SAP Vora: A Distributed Computing Platform for Enterprise Data Lakes

Mark Waldaukat
May 3rd, 2017
WE COLLECTED ALL THIS DATA NOW WHAT?
SAP Vora
A scale-out database system
Agenda

1. Overview
2. Distributed Computing Platform
3. Extensibility
4. Integration
5. Containerization
6. Data flow processing
7. Outlook
Overview
Organization

**SAP BIG DATA**
- New organization within SAP
- SAP Vora is the core product
- +350 people worldwide (Walldorf, Palo Alto, Istanbul, Waterloo, Potsdam)
- Collaboration with SAP groups
  - SAP Cloud Platform
  - Internet of Things Applications (SAP Leonardo)
  - Data Warehousing (SAP BW)

**SAP Vora**
- Foundation for Big Data services
- Started ~3 years ago, first delivery Q3/2015
SAP Vora

What is Vora?
- Scale-out distributed database system
- Heterogeneous data types in one system
- Integrates with existing Big Data infrastructure (Hadoop, Spark)
- Traditional use cases (OLAP, OLTP)
- Future use cases (ML, data ingestion)
Overall Design Goals

**OLxP Workloads**
- Allow for analytic workloads (OLAP, read-only)
- Allow for transactional workloads (OLTP, read-write)
- One common infrastructure to scale both

**Scalable**
- Grow processing and storage capabilities with data
- Run on shared-nothing clusters
- Failure tolerant
- On-demand
Overall Design Goals /2

Extensible

- General compute platform
- Easy to add new functionality (compute engines)

Integration

- Leverage existing Big Data infrastructures (i.e., Hadoop cluster)
- Native integration into enterprise systems (SAP HANA)
- Native integration into state-of-the-art big data tooling (i.e. Spark)

Enterprise Features

- Unified SQL interface
- Currency conversion (i.e., for Data Warehousing)
- Hierarchies handling (i.e., for OLAP)
Overall Design Goals /3

**Independent Components**

- Easy distribution on a cluster
- Database components as a service
- Independent development
  - Collaborating teams
  - Independent release cycles
Distributed Computing Platform
General Architecture

**Components**
- Transaction Handling (Coordinator, Broker)
- Distributed Shared Log (DLOG)
- Distributed Query Processor (DQP)
- Query Engines
  - Relational (Disk & In-Memory)
  - Graph
  - DocStore
  - Time-series
- Catalog*
- Landscape Service*
- Communication layer (V2Net)*
Transaction Handling

**Transaction Coordinator**
- Session Management
- Meta-Data Updates
- Orchestrates the request
  - Parsing, Analysis, Optimization
  - Scheduling
  - Failure Handling
  - Result serialization

**Transaction Broker & Lock Manager**
- Holds Transaction State
- Locking
- Handle Writes
Query Processing
Distributed Shared Log (DLOG)

**Function**
- Transaction log as a service
- Persistent message bus
- IoT stream ingestion

**Features**
- Strong consistency (global ordering)
- Strong durability (syncs to disks and replicas)
- Partitioned, replicated, fault-tolerant
Distributed Query Processor (DQP)

**DQP Nodes**
- Engines can be queried if they support a plan
- Engines execute plan fragments

**Global Plan Generation (Optimizer)**
- Central component creates the query plan
- Plan spans multiple (remote) engines
- Split into multiple plan fragments (local plans)
- Serialized to be sent to remote execution engines
- Deserialized on the receiving side

**Plan Protocol**
- Support for inheritance (important for representing query operators)
- Tooling for easy visualization and comparison
Extensibility
SAP Vora Technical Background

**Vora Engines**

- **SAP Vora**
  - Started in 2013 by HANA Research Teams code named “velocity”
  - Shared concepts of libraries with HANA but with independent code base
  - As started from scratch: much leaner footprint compared to HANA

- **Engines**
  - Relational (In-memory & Disk)
  - Time-series
  - Document Store
  - Graph
SAP Vora Technical Background
The Vora Relational Engines (in-memory and disk)

- **SAP Vora Relational Engines**
  - Exist in 2 flavors, with in-memory engine and disk flavors
  - Have identical SQL syntax
  - Support various file formats (CSV, Parquet, ORC) and storage layers (HDFS, S3, Swift)

- **In-Memory Relational engine ("velocity")**
  - Implemented from scratch, starting in 2013
  - Only columnar data layout
  - Queries just-in-time compiled to machine code
  - Low memory footprint

- **Disk engine**
  - Optimally suited for large data that shall not be memory resident
  - Creates "indices" for fast data access
SAP Vora Technical Background
The Vora Time Series Engine

**SAP Vora Time Series Engine**
- Efficient processing of time-dependent numerical values for time series analysis
- Shared code base with HANA time series functionality
- Supports CSV file format on HDFS, S3, Swift

**Overall Concepts**
- In-memory processing with lossy compression algorithms
- Time series functionality on distributed and compressed data

**Functionality**
- Auto-Generated columns
- Column functions for trend, median, std deviation, approximations (constant, linear, spline, polynomial)
- Functions for auto correlation, cross correlation, DFT, Histogram, Smoothing
- Data partitioning by (time) ranges
SAP Vora Technical Background
The Vora Graph Engine and Document Store

**SAP Vora Graph Engine ("gravity")**
- In-memory distributed graph database
- SQL-Like interface for graph analytics
- Combination of graph pattern queries with relational operators
- Low memory footprint
- Distributed architecture

**SAP Vora Document Store**
- Storage for semi-structured data in JSON format
- Compressed in-memory representation with optimized data access
- Compiled query execution with NUMA awareness
Integration
Connectivity

SAP HANA

In-memory store

SAP Vora

Enterprise features

SAP Vora spark extension

hadoop

Spark
SAP HANA Wire

HANA Wire Protocol
- Native HANA protocol for data and plan exchange
- Compact (native data types)
- Vora is connected like a HANA machine

Smart Data Access
- HANA data federation layer
- Registration of remote sources and virtual tables
- Push-down of supported plan fragments
- Join re-allocation
SAP Vora Spark Extension

**Spark Integration**
- State-of-the-art big data platform
- Runs on Hadoop clusters (YARN)
- Native integration

**Extension Library**
- Push-down of SparkSQL plan fragments
- Vora DataFrame
- Additional SparkSQL features
  - Views
  - Currency Conversion
  - Hierarchies
SAP Vora SparkSQL example

SparkSQL example

Spark offers a REPL: The Spark shell

Data Scientists can work with Vora in the Spark shell

Create a SapSqlContext:

```java
sqlContext = new SapSqlContext(sparkContext)
```

Create a table:

```sql
sqlContext.sql("CREATE TABLE test (name STRING, age INTEGER) USING com.sap.spark.vora.engines.relational OPTIONS (format "csv", files "/user/vora/test.csv"")")
```

Query the table:

```sql
sqlContext.sql("SELECT AVG(age) FROM test")
```

Containerization
Containerization

What is a container

- Lightweight “virtual machine”
  - Every container has its own isolated user-space
  - Shares kernel with other containers
- The container is self-contained, independent of the host environment

Every SAP Vora component will run in a docker container

- The components are independent of the
  - host operating system
  - actual underlying hardware
  - Hadoop cluster manager
- Installation very simple (helm charts), only prerequisite is a Kubernetes
- Scaling up and down is very simple
Cluster management and container orchestration

**Kubernetes**
- Manages physical cluster resources
- Orchestrates container deployment on the cluster
- Simple configuration of components via YAML files
  - Image for the container
  - Number of replicas
  - Needed CPU/RAM resources
  - Mount persistent volumes
- Kubernetes finds resources for spinning up a container as specified
  - This can be on any node and is transparent for the container
- All big cloud providers offer Kubernetes
  - AWS, Azure
Data flow processing
Data flow processing

Requirements for analytics are changing

ü Customers need to analyze the latest and freshest data
  – Near real-time analytics
  – Working on streams rather than doing batch jobs

ü Examples are:
  – Predictive maintenance
  – Internet of Things

ü Spark Streaming, Apache Flink
VFlow

Modeling data flow with VFlow

- Simple UI to model data flow
- Data flows through reactive components (“in-ports” and “out-ports”)
  - Not intended to stop, reacts to delivered messages
  - Component is unaware of incoming and outgoing connections
- Example IO components
  - Kafka, MQTT, S3, HDFS, SAP Vora, SAP HANA
- Example process components
  - Image processing, Machine learning, any arbitrary program that can be executed inside a container
- Every component runs isolated in a Docker container
Demo video
Outlook
Outlook

More engines
- Text search
- Image/Video processing

Model data processing flows
- Model processing flows
- Manage, store, and execute flows
- Governance
Get Started with SAP Vora

https://www.sap.com/developer/topics/hana-vora.html

On your Hardware

- Developer Edition
- Install Official SAP Vora Release

Cloud-Based

- Amazon AWS with “one-click”

Other

- Get in contact: mark.waldaukat@sap.com
Thank you

Contact information:
Mark Waldaukat
mark.waldaukat@sap.com