only ephemeral storage needed (Container aware)

• Optimized for search: Indexing of all fields with Lucene (tuneable)
HANDS-ON

CrateDB
WHAT CAN YOU DO WITH CRATEDB?

- Monitoring (IoT, Industry 4.0, Cyber Security)
- Stream Analysis
- Text Analysis
- Time Series Analysis
- Geospatial Queries
### CrateDB Web Interface

#### Shards

<table>
<thead>
<tr>
<th>Shard</th>
<th>Started Primary</th>
<th>Started Replica</th>
<th>Initializing</th>
<th>Relocating</th>
<th>Unassigned</th>
</tr>
</thead>
<tbody>
<tr>
<td>fhv</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>parkplatz</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>part</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>person</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>replicas</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>test</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>home</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ganttner</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>13</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>14</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>15</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Diagram

A diagram showing the distribution of shards across different databases and tables.
CONCLUSION

CrateDB
WHAT WE HAVE LEARNED

• Elasticsearch used Lucene and Netty to build a distributed search engine

• CrateDB used Elasticsearch, Lucene, and Netty to build a distributed SQL database

• CrateDB is perfect when you
  • want or have to use SQL
  • store large amounts of structured or unstructured data
  • have many thousands of queries per second
SEE FOR YOURSELF!

• Try out CrateDB
  • Download from https://crate.io/download/
  • or $ curl try.crate.io | bash
  • or $ docker run crate
  • or build from source https://github.com/crate/crate
• Check out https://crate.io/docs
• Contributions welcome
  • Check out https://github.com/crate/crate/blob/master/devs/docs/index.rst
  • Check out the issues
  • Stackoverflow
  • Join our Slack channel
THANK YOU!

Maximilian Michels
@stadtlegende
max@crate.io
mxm@apache.org