Distributed Computing with Spark for Actionable Business Insights!

Stephan Kessler

SAP SE, Spark Developer

SPARK SUMMIT
EUROPE 2016

Who I am

- Stephan Kessler
- SAP HANA Vora Team, Walldorf, Germany
- Integrating SAP engines into Apache Spark since almost two years
- 2nd Spark Summit as a speaker

Today's talk

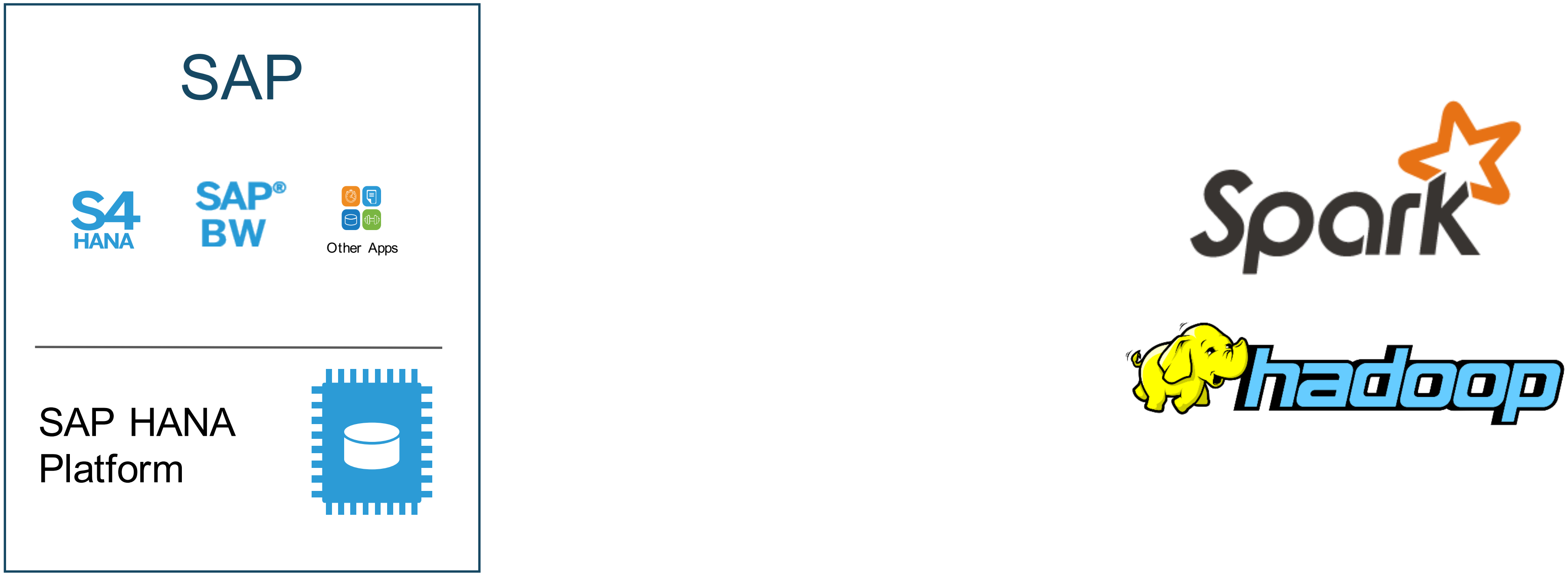
> On average, between 60% and 73% of all data within an enterprise goes unused for business intelligence (BI) and analytics.

The Forrester Wave™: Big Data Hadoop Distributions, Q1 2016," January 19, 2016 by Mike Gualtieri and Noel Yuhanna

> Skills gaps continue to be a major adoption inhibitor for 57% of respondents, while deciding how to get value from Hadoop was cited by 49% of respondents.

Gartner: "Survey Analysis: Hadoop Adoption Drivers and Challenges," May 12, 2015 by Nick Heudecker and Merv Adrian

Current System Landscape

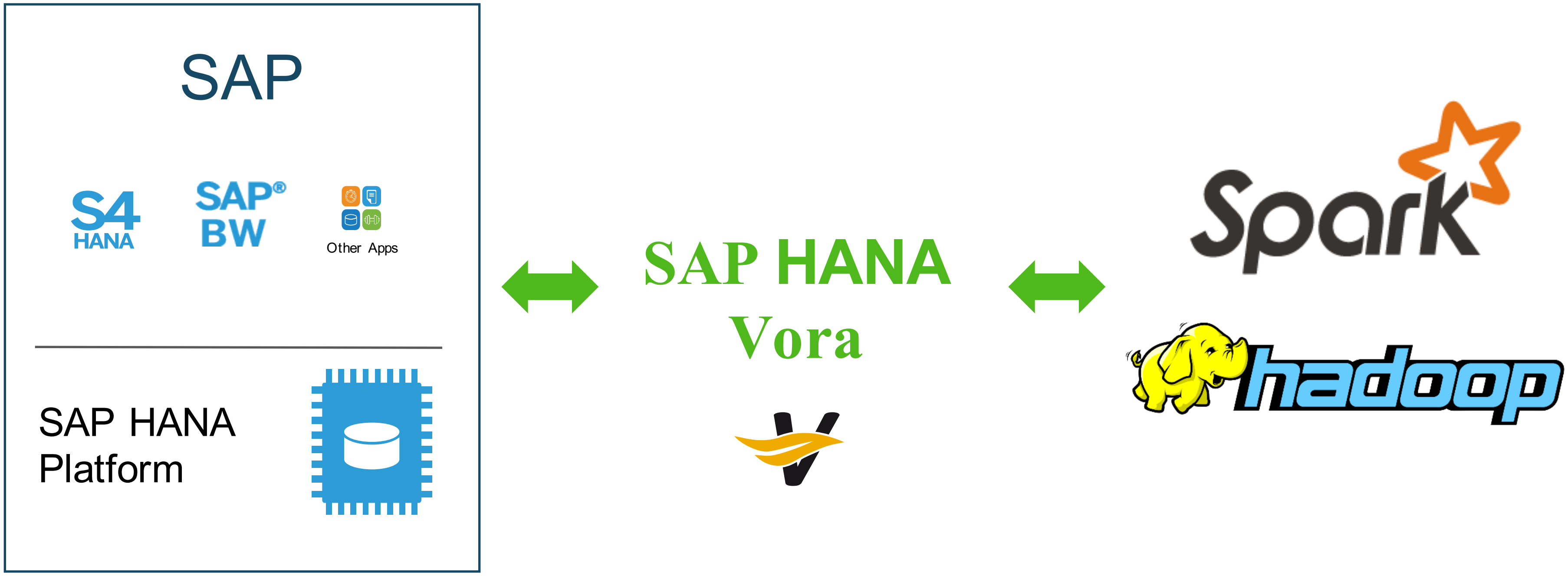


What is missing?

- Business application perspective:
  - Access to Big Data Landscape in a standardized way
  - Similar SQL expressiveness

- Big Data / Data Science perspective:
  - Access to specialized engines to perform analysis close to the data
  - Integration of 'business engines' into Spark

SAP Hana Vora



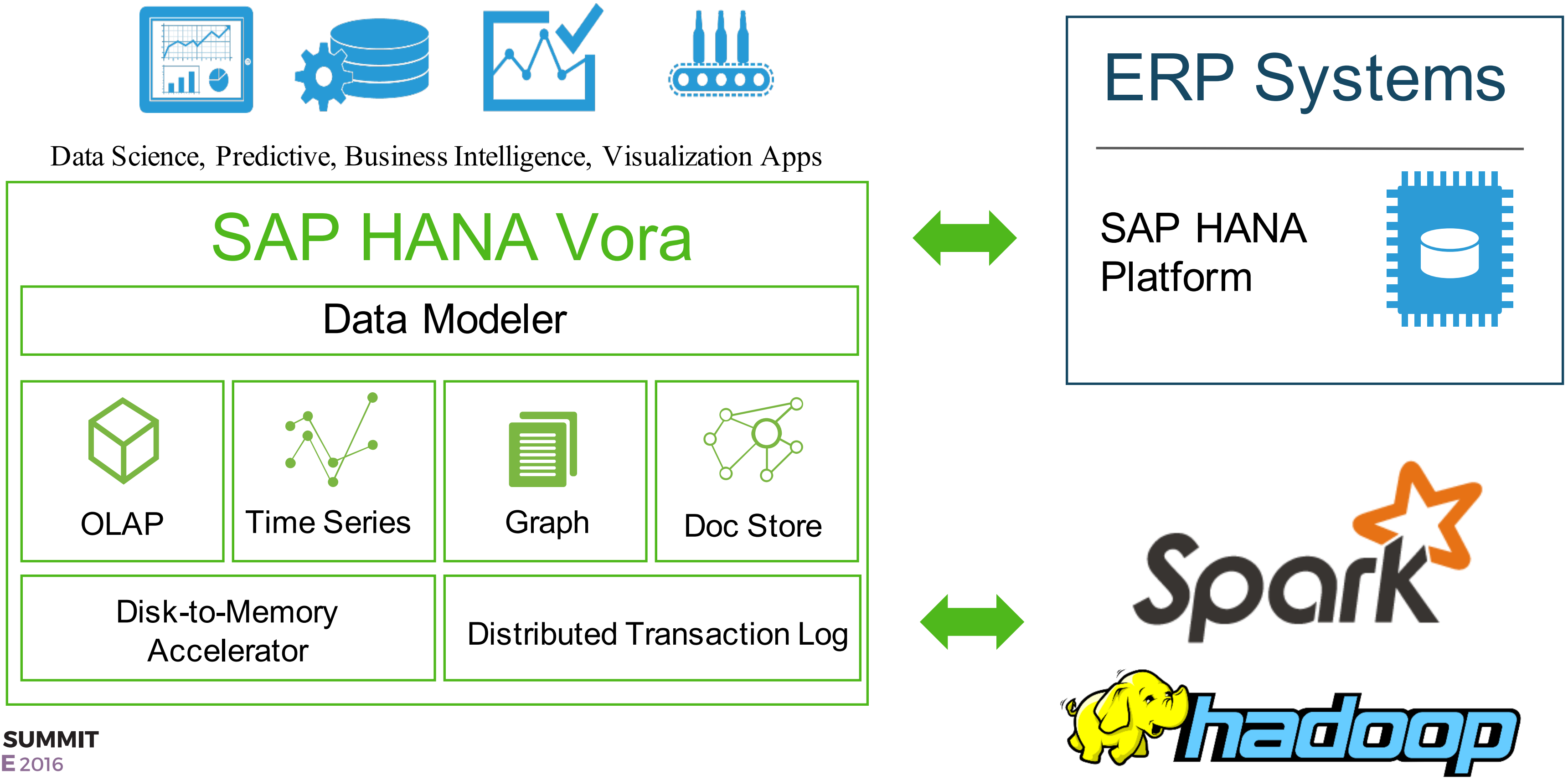
SAP

Other Apps

SAP HANA Platform

SAP HANA Vora

SAP Hana Vora – 10k ft POV



Data Science, Predictive, Business Intelligence, Visualization Apps

SAP HANA Vora

Data Modeler

OLAP

Time Series

Graph

Doc Store

Disk-to-Memory Accelerator

Distributed Transaction Log

ERP Systems

SAP HANA Platform

Spark

hadoop

Goals

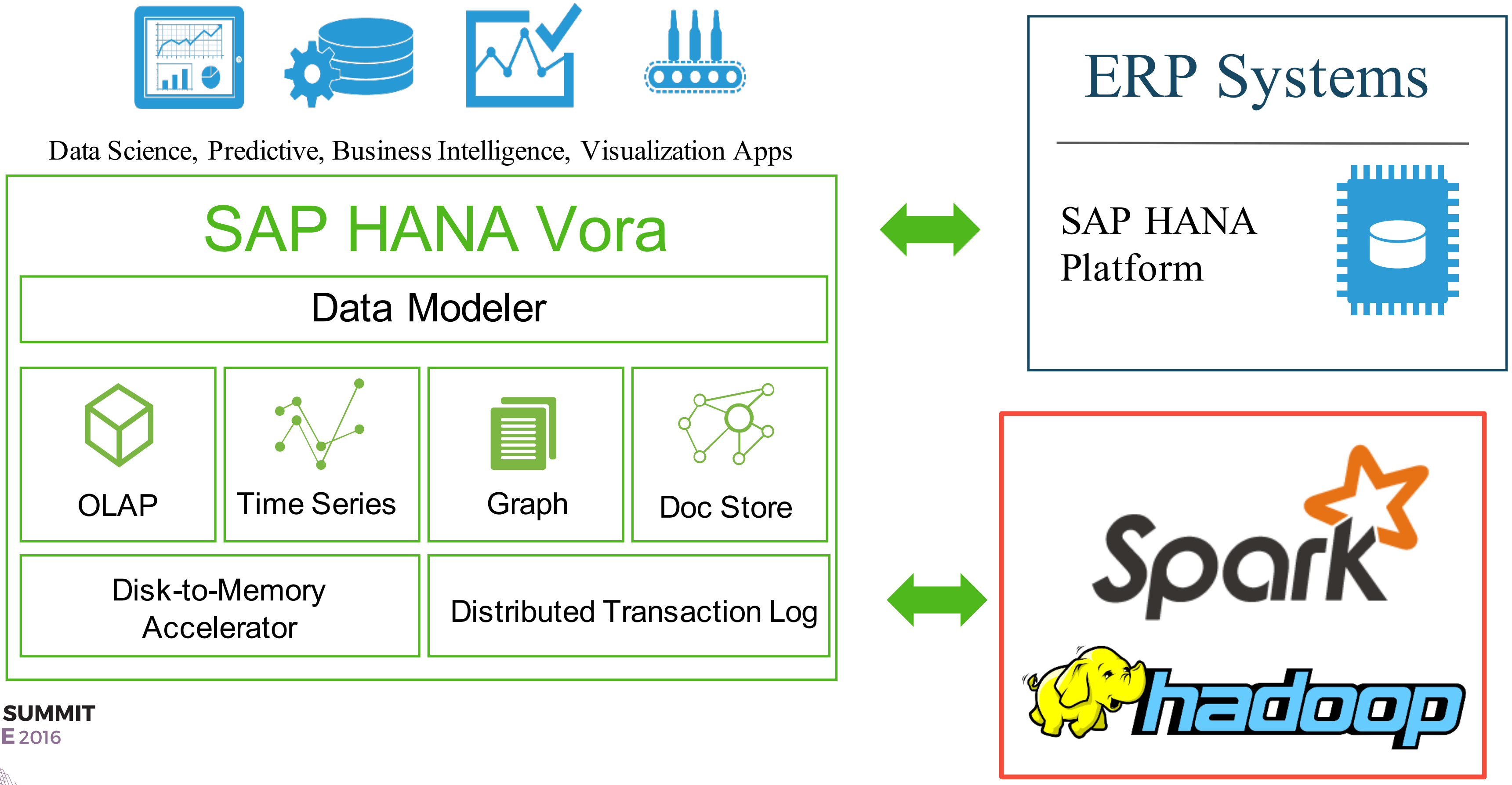
- Make data and functionality available in enterprise applications as well as Spark applications

- Allow an easy consumption, i.e., allow users to write SQL for computation jobs

Agenda

- Business functionality integration in Spark
- Utilizing different data sources in Spark
- HANA Vora 1.3
- Summary & Outlook

Focus on Spark User POV



Data Science, Predictive, Business Intelligence, Visualization Apps

SAP HANA Vora

Data Modeler

OLAP

Time Series

Graph

Doc Store

Disk-to-Memory Accelerator

Distributed Transaction Log

ERP Systems

SAP HANA Platform

Business Functionality in Spark

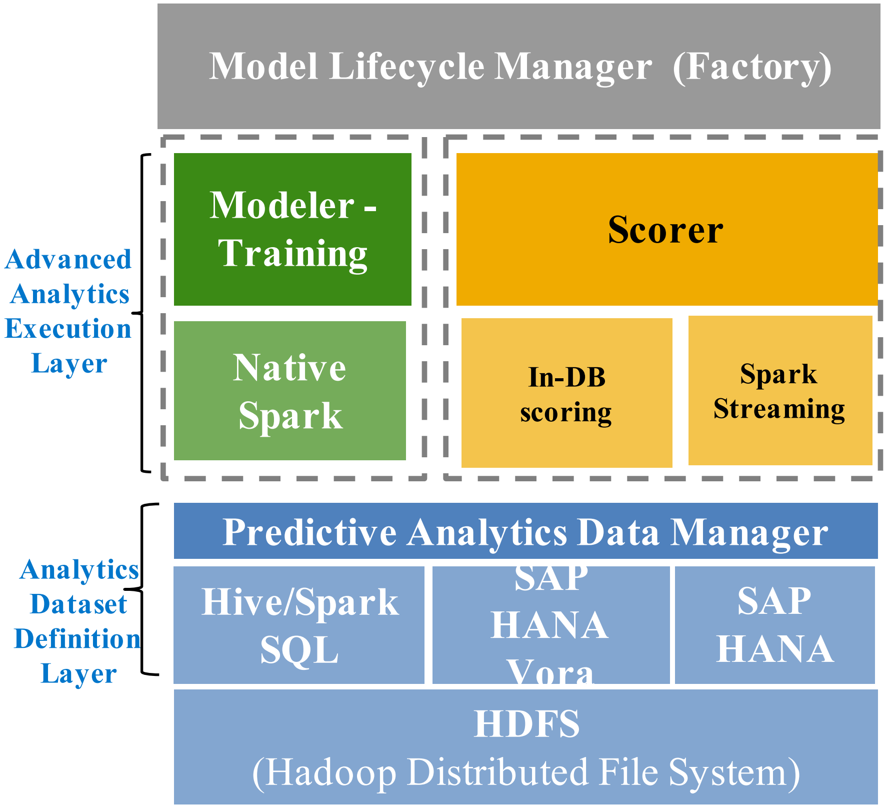
- Questions answered in Spark:
  - What is the sentiment of users for product XYZ?
  - When will a certain part of this machinery fail?
- You might also want to know…
  - What is the sales volume for XYZ?
  - How much did this part cost?

Example: SAP Predictive Analytics and SAP HANA Vora

1. Business Analyst friendly – No coding or in-depth Big Data expertise required
2. Support for end-to-end operationalisation of predictive models on Hadoop
   - Data preparation of Analytical Dataset for modelling
   - Native Spark Modelling – Ultra wide datasets
   - Scoring using In Database Apply or Spark Stream API

- **Usage of different sources (Vora, Hana, Spark, …)**

http://go.sap.com/solution/platform-technology/analytics/predictive-analytics.html



Model Lifecycle Manager (Factory)

Advanced Analytics Execution Layer

Modeler - Training

Native Spark

Scorer

In-DB scoring

Spark Streaming

Analytics Dataset Definition Layer

Predictive Analytics Data Manager

Hive/Spark SQL

SAP HANA Vora

SAP HANA

HDFS
(Hadoop Distributed File System)

Business Functionality in Spark

- Important typical ERP function
  - Currency Conversion (i.e., EUR → GBP)
  - Done via SQL UDF
- Required to analyze enterprise data in Spark

Currency Conversion

TID	USERID	CURRENCY	AMOUNT	ORDERDATE
100	User1	USD	120.10	2014-12-15
101	User1	USD	24.99	2015-01-01
102	User5	EUR	24.11	2015-01-02
103	User3	DBP	542.00	2015-01-02

- Single currency makes transactions comparable
- Conversion not trivial: rates change over time

Currency Conversion

- Introducing an UDF implemented in Spark

```
CC( AMOUNT Double,
       SOURCE_CURRENCY String,
       TARGET_CURRENCY String,
       REF_DATE String )
```

- Converting everything in USD

```
SELECT TID, USERID, ORDERDATE,
       CC( AMOUNT, CURRENCY, "USD", ORDERDATE )
FROM ORDERS
```

Currency Conversion

- Conversion backed by a ‘rates’ table

SOURCE_CUR	TARGET_CUR	REF_DATE	RATE
EUR	USD	2015-01-01	1.32113
EURO	USD	2015-01-02	1.30121
USD	GBP	2015-01-01	0.68960

- Calculation simple, maintenance difficult
- Rates maintained in ERP system
  - Couldn’t we use that?

Specialized Engine: Time Series

- HANA Vora Time Series Engine in a nutshell:
  - Effective Model-Based Compression
  - Multi-representation storage for time series

- Optimized usage for IOT applications
  - Fast injections paths
  - Long running processes in a cluster

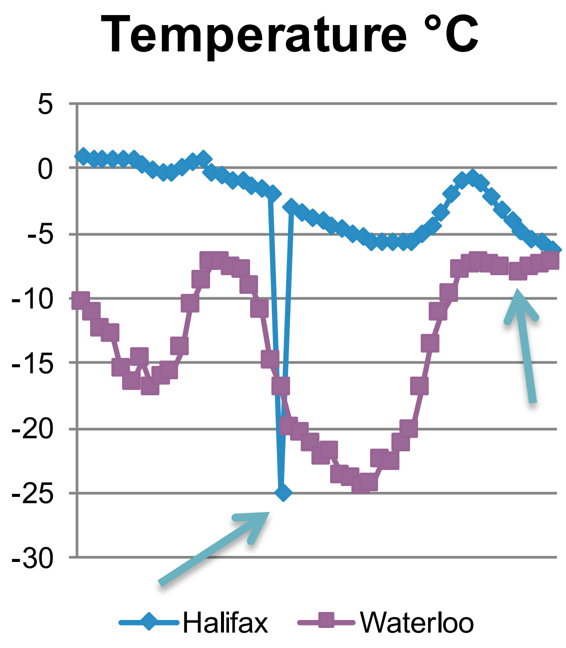
Time Series

Specialized Engine: Time Series

- Query Language: SQL Dialect
- How to use that in Spark?

```
SELECT  STDDEV(val1)
FROM  SERIES  ts
BETWEEN  "2000-01-01",  "2001-12-31″

SELECT  TREND(val1)  OVER  (SERIES)
FROM  SERIES  ts
BETWEEN  "2000-01-01",  "2001-12-31″
```



Agenda

- Business functionality integration in Spark
- Utilizing different data sources in Spark
- HANA Vora 1.3
- Summary & Outlook

What we have seen so far

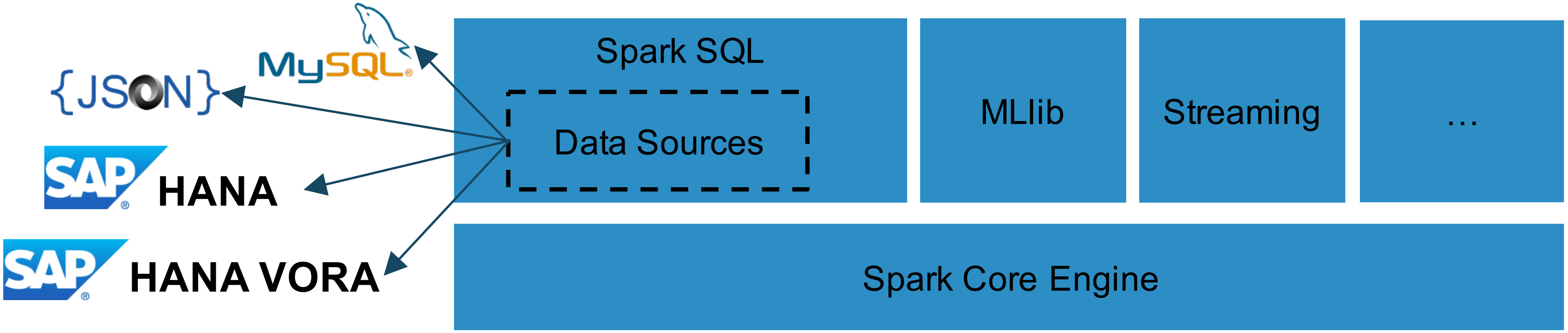
- Currency Conversion:
  - Implementation in Spark SQL
  - Computation could be deferred
- Time Series Engine:
  - Special query language
  - .. but no implementation in Spark

→ "The pushdown of everything"

→ Raw SQL

The Pushdown of Everything

- Spark datasource API



- Limited to *Filters* and *Projects*

Vora Extension to Datasource API

- Datasource indicates its processing capabilities
- Arbitrary parts of logical plan can be computed where the data is
- Details in Spark Summit Europe Talk 2015
  - *https://www.youtube.com/watch?v=QNaf2Z8I8IY*
  - *“The Pushdown of Everything”*

Vora Extension to Datasource API

- Consider this query

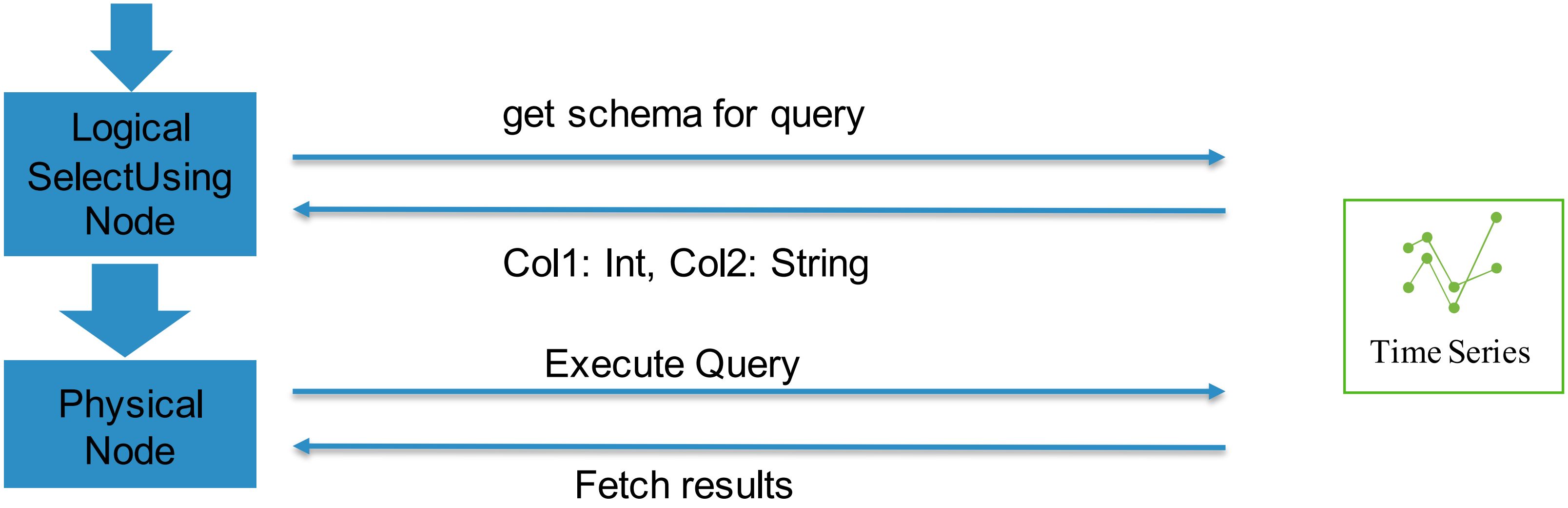
```
SELECT ORDERDATE,
       AVG(CC( AMOUNT, CURRENCY, "USD", ORDERDATE ))
FROM ORDERS
GROUP BY ORDERDATE
```

- Pushing down filters an projects: *SCAN on orders*
- Pushing down arbitrary parts returns:
  - One row per orderdate
  - Converted currency

Raw SQL Extension

- Query Syntax on SparkSQL not supported but in the datasource → Raw SQL

```
``select trend(val1) from ts …`` using com.sap.spark.engines
```



Pushdown & Raw SQL

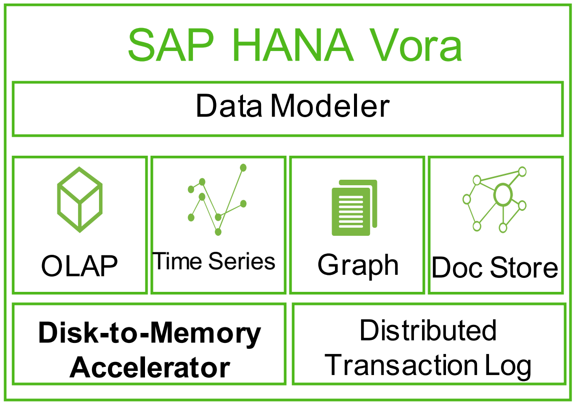
- Both extensions allow to incorporate other data sources extensively
- Computation happens where the data is
- Integration is mostly seamless for Spark developer
- Interfaces are open source:
  - https://github.com/SAP/HANAVora-Extensions

Agenda

- Business functionality integration in Spark
- Utilizing different data sources in Spark
- HANA Vora 1.3
- Summary & Outlook

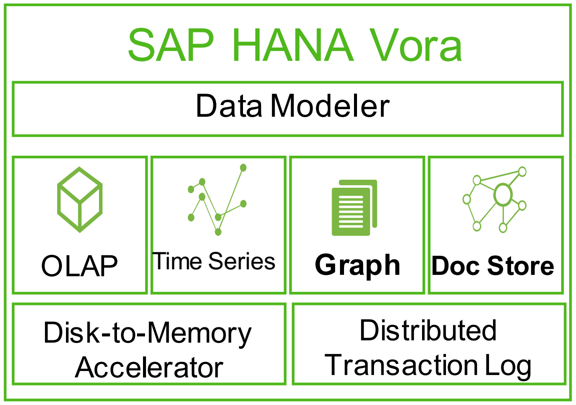
HANA Vora 1.3 - Relational

- Relational Engines in memory and disk
- In-memory
  - Query compilation
  - Columnar data layout
- Disk based
  - Indices for fast data access



HANA Vora 1.3 – Graph & Doc Store

- Graph:
  - In-memory and distributed
  - SQL-Like interface for graph analysis
  - Combination of Graph patterns with relational operators
- Doc Store
  - Stored semi structured JSON
  - Compresses in-memory representation
  - Compiled queries with NUMA awareness



SAP HANA Vora

Data Modeler

OLAP

Time Series

Graph

Doc Store

Disk-to-Memory Accelerator

Distributed Transaction Log

Agenda

- Business functionality integration in Spark
- Utilizing different data sources in Spark
- HANA Vora 1.3
- Summary & Outlook

Summary and Outlook

- Vora allows to combine all data source in an enterprise environment
  - Across different query languages
  - While moving computation close to the data
- Business Insights are driven by all the available data in the enterprise
- Integration into SQL makes it easily consumable

THANK YOU.

Stephan Kessler – stephan.kessler@sap.com

SPARK SUMMIT EUROPE 2016

BACKUP

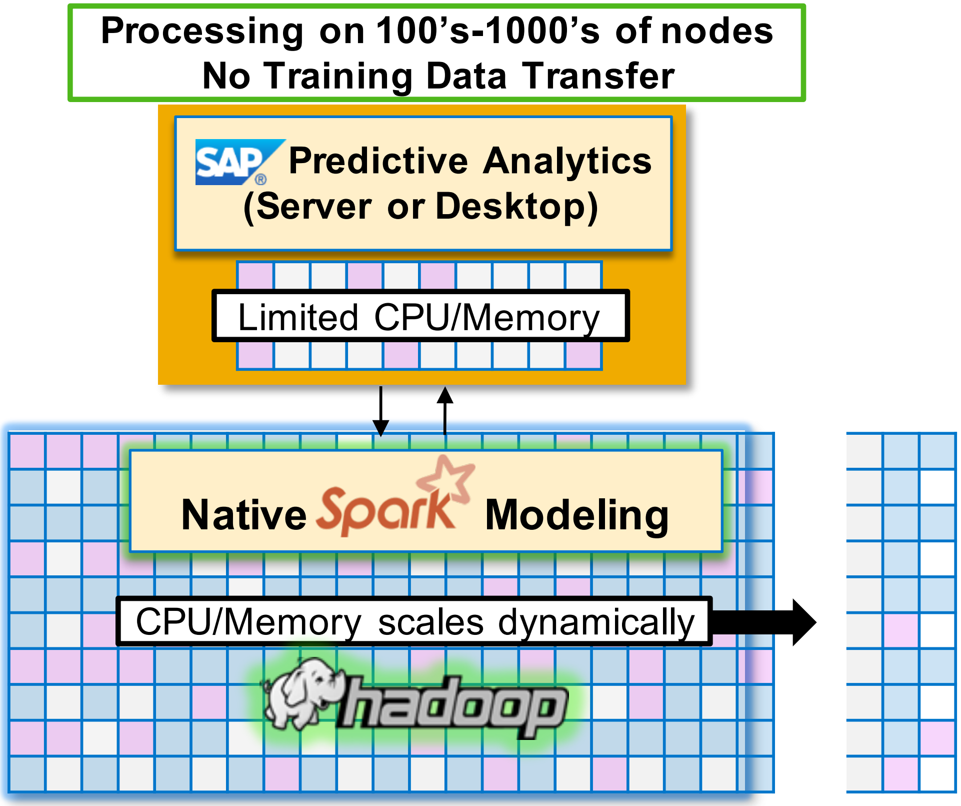
HANA Vora 1.3

- Project started 2013 by HANA Research Teams
- Shared concepts and libraries with HANA but independently developed
- Concepts
  - In memory
  - Distributed engines
  - Low memory footprint

SAP HANA Vora

SAP Predictive Analytics: Optimised for Big Data

- ✓ Native (scala code) Spark approach goes deeper than SQL
- ✓ Performance and Scalability with **Ultra wide datasets**
- ✓ Processing close to the data distributed across the cluster
- ✓ No data transfer



Processing on 100's-1000's of nodes
No Training Data Transfer