Business Intelligence with Apache Druid

Technical Tutorial

Data Science Conference Europe
15-19th November 2020

About mentor

- Vladimir Ivković
- Teaching Assistant at
  Faculty of Technical Sciences,
  University of Novi Sad
- Computing and Control Department
- Chair for Applied Computer Science
- Group for Data Science and
  Information Systems

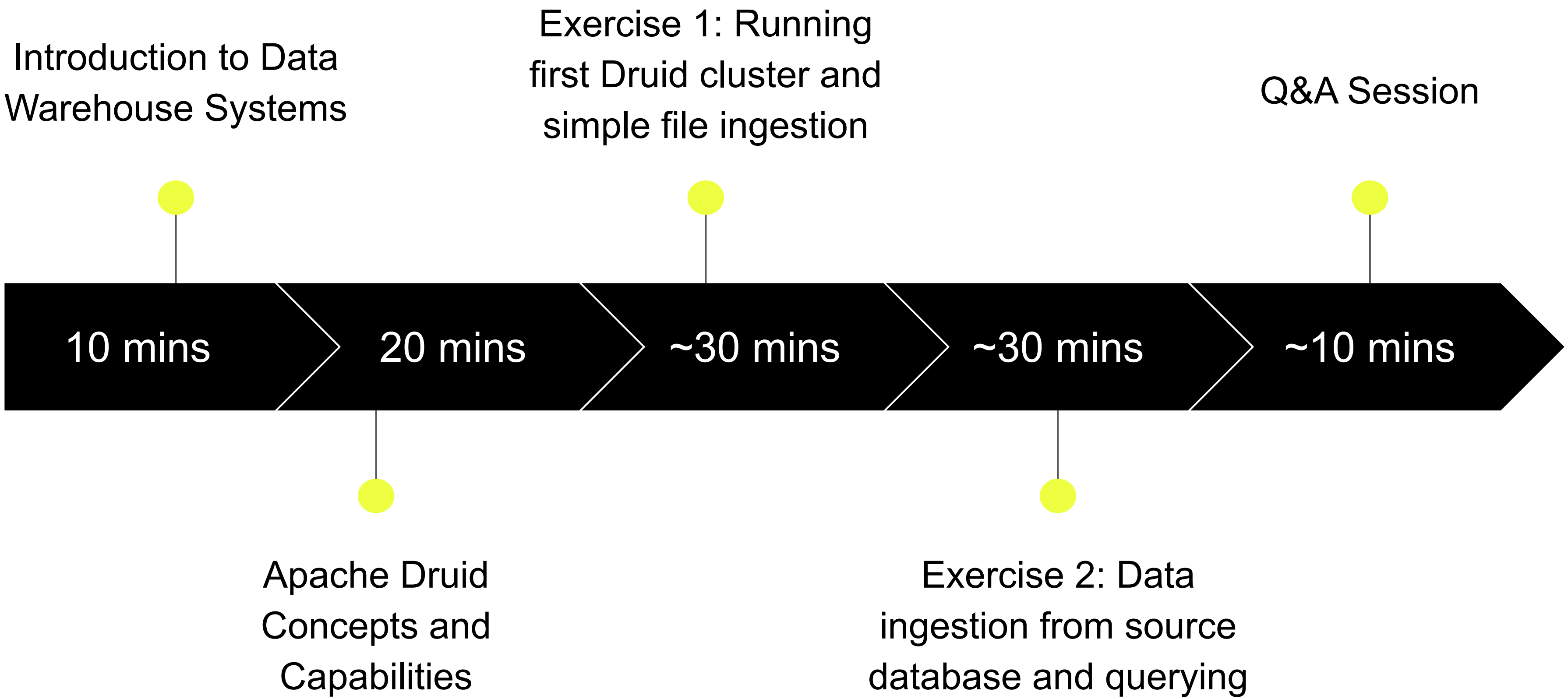
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА НОВИ САД

DAIS Group

Prerequisites

- Software and tools
  - Docker & Docker Compose
  - cURL
  - JDK 8 (optionally)
  - DB client (optionally)
- Knowledge
  - SQL
  - Database Design

Tutorial Agenda



10 mins
Introduction to Data Warehouse Systems

20 mins
Apache Druid Concepts and Capabilities

~30 mins
Exercise 1: Running first Druid cluster and simple file ingestion

~30 mins
Exercise 2: Data ingestion from source database and querying

~10 mins
Q&A Session

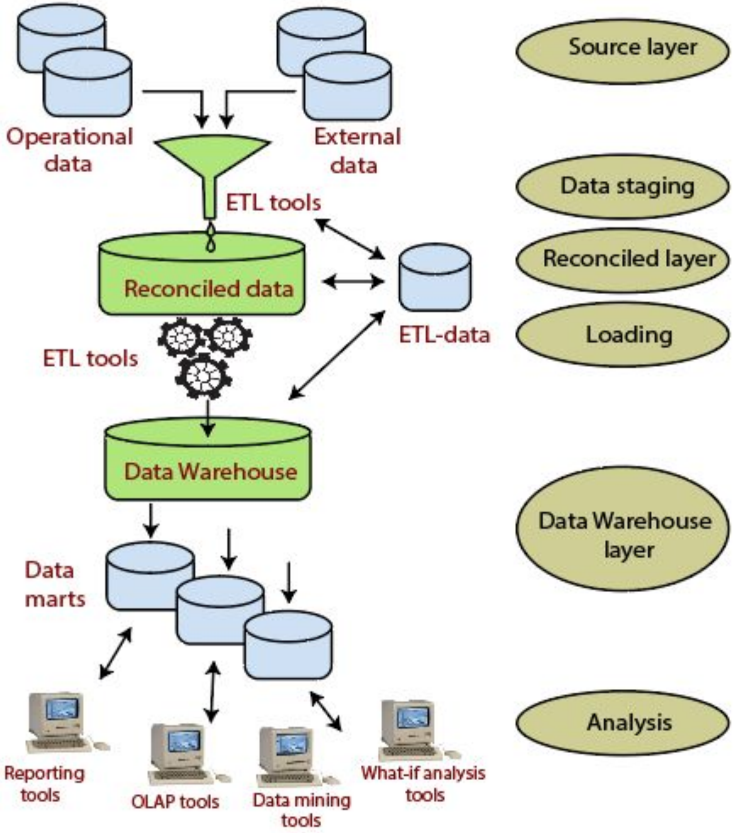
Data Warehouse - Introduction

- **Data Warehousing** is process for collecting and managing data from varied sources to provide meaningful business insights
- **Data Warehouse** is a central repository of information that can be analyzed to make more informed decisions
- Heterogeneous data sources
  - transactional systems
  - relational databases
  - all sorts of files and documents
- DW supports analytical reporting, structured and/or ad hoc queries, and decision making

Data Warehouse - Main Features

- Subject Oriented
    - DW is designed around “subjects” rather than processes.
    - DW offers information regarding a theme instead of companies' ongoing operations. These subjects can be sales, marketing, distributions, etc.
- Integrated
    - the establishment of a common unit of measure for all similar data from the various sources
- Nonvolatile
    - once entered into the data warehouse, data should not change
- Time Variant
    - data warehouse's focus on change over time
    - Historical data for trend analysis

Data Warehouse - Architecture



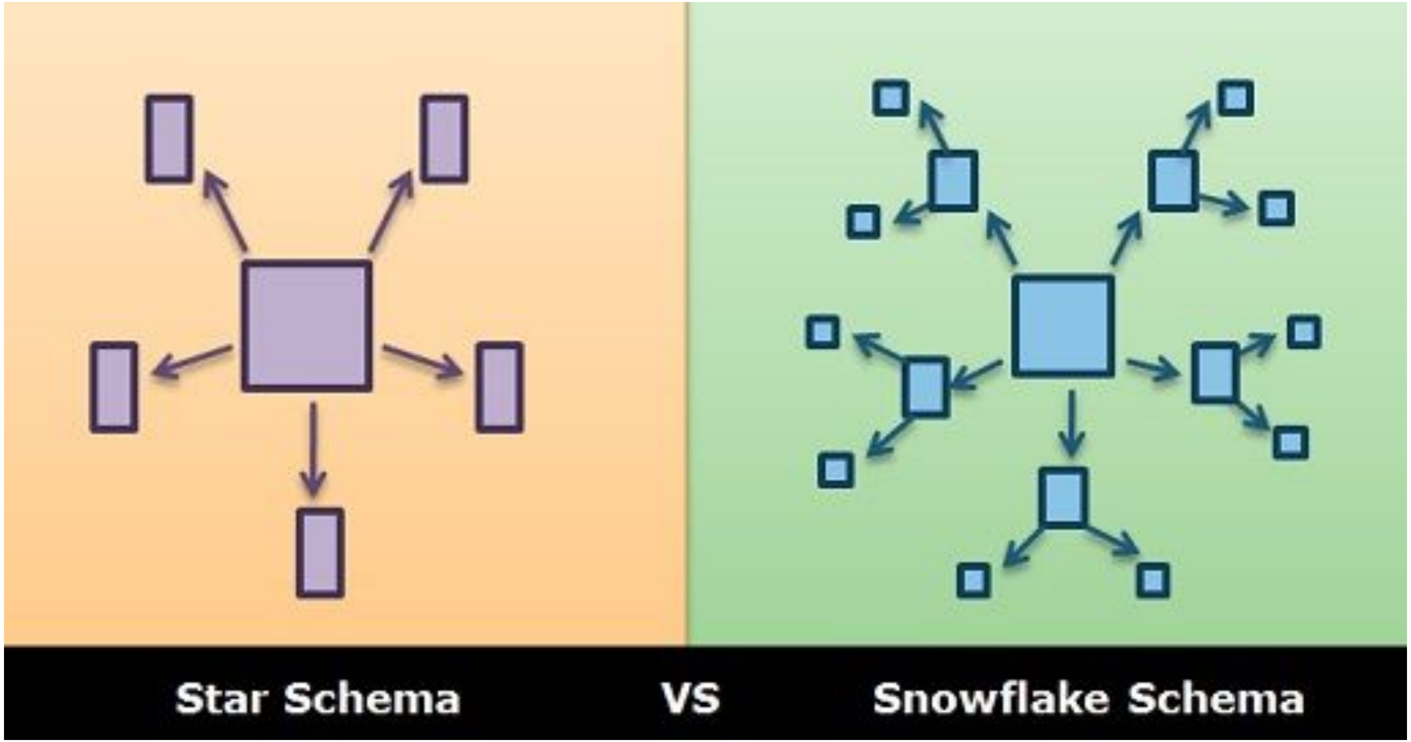
Three-Tier Architecture for a data warehouse system

source: https://www.javatpoint.com/data-warehouse-architecture

Data Warehouse Schema Design

- OLTP vs. OLAP
- Bulk insert - no update - frequent and complex queries
- Data warehouses often use denormalized or partially denormalized schemas to optimize query and analytical performance.
- Dimensional modeling approach
  - Dimensions - provide the “who, what, where, when, why, and how” context surrounding a business process event; descriptive context
  - Facts - the measurements that result from a business process event and are almost always numeric

Data Warehouse Schema Design



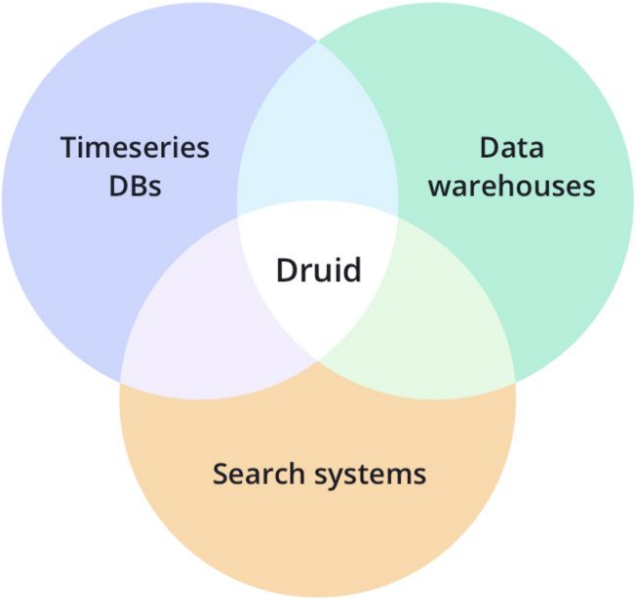
source: https://techdifferences.com/difference-between-star-and-snowflake-schema.html

Traditional Data Warehouse Systems

- Relational databases - primarily data source
- Structured data
- Relational database - choice for DW database
- Methodology matured in 90s
  - The Data Warehouse Toolkit by Ralph Kimball (John Wiley and Sons, 1996)
  - Building the Data Warehouse by William Inmon (John Wiley and Sons, 1996)
- Indexing - optimizing query execution
- Materialized views
  - used to precompute and store aggregated data such as the sum of sales
  - eliminates the overhead associated with expensive joins and aggregations for a large or important class of queries

Apache Druid - Introduction

- *Open source distributed data store*
- Combines ideas from data warehouses, time-series databases, and search systems
- High performance real-time analytics database for a broad range of use cases



Apache Druid - Key Features

- Column-oriented storage
- Native search indexes
- Streaming and batch ingest
- Flexible schemas
- Time-optimized partitioning
- SQL support
- Horizontal scalability
- Easy operation

- Initial requirements
  - Arbitrary queries
  - Scalability: trillions events/day
  - Interactive: low latency queries
  - Real-time: data freshness
  - High availability
  - Rolling upgrades
- Initial motivation
  - Business intelligence queries
  - Arbitrary slicing and dicing of data
  - Interactive real-time visualizations on complex data streams

Apache Druid - Common Use Cases

- Clickstream analytics (web and mobile analytics)
- Risk/fraud analysis
- Network telemetry analytics (network performance monitoring)
- Server metrics storage
- Supply chain analytics (manufacturing metrics)
- Application performance metrics
- **Business intelligence / OLAP**

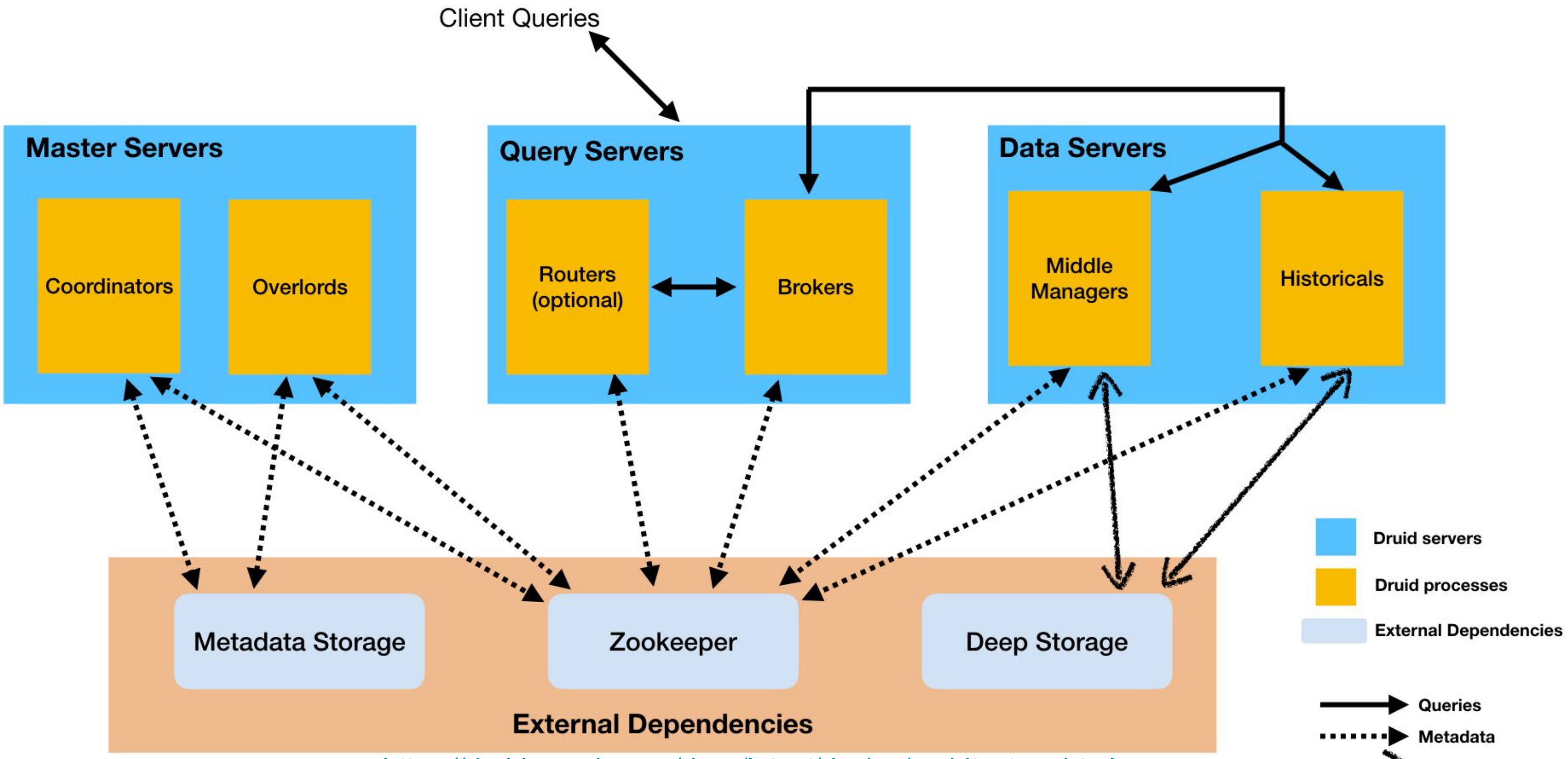
Apache Druid - Process Types

- *Druid architecture* - multi-process, distributed, cloud-friendly
- **Coordinator** - manage data availability on the cluster.
- **Overlord** - control the assignment of data ingestion workloads.
- **Broker** - handle queries from external clients.
- **Router** - are optional processes that can route requests to Brokers, Coordinators, and Overlords.
- **Historical** - store queryable data.
- **MiddleManager** - responsible for ingesting data.

Apache Druid - Server Types

- It is suggested organizing processes into server types
- **Master**
  - runs *Coordinator* and *Overlord* processes
  - manages data availability and ingestion
- **Query**
  - runs *Broker* and optional *Router* processes
  - handles queries from external clients
- **Data**
  - runs *Historical* and *MiddleManager* processes
  - executes ingestion workloads and stores all queryable data

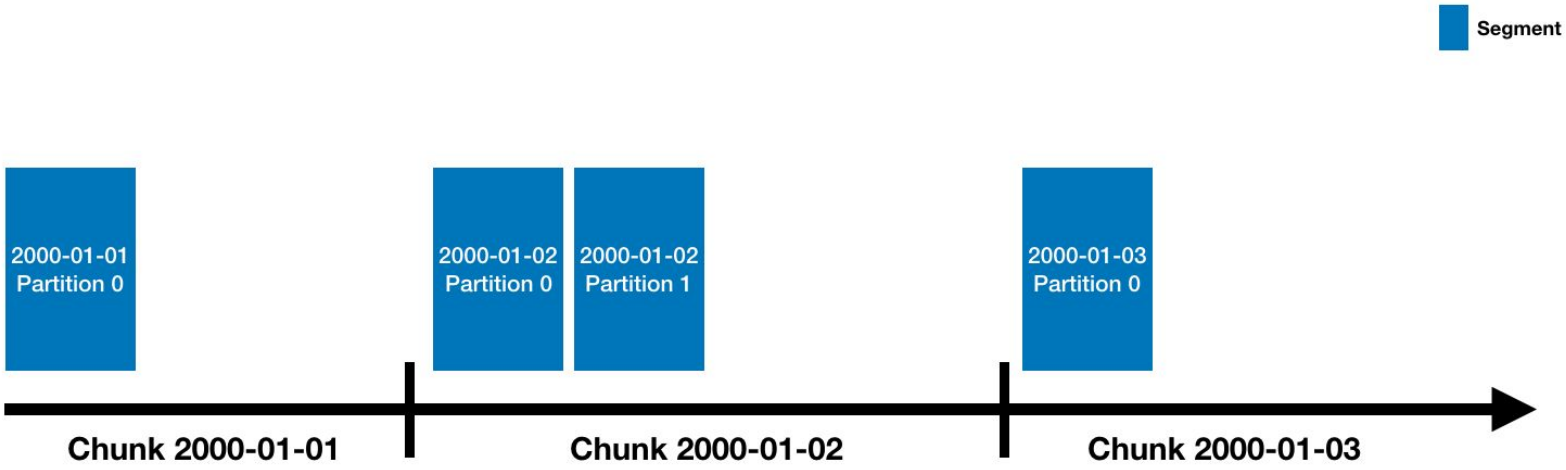
Apache Druid - Architecture



source: https://druid.apache.org/docs/latest/design/architecture.html

Apache Druid - Storage design

- Data is stored in ***datasources***
- Datasources are partitioned by time in ***chunks***
- Chunk contains one of more ***segments***
- Segment is a single file



source: https://druid.apache.org/docs/latest/design/architecture.html

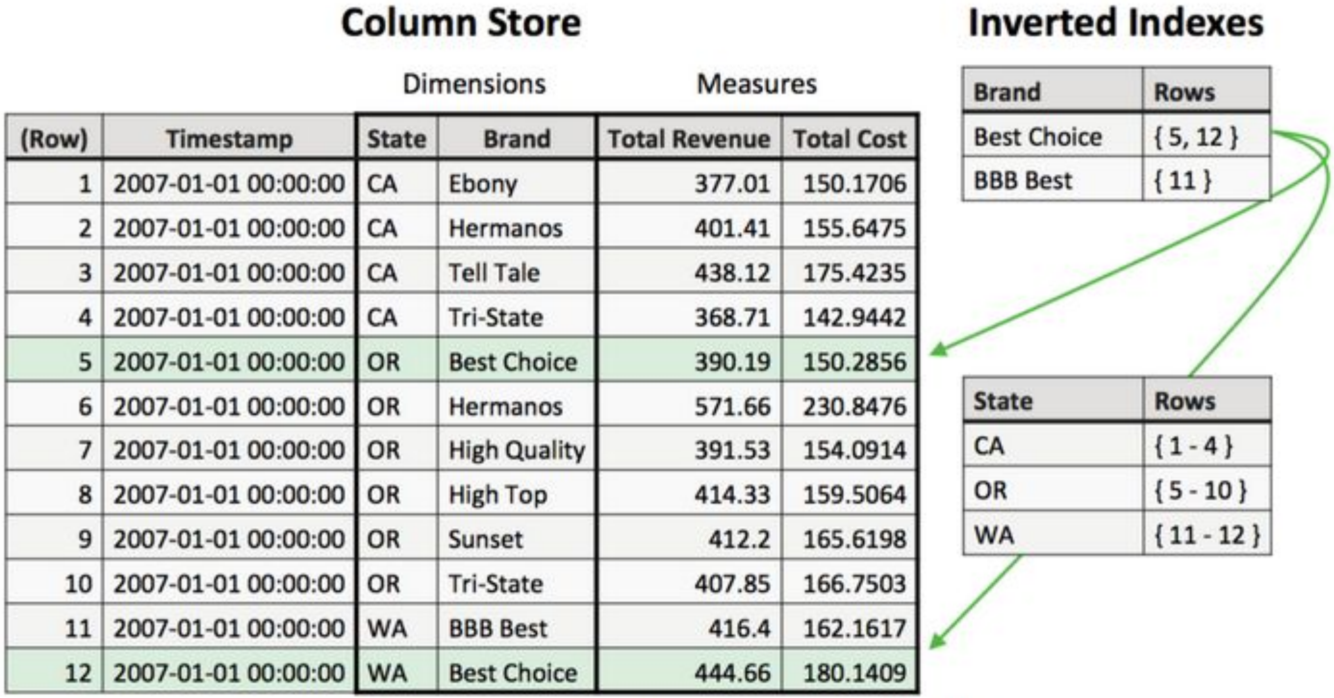
Apache Druid - Ingestion

- Loading data in Druid is called **ingestion** or **indexing**
  - consists of reading data from a source system and creating segments based on that data
- Streaming ingestion methods - supervisor types
  - Kafka
  - Kinesis
  - Tranquility
- Batch ingestion methods - task types
  - Native batch - simple
  - Native batch - parallel
  - Hadoop

Apache Druid - Ingestion

- *Primary timestamp* - used for partitioning and sorting data
- *Dimensions* - columns stored as-is and can be used for any purpose
- *Metrics* - columns stored in an aggregated form
- *Rollup* - form of summarization or pre-aggregation used to minimize the amount of raw data that needs to be stored
- *Partitioning* - secondary partitioning using a particular dimension will improve locality
- **Ingestion specs** - JSON file consists of three main parts
  - **dataSchema** - definition of datasource, primary timestamp, dimensions, metrics, transformation and filtering
  - **ioConfig** - description of source system, connection method and data parsing
  - **tuningConfig** - controls various tuning parameters specific to each ingestion method

Apache Druid - Datasource



Column Store

		Dimensions		Measures	
(Row)	Timestamp	State	Brand	Total Revenue	Total Cost
1	2007-01-01 00:00:00	CA	Ebony	377.01	150.1706
2	2007-01-01 00:00:00	CA	Hermanos	401.41	155.6475
3	2007-01-01 00:00:00	CA	Tell Tale	438.12	175.4235
4	2007-01-01 00:00:00	CA	Tri-State	368.71	142.9442
5	2007-01-01 00:00:00	OR	Best Choice	390.19	150.2856
6	2007-01-01 00:00:00	OR	Hermanos	571.66	230.8476
7	2007-01-01 00:00:00	OR	High Quality	391.53	154.0914
8	2007-01-01 00:00:00	OR	High Top	414.33	159.5064
9	2007-01-01 00:00:00	OR	Sunset	412.2	165.6198
10	2007-01-01 00:00:00	OR	Tri-State	407.85	166.7503
11	2007-01-01 00:00:00	WA	BBB Best	416.4	162.1617
12	2007-01-01 00:00:00	WA	Best Choice	444.66	180.1409

Inverted Indexes

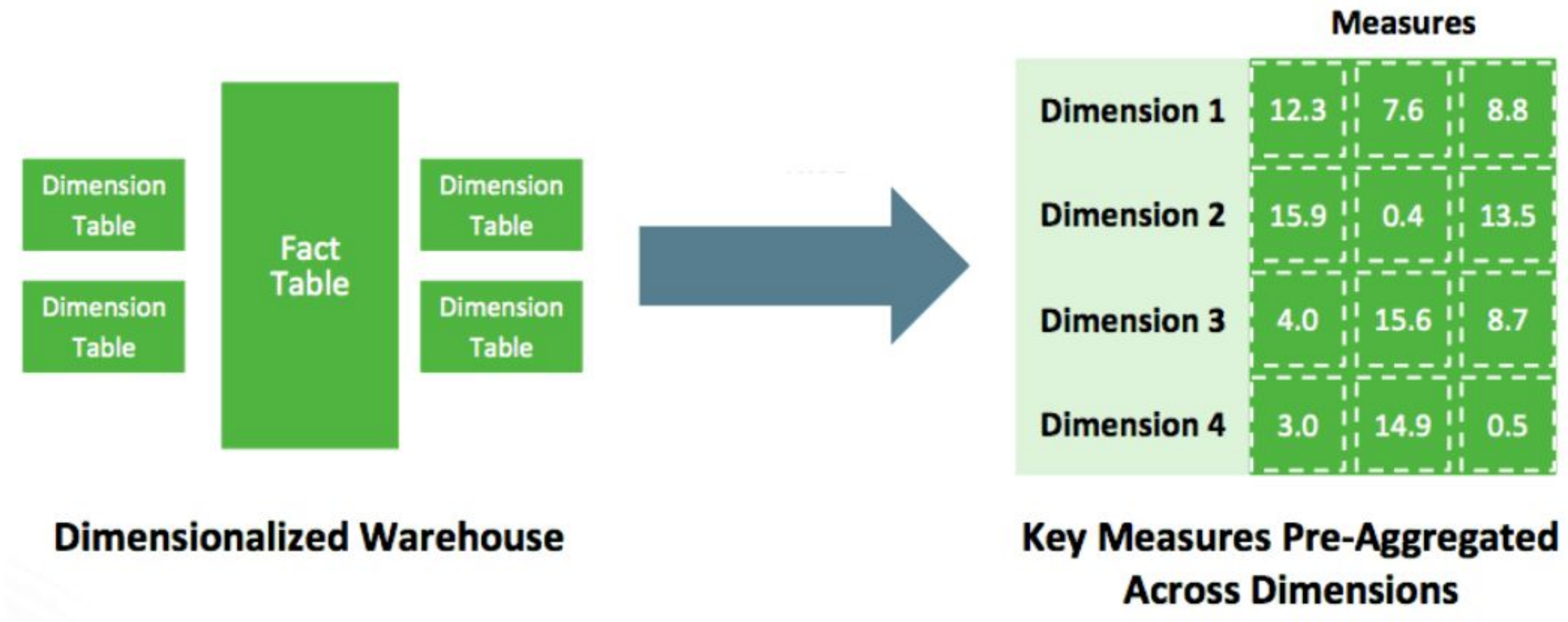
Brand	Rows
Best Choice	{ 5, 12 }
BBB Best	{ 11 }

State	Rows
CA	{ 1 - 4 }
OR	{ 5 - 10 }
WA	{ 11 - 12 }

Figure 1: Druid combines the best qualities of a column store and inverted indexing

source: https://blog.cloudera.com/apache-hive-druid-part-1-3/

Dimesionalized DW vs Druid datastore



Dimensionalized Warehouse

	Measures		
Dimension 1	12.3	7.6	8.8
Dimension 2	15.9	0.4	13.5
Dimension 3	4.0	15.6	8.7
Dimension 4	3.0	14.9	0.5

Key Measures Pre-Aggregated Across Dimensions

source: https://blog.cloudera.com/apache-hive-druid-part-1-3/

Apache Druid - Standalone Installation

- Prereqs
  - Linux, Mac OS X, or other Unix-like OS (Windows is not supported)
  - Java 8, Update 92 or later (8u92+)
- Install Druid
  - Download binaries
  - Extract Druid
- Start up Druid services
  - Single machine configuration - `./bin/start-micro-quickstart`
- Open the Druid console
  - Visit http://localhost:8888

Apache Druid - Data Loader (1/2)

- Click **Load data** from the Druid console header
- Select the **Local disk** tile and then click **Connect data**.
- Enter the following values:
  - **Base directory**: quickstart/tutorial/
  - **File filter**: wikiticker-2015-09-12-sampled.json.gz
- Click **Apply**.
- Click **Next: Parse data**.
- With the JSON parser selected, click **Next: Parse time**.
- Click **Next: Transform**, **Next: Filter**, and then **Next: Configure schema**, skipping a few steps. Disable **Rollup**.

Apache Druid - Data Loader (2/2)

- Click **Next: Partition** to configure how the data will be split into segments. In this case, choose **DAY** as the **Segment granularity**.
- Click **Next: Tune** and **Next: Publish**.
- Let's change the default name from *wikiticker-2015-09-12-sampled* to *wikipedia*.
- Click **Next: Edit spec** to review the ingestion spec we've constructed with the data loader.
- Once you are satisfied with the spec, click **Submit**.
- Open **Ingestion** section.
- Once a task is completed, open **Datasources** section.
- Finally, go to **Query** section.

Apache Druid - Querying

- Apache Druid supports two query languages:
  - Druid SQL - powered by a parser and planner based on Apache Calcite
  - native queries - JSON-based query language
- Query examples for *wikipedia* datasource
  - `SELECT count(*) FROM wikipedia WHERE isAnonymous = TRUE`
  - `SELECT count(*) FROM wikipedia WHERE "comment" LIKE '%clean%'`

  -
    ```
    SELECT page, cityName, countryName, isAnonymous
    FROM "wikipedia" WHERE channel LIKE '%sr.%'
    ```

  -
    ```
    SELECT "countryName", count(*) "cnt" FROM "wikipedia"
    GROUP BY "countryName" ORDER BY "cnt" DESC LIMIT 10
    ```

  -
    ```
    SELECT "user", count(*) "cnt" FROM "wikipedia"
    WHERE "isRobot" = FALSE GROUP BY "user"
    ORDER BY "cnt" DESC LIMIT 10
    ```
- Druid console includes query editor with query building options

Apache Druid - Docker Compose

- Prereqs
  - Docker & Docker Compose
  - Download or clone https://github.com/vladimirivkovic/druid-tutorial
- Compose file - *docker-compose.yml*
  - a container for each Druid service
  - Zookeeper service
  - PostgreSQL container as the metadata store and OLTP database
  - Metabase as a graphical UI tool
- Configuration
  - Using environment variables, e.g. Druid extensions
  - Segments stored in a shared directory
- Launching the cluster
  - `docker-compose up`

Apache Druid - File Ingestion

- Sample ingestion specs for local file
  - On each container /quickstart/tutorial directory
- Several ways to submit ingestion task
  - Loading data with a spec (via console)
    - Load Data > Edit Spec > Submit
  - Loading data with a spec (via command line)

    -
      ```
      bin/post-index-task --file quickstart/tutorial/wikipedia-index.json
      --url http://localhost:8081
      ```
  - Loading data without the script

    -
      ```
      curl -X 'POST' -H 'Content-Type:application/json' -d
      @quickstart/tutorial/wikipedia-index.json
      http://localhost:8081/druid/indexer/v1/task
      ```

Ingestion from Source Database 1/4

- Batch ingestion using *parallel_index* task type
- Sample OLTP database - AdventureWorks
  - Official Microsoft sample database
  - Several schemas: purchasing, sales, person, production
  - Sample DW diagrams
- Source database originally stored in *postgre* container
  - Run *load-data.sh* in order to prepare source relational database
- Ingestion task uses JDBC to retrieve data from database
  - *inputSource* type - sql
  - *connectorConfig* with connection string

Ingestion from Source Database 2/4

- Source tables from *production* schema
  - products, (sub)categories, work orders, work order routings
- Dimensions and metrics
  - Timestamp - e.g. start date of work order
  - Dimensions - product ID, product category
  - Metrics - ordered quantity, scraped quantity
  - Ingestion specs examples with and without metrics
- Granularity
  - Segment granularity in months
  - Query granularity in days
- Running ingestion task
  -
    ```
    curl -X 'POST' -H 'Content-Type:application/json' -d
    @spec/spec-wo.json 'http://localhost:8888/druid/indexer/v1/task'
    ```

Ingestion from Source Database 3/4

- Ingestion specs for
  - Simple work order

```
SELECT startDate, workOrderId, productId, orderQty
FROM production.workorder
```

  - Extended work order with product categories

```
SELECT workorderid, startdate, orderqty, w.productid, p.name as productname,
       psc.name as subcategory, pc.name as category
FROM production.workorder w
     JOIN production.product p ON w.productid = p.productid
     JOIN production.productsubcategory psc ON p.productsubcategoryid =
sc.productsubcategyid
     JOIN production.productcategory pc ON psc.productcategoryid = pc.productcategoryid
```

  - Work order routings

```
SELECT wo.workOrderId AS workOrderId, p.name AS productName, l.name AS locationName,
       wo.orderQty, operationSequence, plannedCost, actualCost, actualStartDate
FROM production.workOrderRouting wor
     JOIN production.location l ON wor.locationId = l.locationId
     JOIN production.product p ON wor.productId = p.productId
     JOIN production.workOrder wo ON wor.workOrderId = wo.workOrderId
```

Ingestion from Source Database 4/4

- Sample Druid SQL queries
- Work order
  - Work orders with largest quantity
  - Top 20 most ordered products
  - Products with multiple daily orders
  - Product with most *scrappedqty*
  - Weekly scrap rate
- Extended work order with categories
  - Top 10 most ordered bikes
  - Total ordered quantity per month
  - Most ordered subcategories
  - Subcategories with largest scrap rate
- Work order routing
  - Work orders with most routings
  - Locations sorted by routings
  - Painted products
  - Products with most resource hours

Querying and Analytics

- Druid SQL
  - Filtering, grouping, ordering
  - Joining datasources
    - SQL join
    - lookup
- Native queries - JSON based
  - Aggregation queries - TopN, GroupBy, Timeseries
  - Metadata queries
  - Other queries - Scan, Search

```
{
  "queryType": "topN",
  "dataSource": "sample_data",
  "dimension": "sample_dim",
  "threshold": 5,
  "metric": "count",
  "granularity": "all",
  "filter": {...},
  "aggregations": [...],
  "postAggregations": [...],
  "intervals": [...]
}
```

Ingestion - Individual exercises

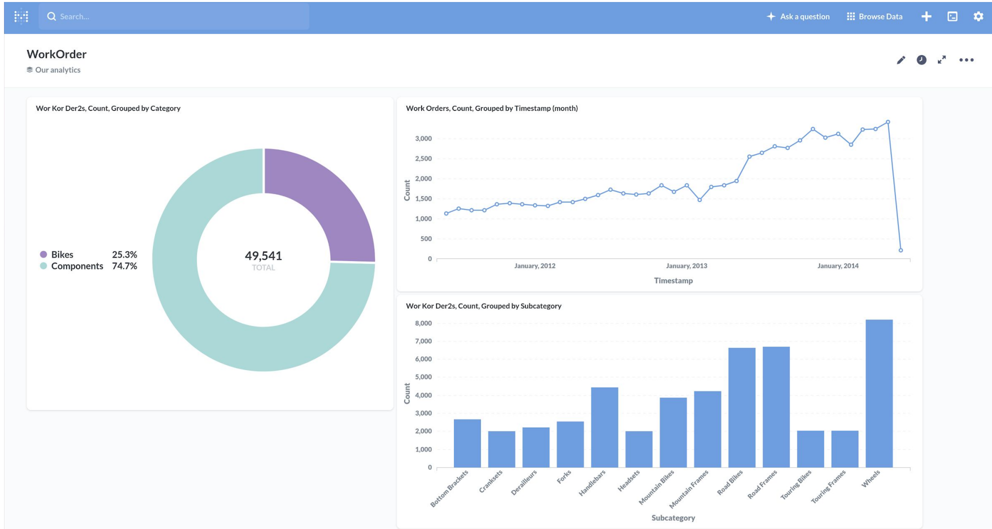
- Join multiple table from *sales* schema
  - SalesOrderDetail, SalesOrderHeader, Customer, SalesTerritory, SalesPerson, Product
- Define ingestion specs with metrics
  - Quantity, total amount, discount amount, tax amount, freight
- Submit ingestion task
- Monitor task execution
- Run some SQL queries on new datasource
  - Most popular products by territory
  - Customers by total amount
- Remove datasource

Apache Druid - Third Party Clients

- Druid’s community developed several client libraries
  - Python, R, JavaScript, Clojure, Elixir, Ruby, SQL, PHP, Scala, Java, .NET, Rust
- Graphical UIs
  - Superset
  - Grafana
  - Pivot
  - Metabase
  - Metatron
- Other distributions, tools, and extensions

Metabase with Druid

- Web-based data analytics tool
- Connection to Druid broker
- Automatic retrieval of metadata
- User friendly
  - Asking questions using simple interface, complex custom queries, or native queries
  - Custom interactive dashboards with multiple graphs and diagrams based on questions
- Supports only Druid native queries



Metabase

Metabase with Druid

- Visit http://localhost:3000
- Create a new dashboard
- Ask question
- Add question to dashboard
- Place a graph/diagram on a dashboard
- Repeat the same steps for several questions

What’s next?

- Streaming ingestion
  - Polling data from web (news, wikipedia, livescores)
  - Sending to Kafka
  - Druid ingests data stream from Kafka
  - Real-time analytics in Druid
- Batch ingestion
  - Hadoop data
- Complex ingestion
  - Transformations and filtering
- Druid Community events

Thank You!