Apache Ignite as MPP Accelerator

Alexander Ermakov, CTO

ARENADATA

Agenda

- About us
- Why do traditional DWH needs in-memory grid?
- Real Time Analytics for Telco Cases
- Integrating Apache Ignite with Arenadata DB
- Using the power of in-memory computing with MPP (Example)

<About us>

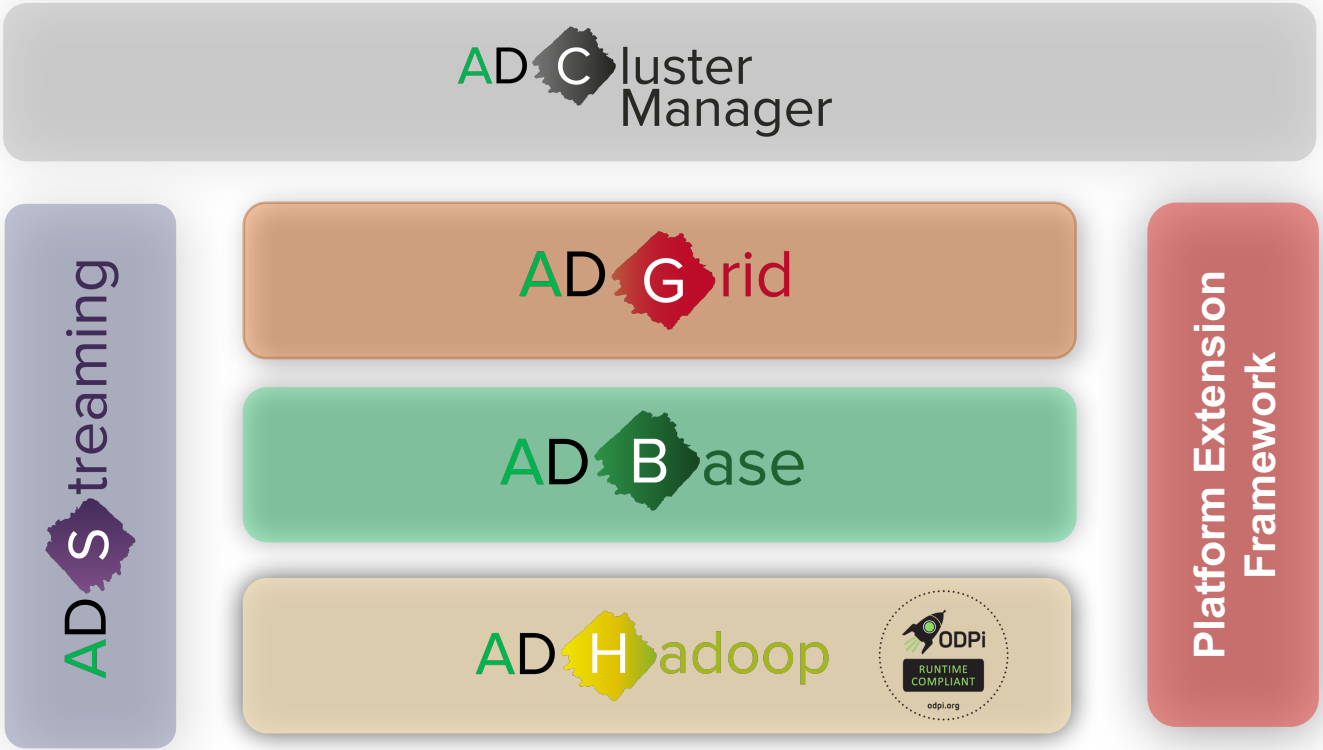
ARENADATA

Who we are?

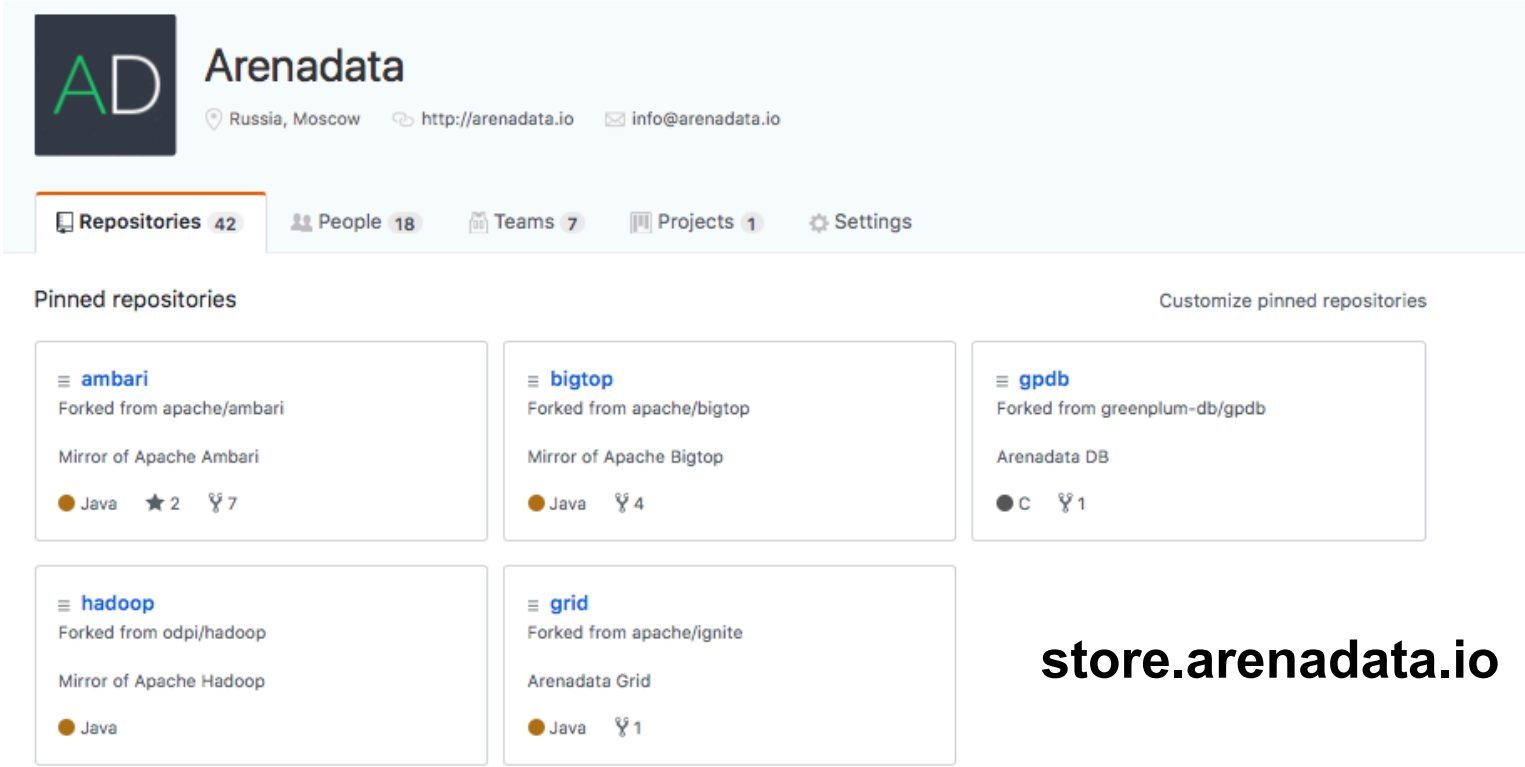
- Arenadata unites a keen team of developers & engineers working on building enterprise data platform.
- We are contributors of Open Source Projects:
  - Greenplum
  - Apache PXF
  - Apache Bigtop
- Members of ODPi (Linux Foundation) since 2015

ODPi Compliant Platforms

Arenadata Enterprise Data Platform



Arenadata - Open Source



store.arenadata.io

ARENADATA

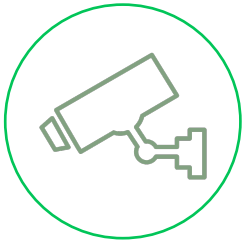
Our Partners

ARENADATA

Why DWH needs in-memory grid?

New Generation of Business Cases

Mobile Sensors



Video Surveillance

FACEBOOK UPLOADS
250 MILLION
PHOTOS EACH DAY

Social Media

READING SMART METERS
EVERY 15 MINUTES IS
3000X MORE
DATA INTENSIVE

Smart Grids

Medical Imaging

OIL RIGS GENERATE
25000
DATA POINTS PER
SECOND

Oil Exploration

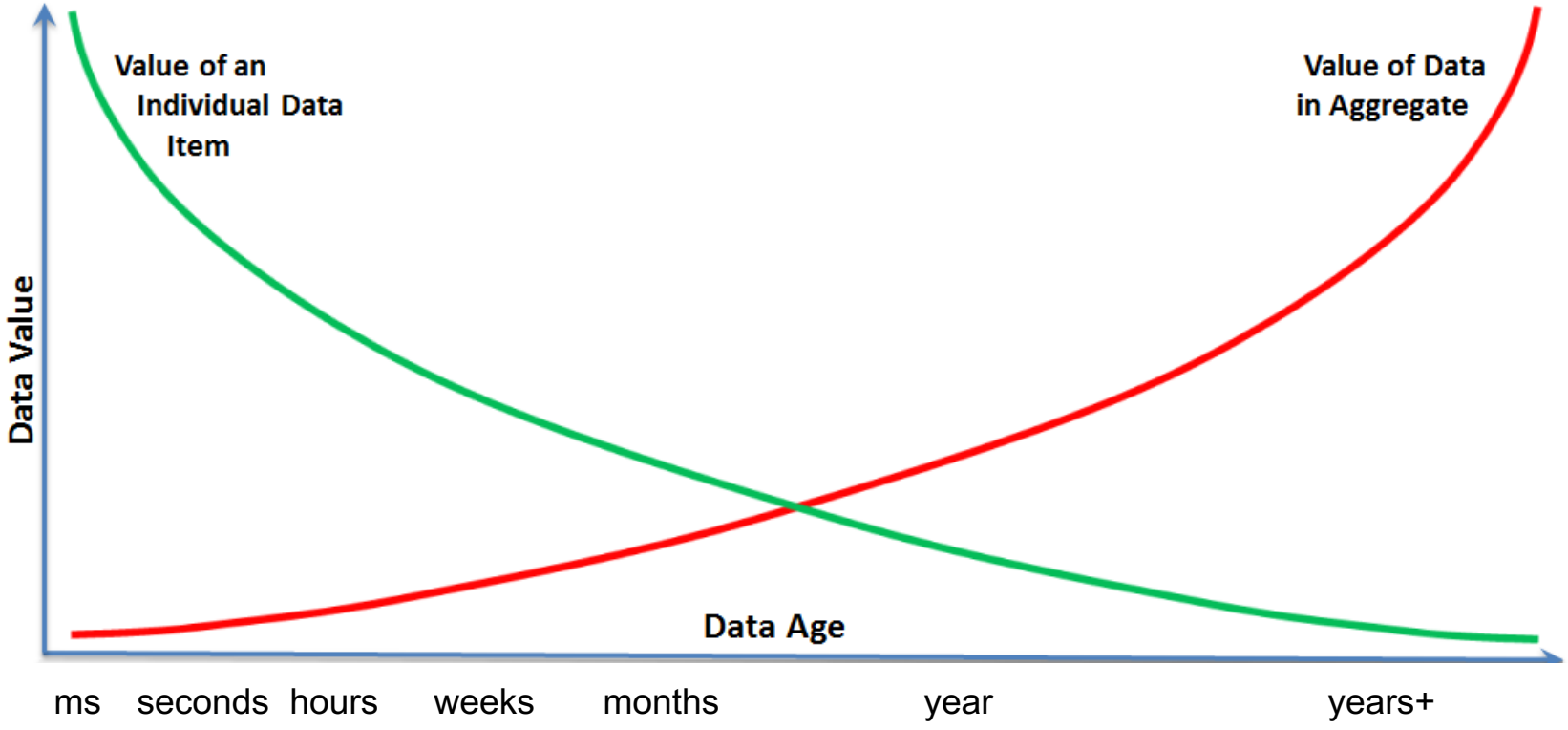
Stock Market

COST TO SEQUENCE
ONE GENOME
HAS FALLEN FROM $100M
IN 2001
TO $10K IN 2011
TO $1K IN 2014

Gene Sequencing

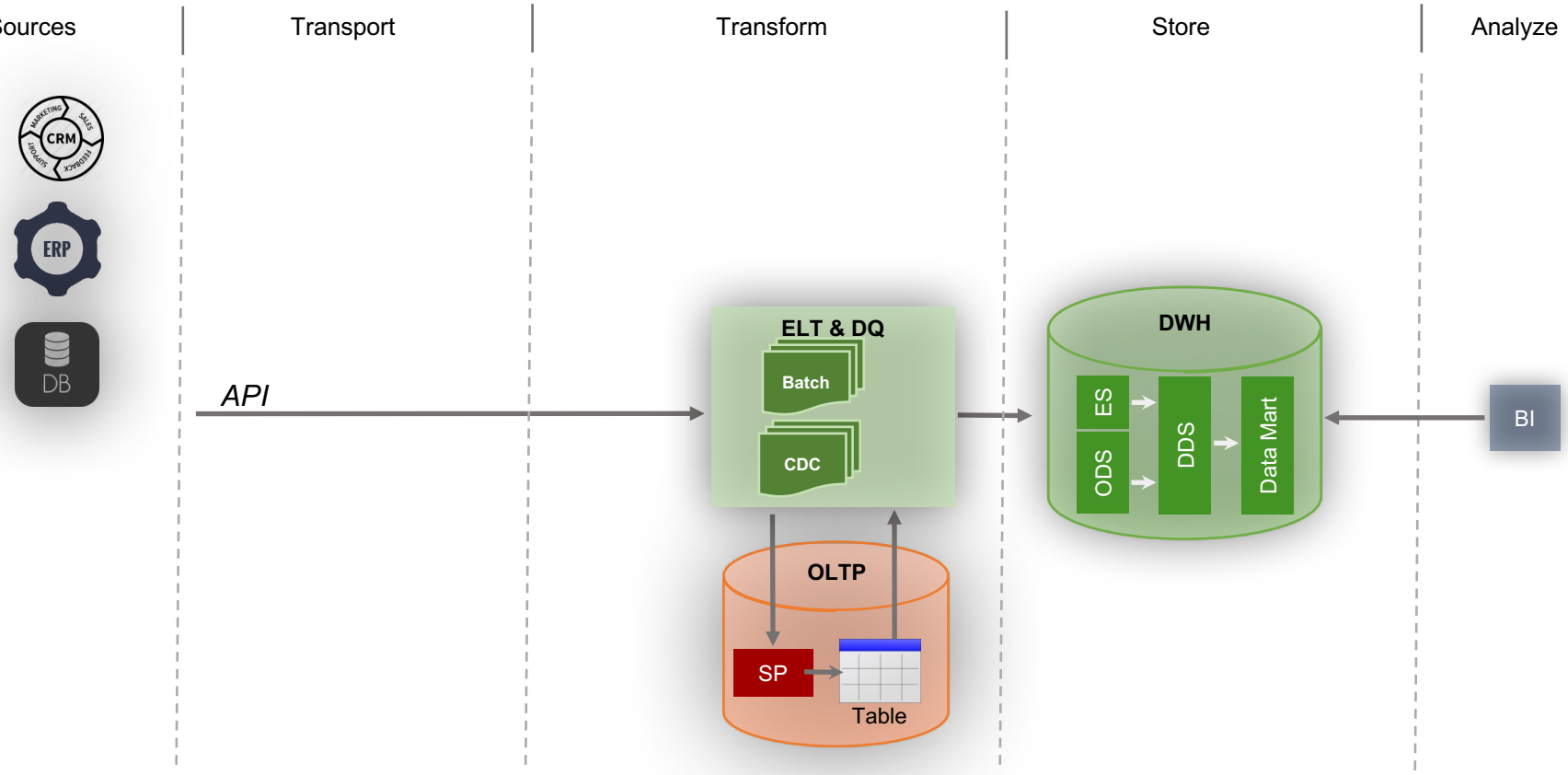
ARENADATA

Data Value Chain



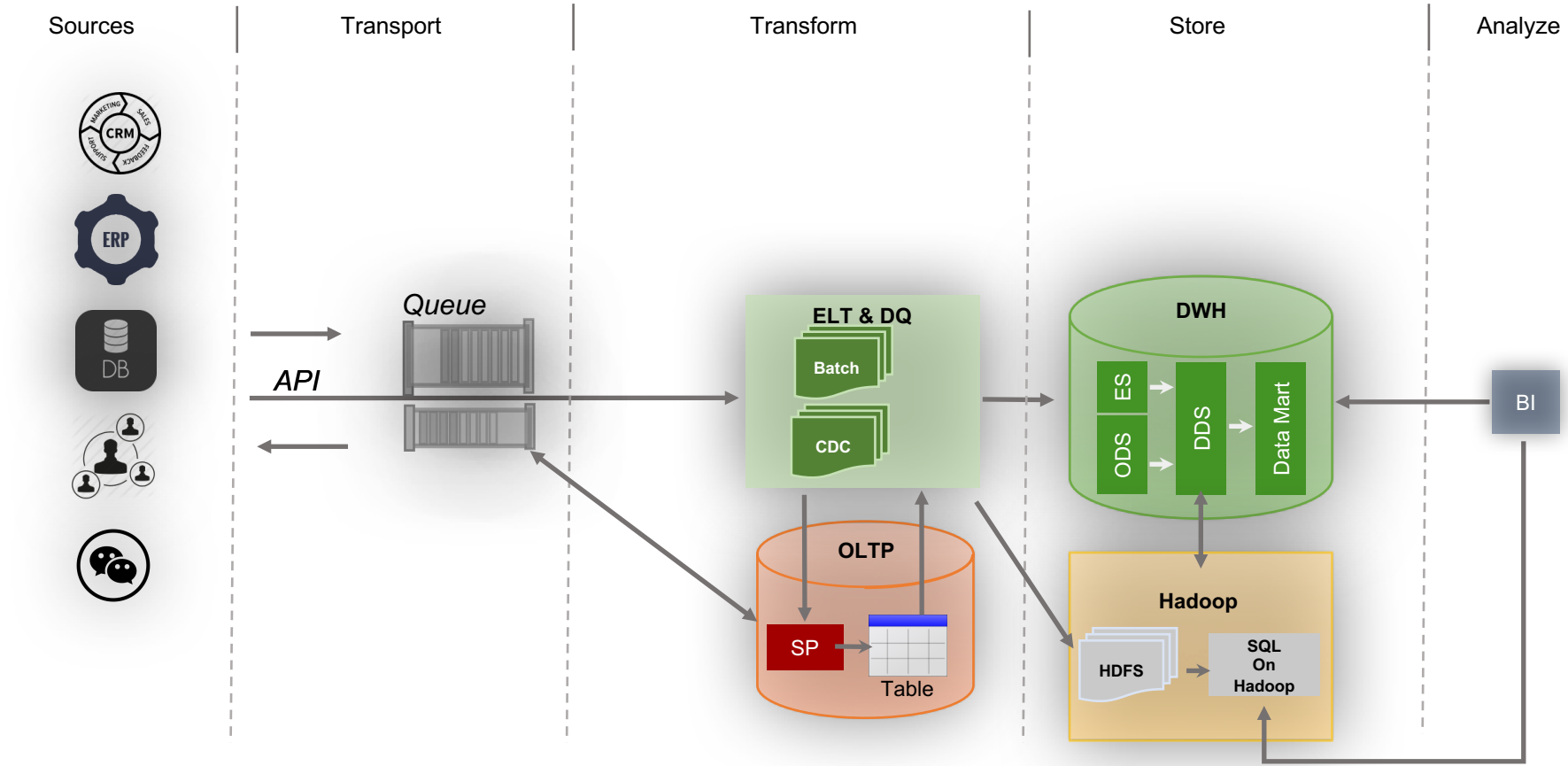
ARENADATA

Data Warehouse



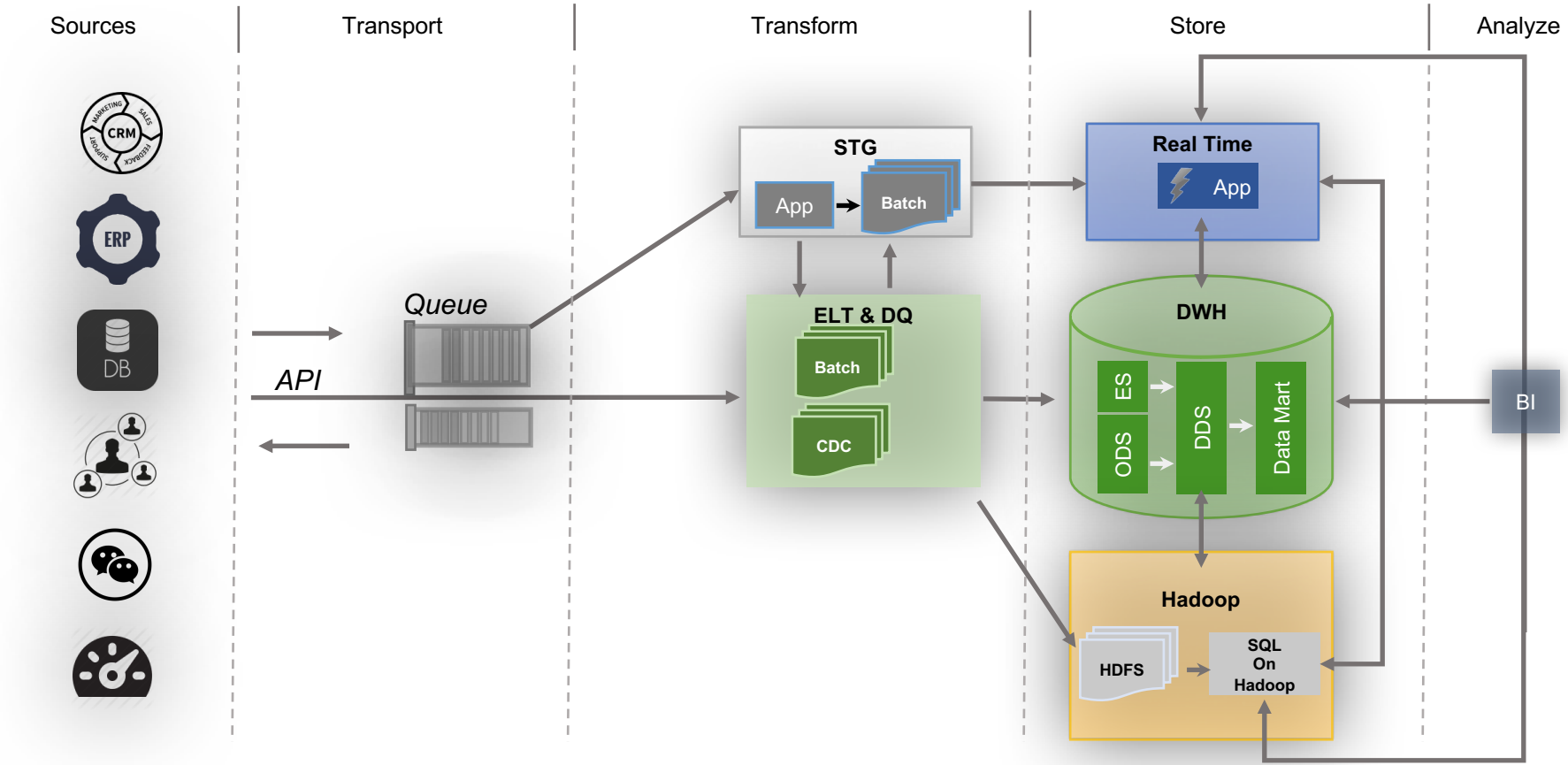
ARENADATA

Data Lake



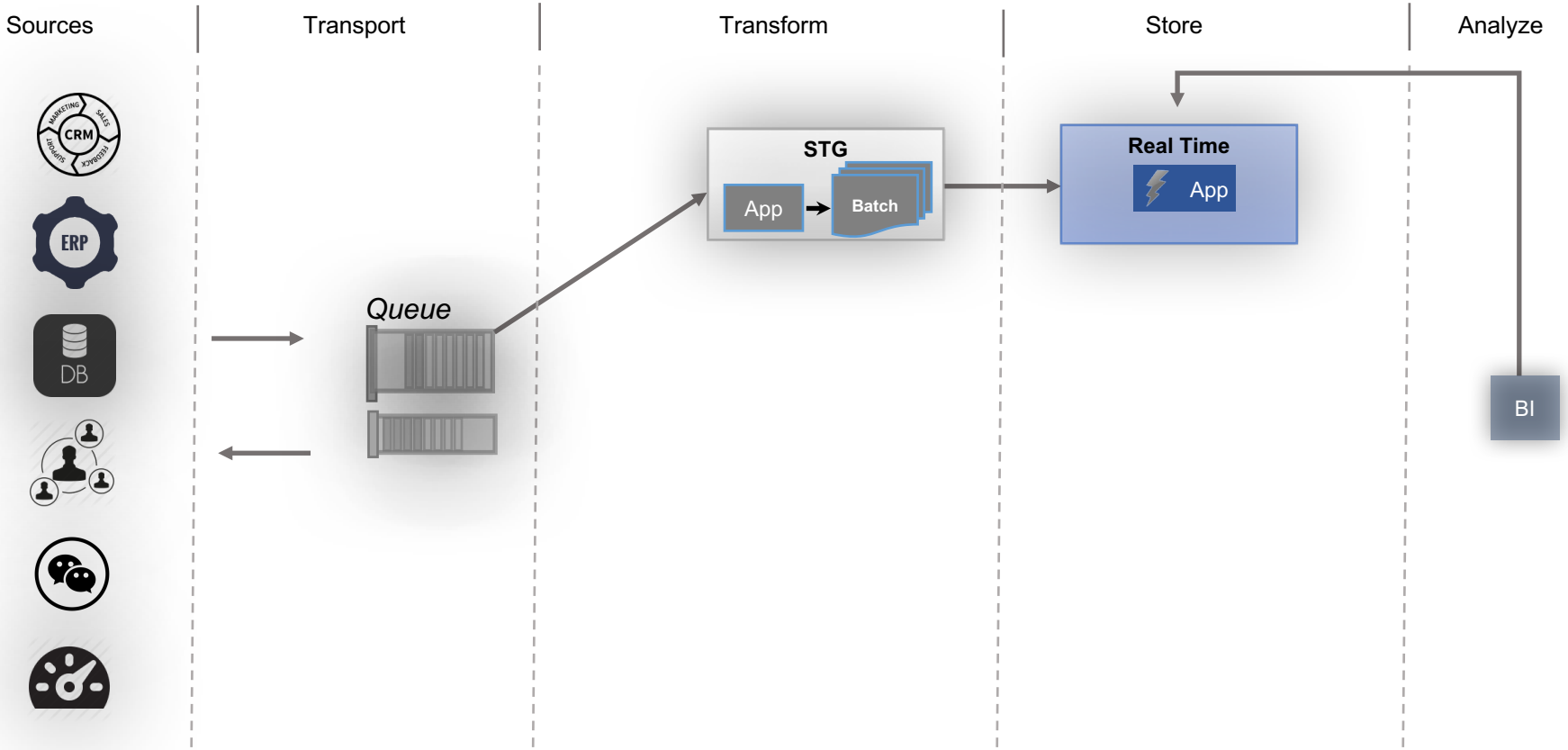
ARENADATA

Lambda Architecture



ARENADATA

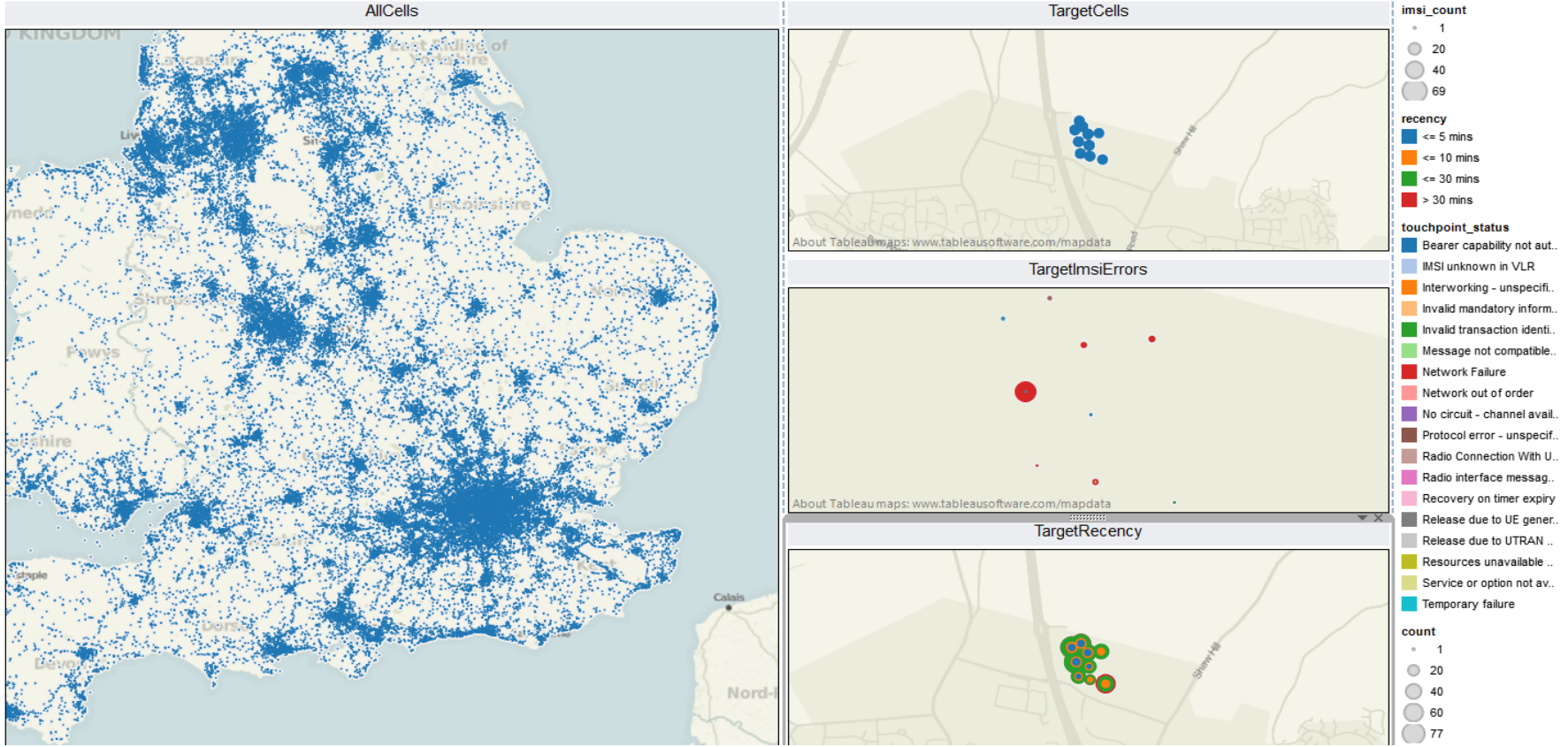
Kappa Architecture



Real Time Analytics for Telco Cases

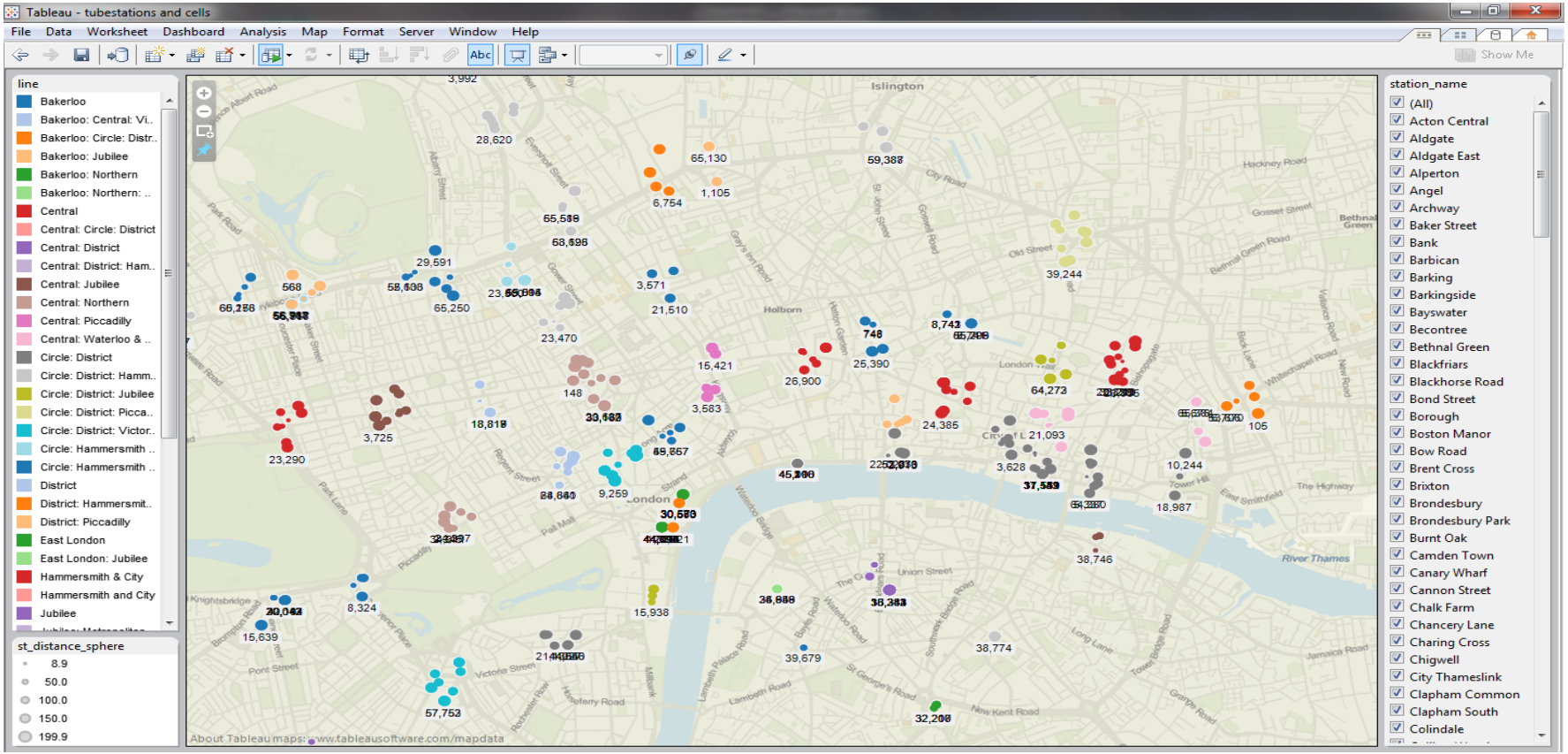
ARENADATA

Customer Retention / Connection Breakdowns



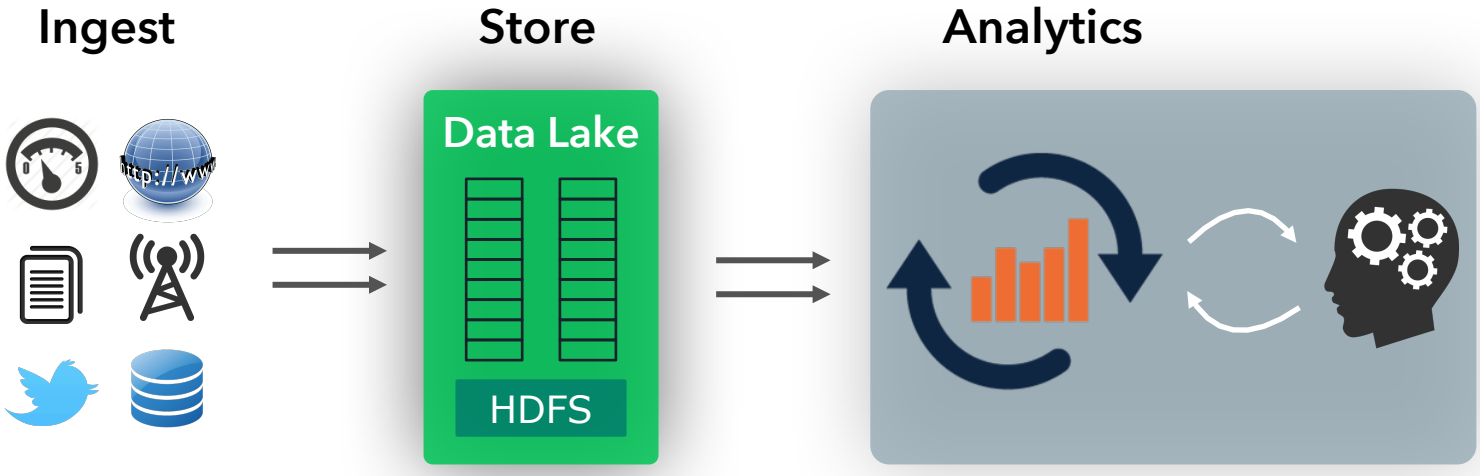
ARENADATA

Geo Marketing



ARENADATA

Migrating from a Reactive, Static and Constrained Model…

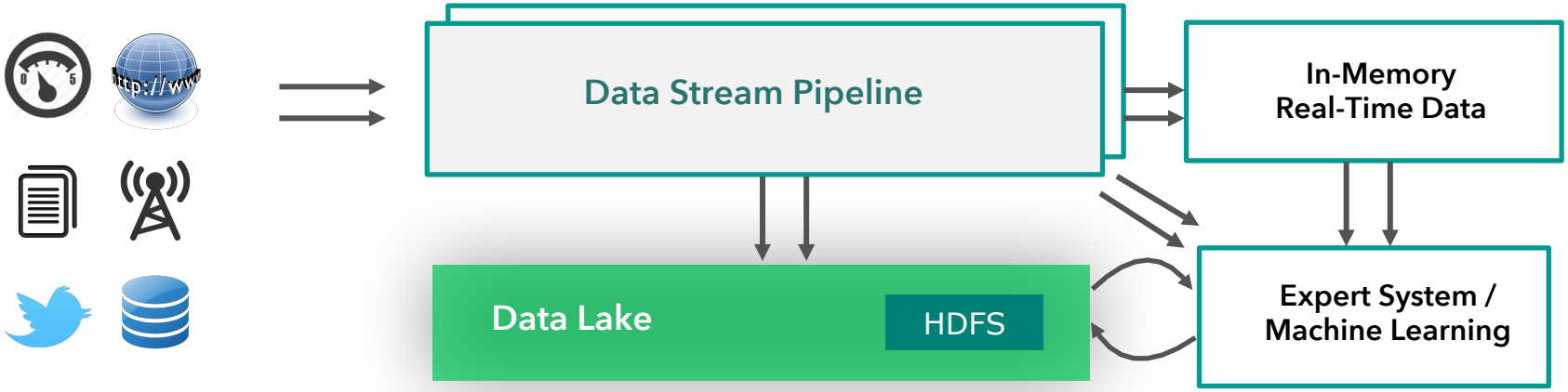


Coding based
No real-time information
Based on expensive ETL

Hard to change
Labor intensive
Inefficient

ARENADATA

To Pro-Active, Self-Improving, Machine Learning Systems

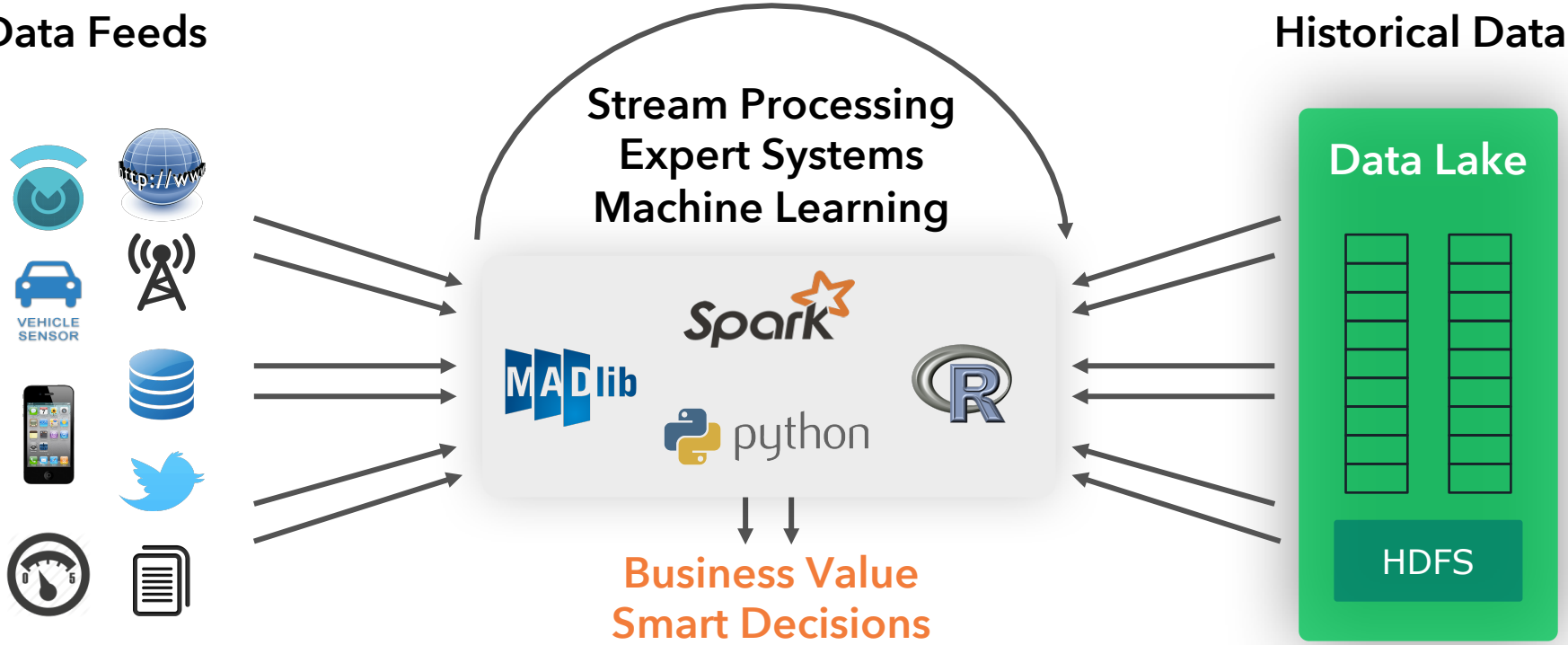


Multiple Data Sources
Real-Time Processing
Store Everything

Continuous Learning
Continuous Improvement
Continuous Adapting

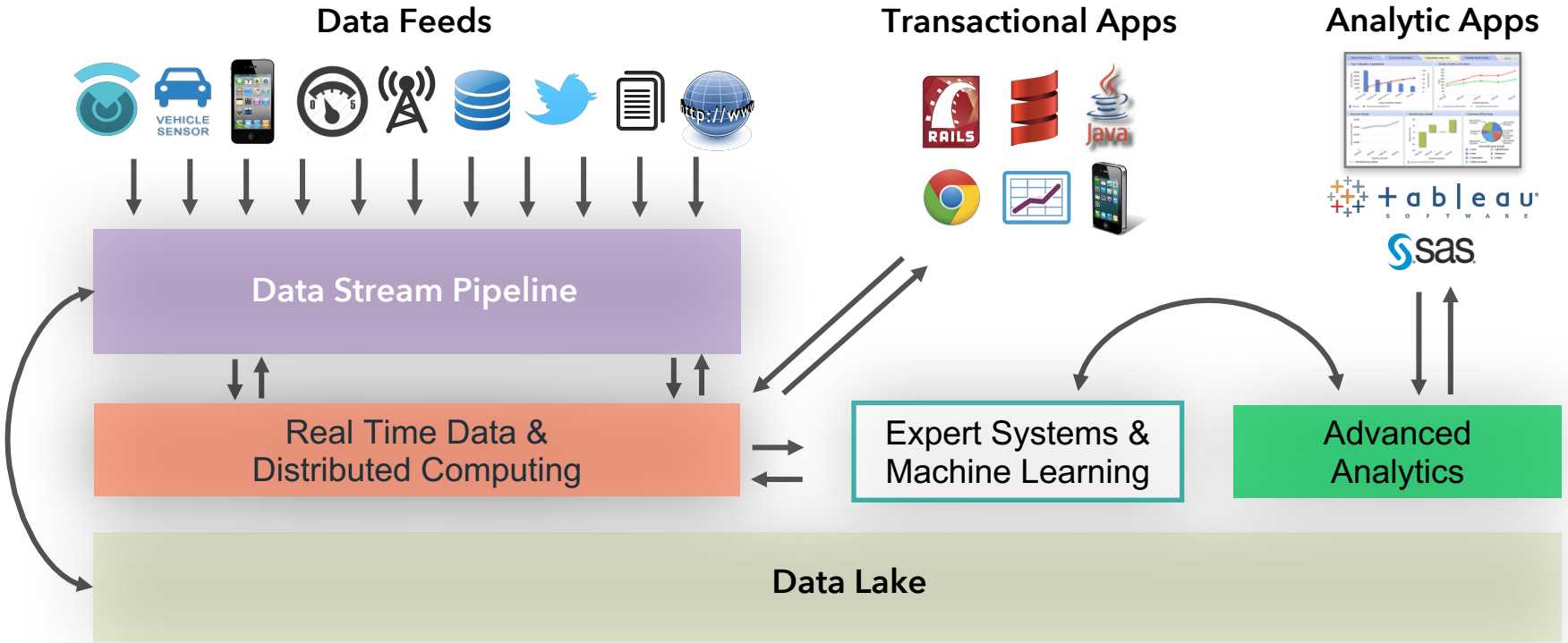
ARENADATA

Sandboxes

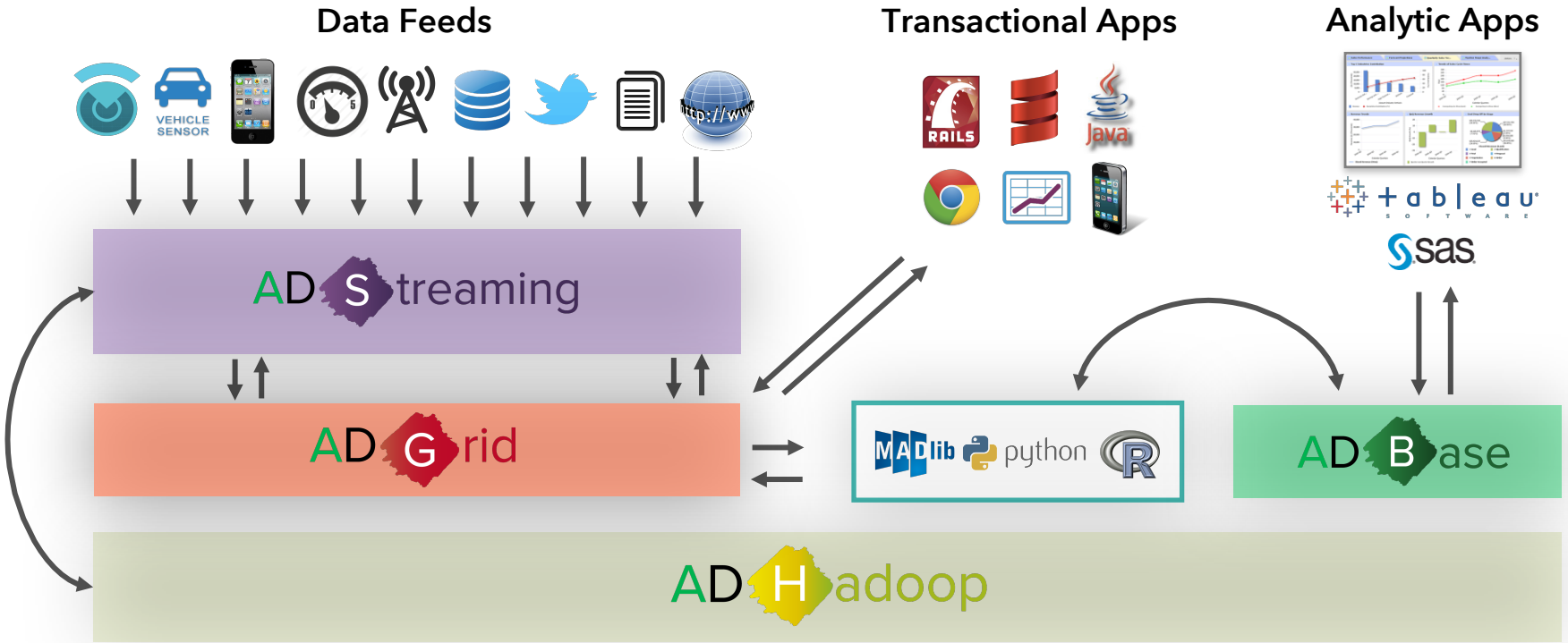


ARENADATA

Data Streaming Reference Architecture



Data Streaming Reference Architecture

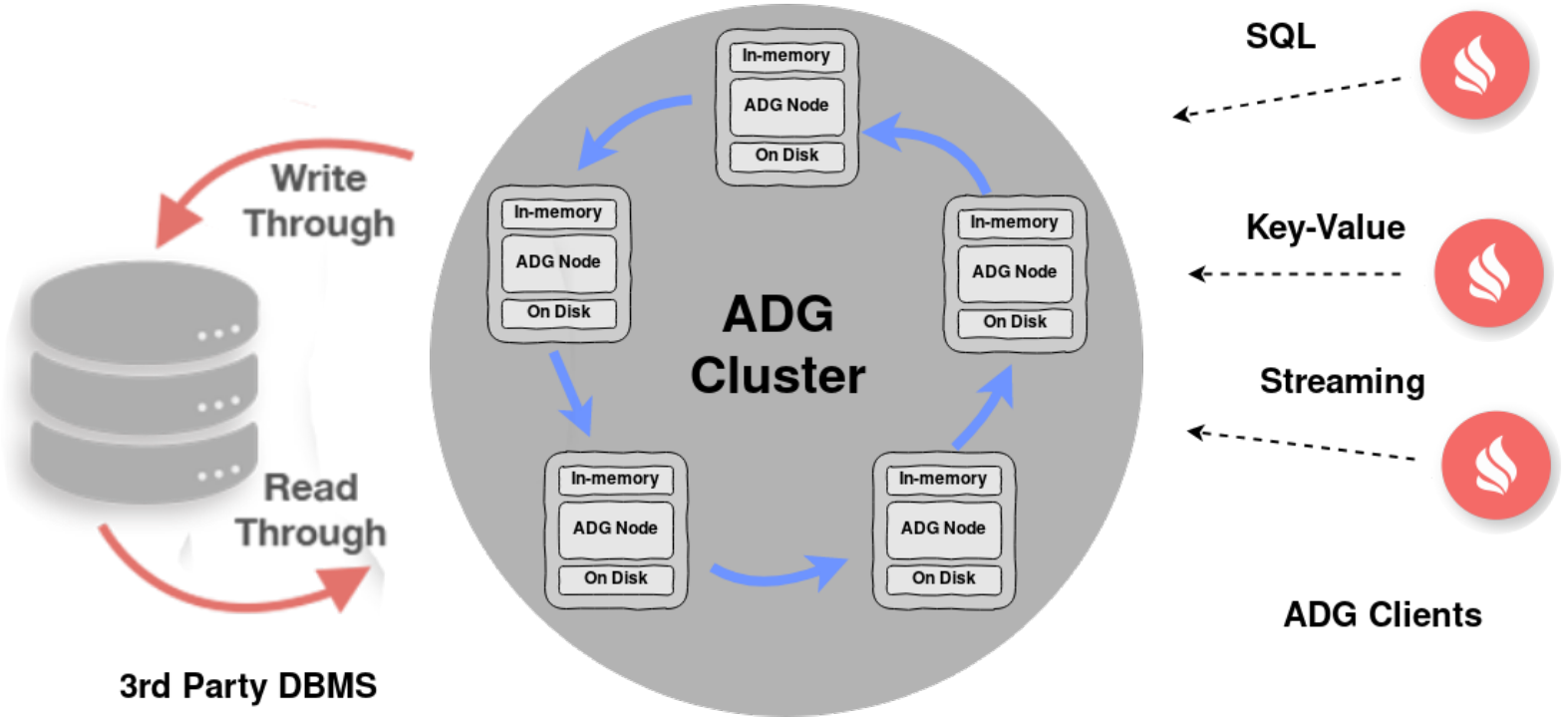


ARENADATA

Integrate Apache Ignite with Arenadata DB

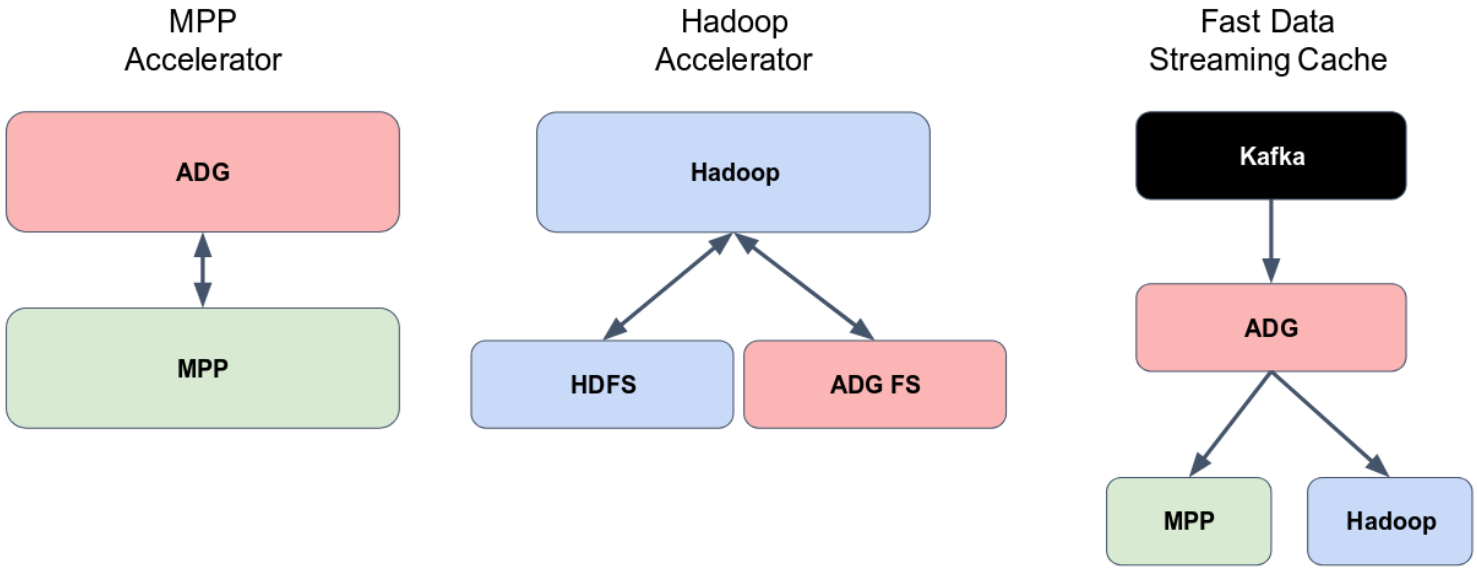
ARENADATA

Arenadata Grid



ARENADATA

Arenadata Grid Use Cases



Arenadata DB Architecture

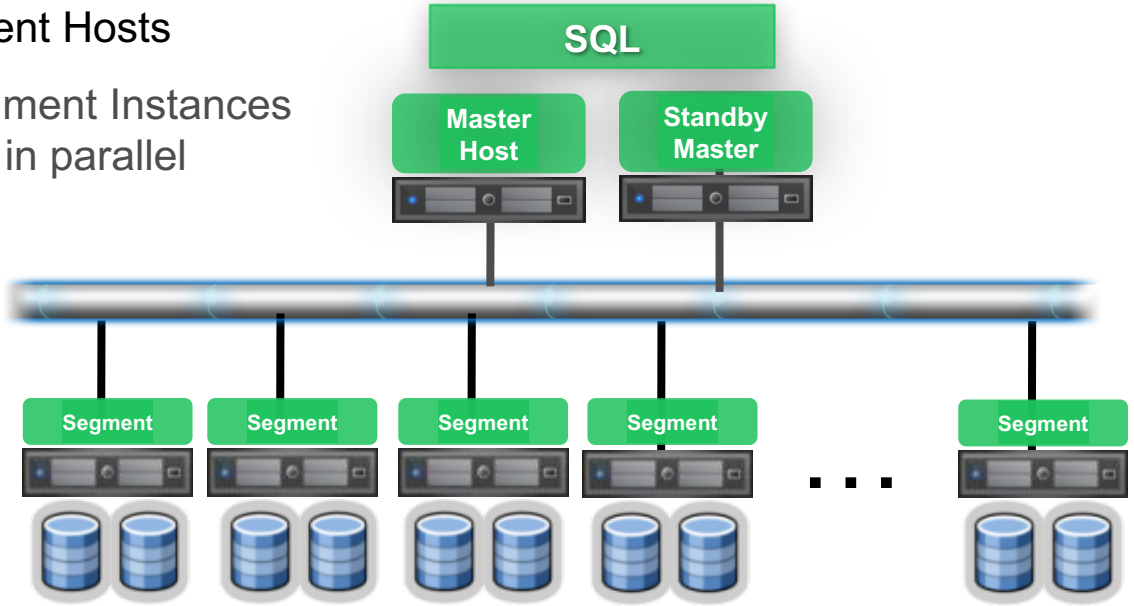
Flexible framework for processing large datasets

Master Host and Standby Master Host
Master coordinates work with Segment Hosts

Segment Host with one or more Segment Instances
Segment Instances process queries in parallel

Segment Hosts have their own CPU, disk and memory (shared nothing)

High speed interconnect for continuous pipelining of data processing



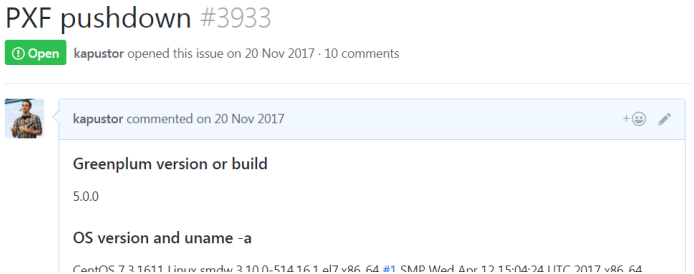
ARENADATA

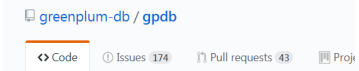
Greenpum Core Development

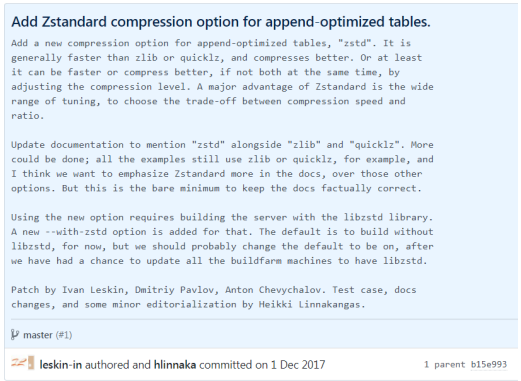
- Zstandard support (will be added to stable at 6.0.0 due to naming convention)

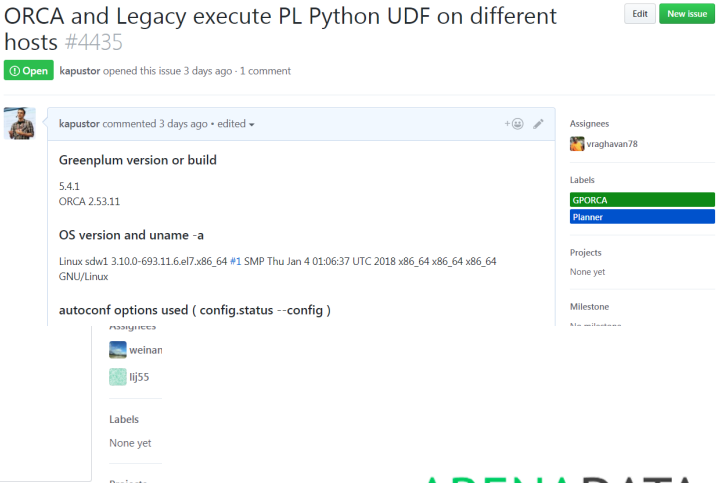
PXF development: we bet a lot. Ignite integration, push down feature, JDBC & Ignite stable release

- Few bugs and a lot of issues





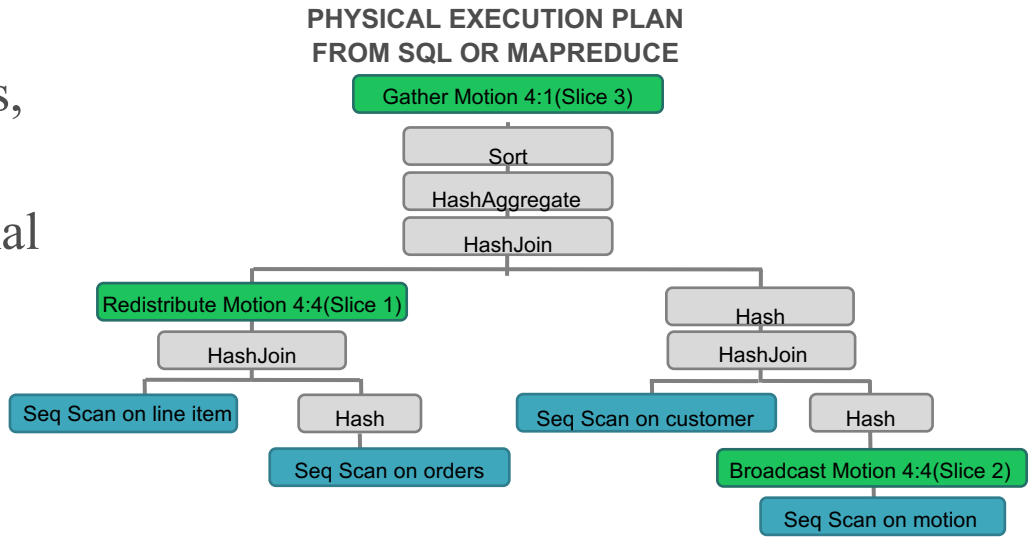




ARENADATA

Parallel Query Optimizer

- Cost-based optimization looks for the most efficient plan
- Physical plan contains scans, joins, sorts, aggregations, etc.
- Global planning avoids sub-optimal ‘SQL pushing’ to segments
- Directly inserts ‘motion’ nodes for inter-segment communication

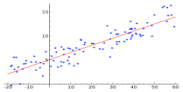
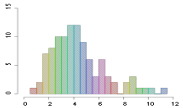
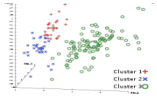


ARENADATA

MADlib: Toolkit for Advanced Big Data Analytics

- Better Parallelism
  - Algorithms designed to leverage MPP or Hadoop architecture
- Better Scalability
  - Algorithms scale as your data set scales
  - No data movement
- Better Predictive Accuracy
  - Using all data, not a sample, may improve accuracy
- Open Source
  - Available for customization and optimization by user

MADlib In-Database Functions

Predictive Modeling Library

Generalized Linear Models
- Linear Regression
- Logistic Regression
- Multinomial Logistic Regression
- Cox Proportional Hazards
- Regression
- Elastic Net Regularization
- Sandwich Estimators (Huber white, clustered, marginal effects)

Machine Learning Algorithms
- ARIMA
- Principal Component Analysis (PCA)
- Association Rules (Affinity Analysis, Market Basket)
- Topic Modeling (Parallel LDA)
- Decision Trees
- Ensemble Learners (Random Forests)
- Support Vector Machines
- Conditional Random Field (CRF)
- Clustering (K-means)
- Cross Validation

Matrix Factorization
- Singular Value Decomposition (SVD)

Linear Systems
- Sparse and Dense Solvers

Descriptive Statistics

Sketch-based Estimators
- CountMin (Cormode-Muthukrishnan)
- FM (Flajolet-Martin)
- MFV (Most Frequent Values)

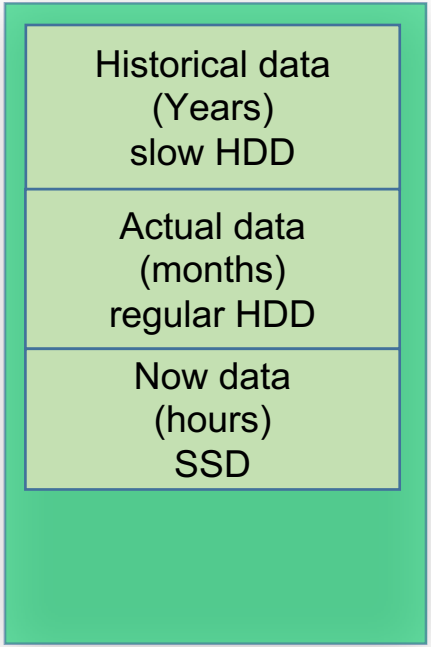
Correlation
Summary

Support Modules

Array Operations
Sparse Vectors
Random Sampling
Probability Functions

ARENADATA

Polymorphic Table Storage

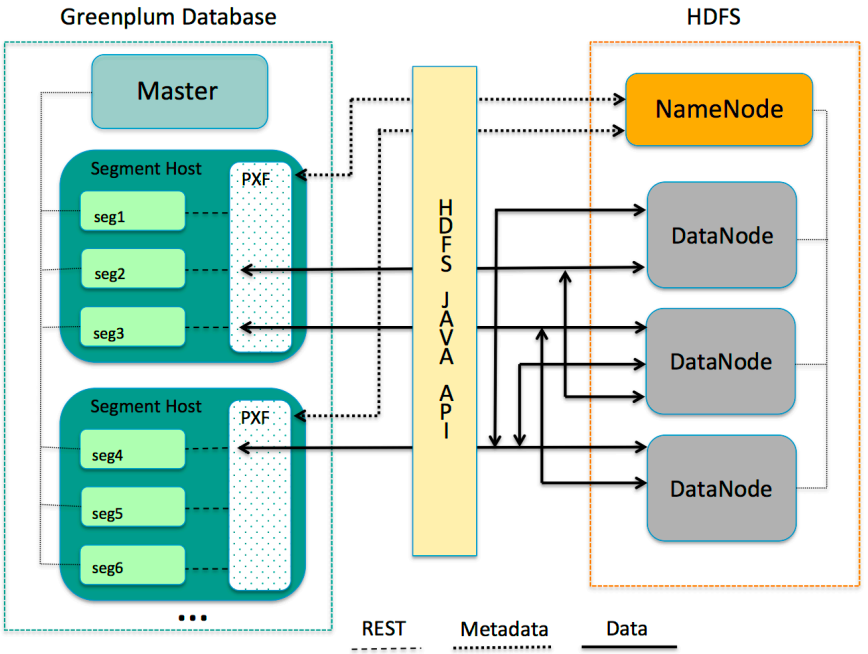


Single table

- Provide the choice of processing model for any table or any individual partition
  - Enable Information Lifecycle Management (ILM)
- Storage types can be mixed within a table or database
  - Four table types: heap, row-oriented AO, column-oriented, external
  - Block compression: Gzip (levels 1-9), Zstd
  - Columnar compression: RLE

Platform eXtension Framework (PXF)

- An advanced version of Greenplum external tables
- Supports connectors for HDFS, HBase and Hive, JDBC, Ignite (Arenadata DB)
- Provides extensible framework API to enable custom connector



ARENADATA

PXF Profiles

- HDFS Files
- Ignite
- JDBC
- Avro
- HBase
- Hive
  - Text based
  - SequenceFile
  - RCFile
  - ORCFile

```
CREATE EXTERNAL TABLE pxf_sales_part(
  item_name TEXT,
  item_type TEXT,
  supplier_key INTEGER,
  item_price DOUBLE PRECISION,
  delivery_state TEXT,
  delivery_city TEXT
)
LOCATION
('pxf://grid_host?Profile=Ingite&IGNITE_CACHE=test&BUFFER_
SIZE=10000');
```

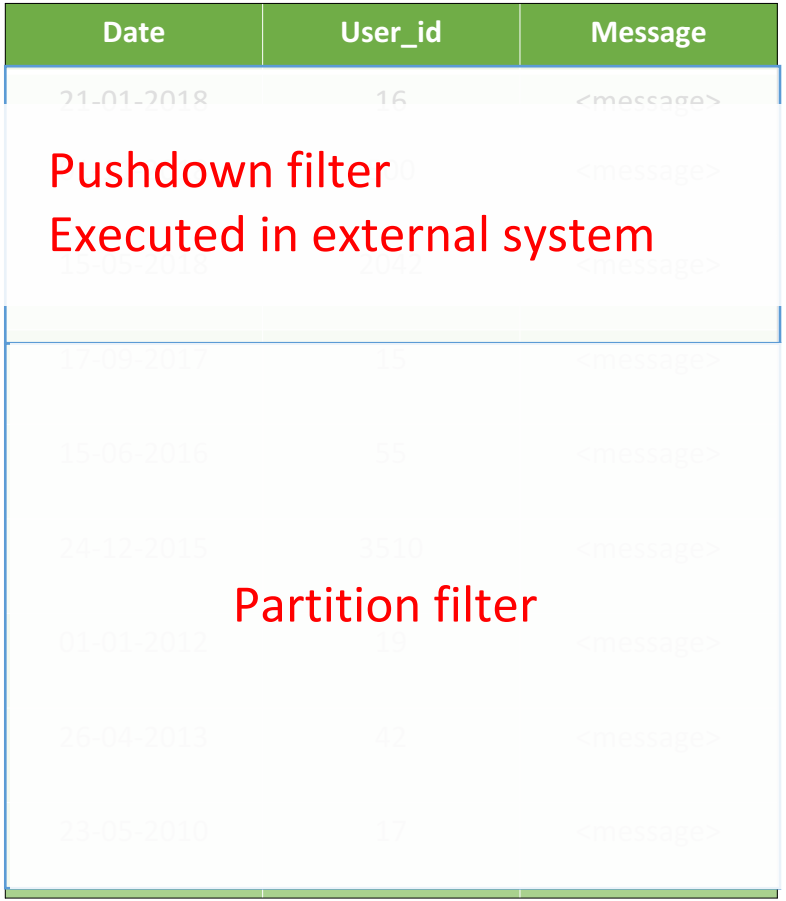
PXF Profiles

```
<profile>
  <name>Ignite</name>
  <plugins>
    <fragmenter>IgniteFragmenter</fragmenter>
    <accessor>IgniteAccessor</accessor>
    <resolver>IgniteResolver</resolver>
    <analyzer>IgniteAnalyzer</analyzer>
  </plugins>
</profile>
```

PXF Classes

- Fragmenter – returns a list of source data fragments and their location
- Accessor – access a given list of fragments read them and return records
- Resolver – deserialize each record according to a given schema or technique
- Analyzer – returns statistics about the source data

PXF Pushdown Feature



Grid external table
Latency: milliseconds
Cost per GB: $$$

... partition by Date (
partition1: Date => 01-01-2018
partition2: Date < 01-01-2018 and Date => 01-01-2015
partition3: Date < 01-01-2015)

Regular ADB table
Latency: seconds
Cost per GB: $$

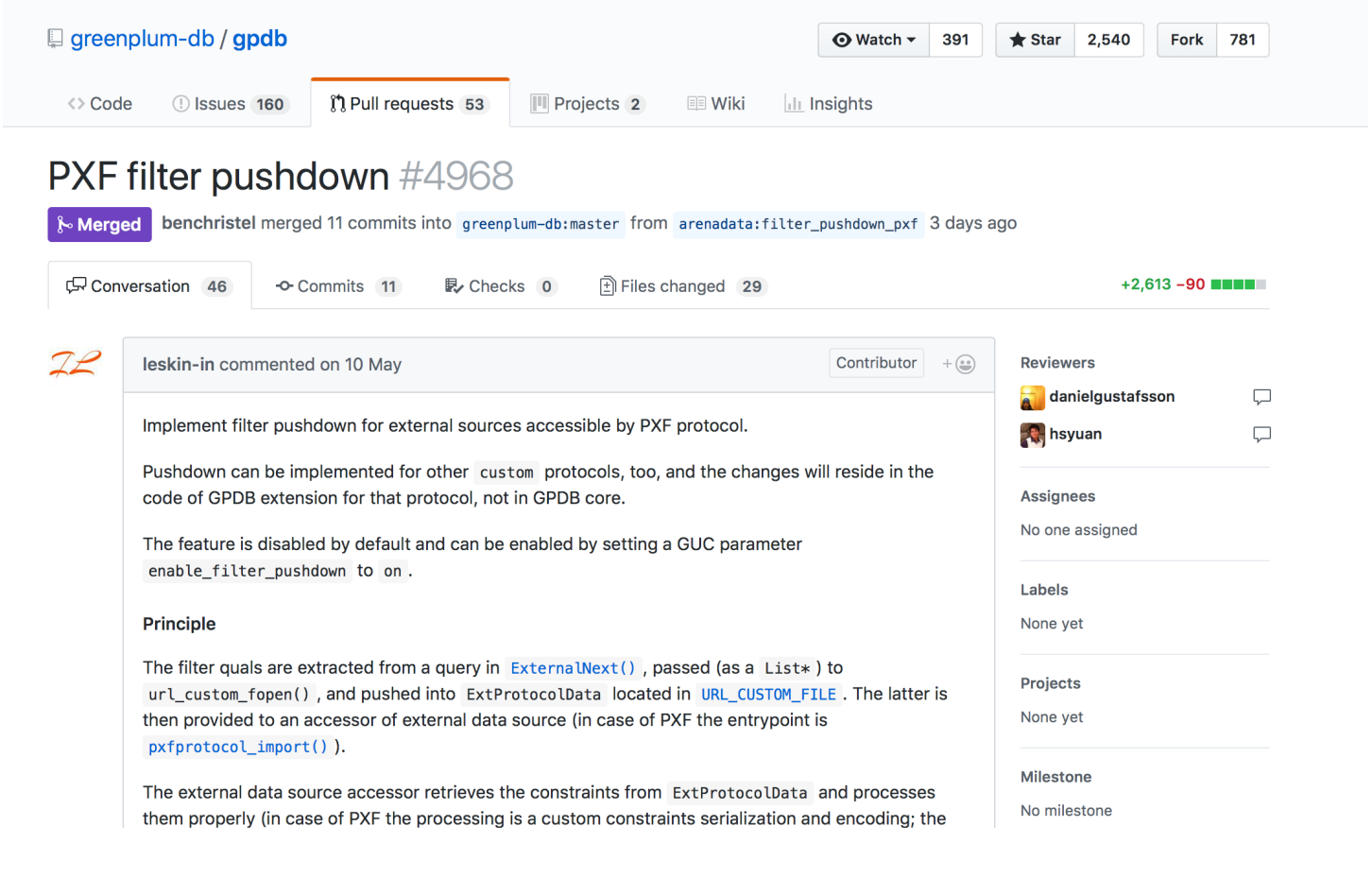
... where Date > 20-01-2018
... where Date < 18-09-2017
... where Date > 16-06-2017
AND User_id < 400

Pushdown

Hadoop external table
Latency: tens of seconds
Cost per GB: $

ARENADATA

PXF Pushdown Feature



greenplum-db / gpdb

Watch 391 | Star 2,540 | Fork 781

Code | Issues 160 | Pull requests 53 | Projects 2 | Wiki | Insights

PXF filter pushdown #4968

Merged **benchristel** merged 11 commits into `greenplum-db:master` from `arenadata:filter_pushdown_pxf` 3 days ago

Conversation 46 | Commits 11 | Checks 0 | Files changed 29 | +2,613 −90

leskin-in commented on 10 May — Contributor

Implement filter pushdown for external sources accessible by PXF protocol.

Pushdown can be implemented for other `custom` protocols, too, and the changes will reside in the code of GPDB extension for that protocol, not in GPDB core.

The feature is disabled by default and can be enabled by setting a GUC parameter `enable_filter_pushdown` to `on`.

Principle

The filter quals are extracted from a query in `ExternalNext()`, passed (as a `List*`) to `url_custom_fopen()`, and pushed into `ExtProtocolData` located in `URL_CUSTOM_FILE`. The latter is then provided to an accessor of external data source (in case of PXF the entrypoint is `pxfprotocol_import()`).

The external data source accessor retrieves the constraints from `ExtProtocolData` and processes them properly (in case of PXF the processing is a custom constraints serialization and encoding; the

Reviewers
danielgustafsson
hsyuan

Assignees
No one assigned

Labels
None yet

Projects
None yet

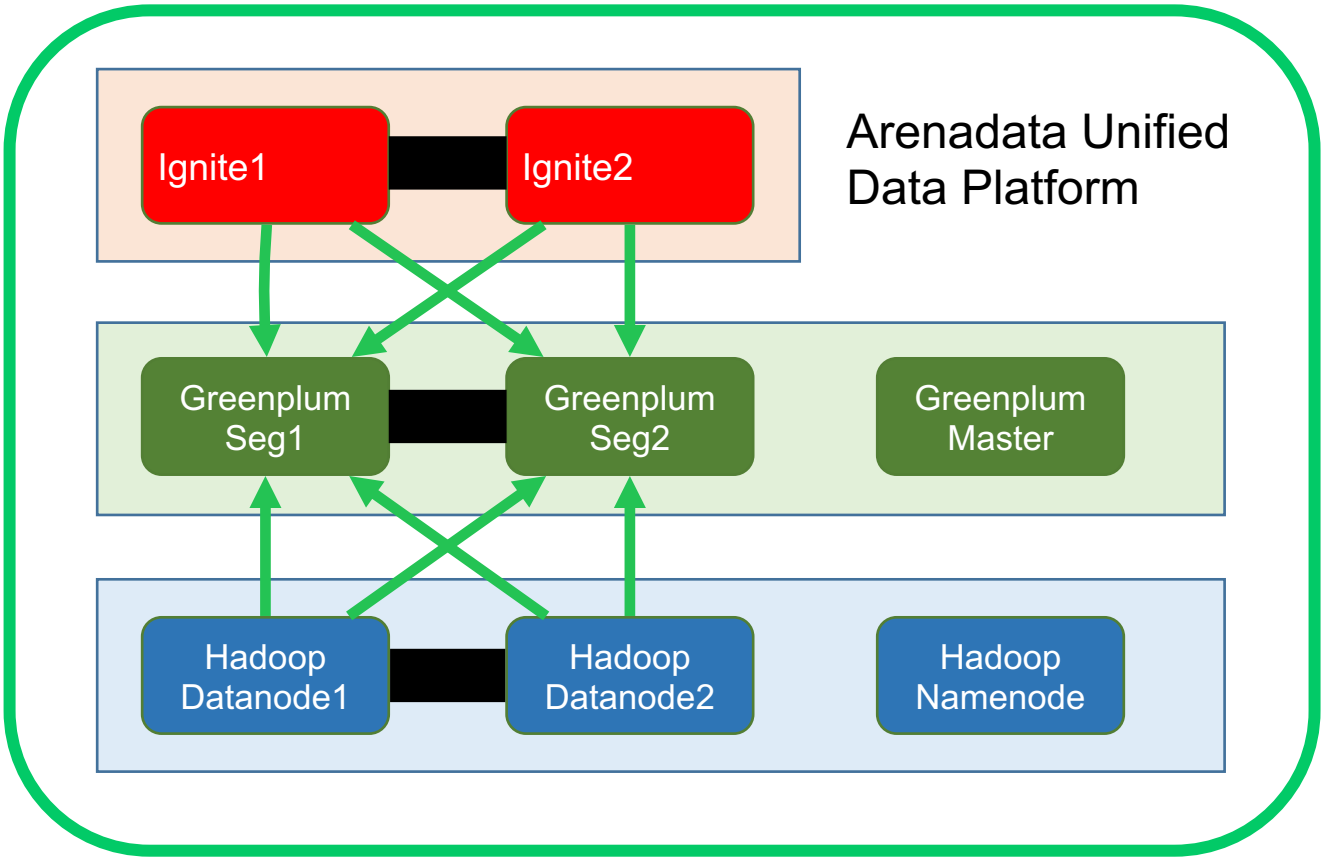
Milestone
No milestone

ARENADATA

Using power of In-Memory computing with MPP

ARENADATA

Test Bench



ARENADATA

Creating Table in MPP

Create table in Greenplum

```
%jdbc
drop  table if exists bcs_gp_1;
```

id	name	price	time
24	paper	34.5	2017-02-15 15:22:00
44	buttons	34	2017-02-15 15:22:00
22	plastic	17.5	2017-02-15 15:20:00
26	bananas	1.5	2017-02-15 15:23:00
42	plants	17.5	2017-02-15 15:20:00
46	water	1.5	2017-02-15 15:23:00
23	glass	18	2017-02-15 15:21:00
43	wood	18.5	2017-02-15 15:21:00
25	apples	1.5	2017-02-15 15:23:00

```
(44,'buttons', '34.0','2017-02-15 15:22:00'),
(45,'coal', '1.0','2017-02-15 15:23:00'),
(46,'water', '1.5','2017-02-15 15:23:00');
select * from bcs_gp_1
```

Creating External Table for Apache Ignite & Load Data

Create in-mem external table

id	name	price	time
3	uran	18.5	2017-02-15 15:16:00
1	gold	15.5	2017-02-15 15:15:00
2	silver	16.5	2017-02-15 15:16:00
4	steel	15.5	2017-02-15 15:17:00
5	aluminium	25.5	2017-02-15 15:18:00
6	4ugun	18.5	2017-02-15 15:19:00

Creating External Table in Hive & Load Data

Create external hive table

id	name	price	time
61	baskets	15.5	2017-02-15 15:42:00
62	notebooks	16.5	2017-02-15 15:19:00
63	books	18.5	2017-02-15 15:26:00
64	soft	15.5	2017-02-15 15:57:00
65	tables	25	2017-02-15 15:48:00
66	chairs	18	2017-02-15 15:39:00

Exchange Partitions with External Tables

Exchanging partitions

```
%jdbc
alter table bcs_gp_1 EXCHANGE PARTITION for (RANK(1)) with table public.bcs_gp_1_ext_inmem WITHOUT VALIDATION;
alter table bcs_gp_1 EXCHANGE PARTITION for (RANK(4)) with table public.bcs_gp_1_ext_hive WITHOUT VALIDATION;
```

```
Query executed successfully. Affected rows : 0

Query executed successfully. Affected rows : 0
```

ARENADATA

Target Table

```
select relname "child table", consrc "check",relstorage "storage"
from pg_inherits i
join pg_class c on c.oid = inhrelid
join pg_constraint on c.oid = conrelid
where contype = 'c'
and inhparent = 'bcs_gp_1'::regclass order by 1
```

FINISHED

child table	check	storage
bcs_gp_1_1_prt_1	((id >= 0) AND (id < 20))	x
bcs_gp_1_1_prt_2	((id >= 21) AND (id < 40))	h
bcs_gp_1_1_prt_3	((id >= 41) AND (id < 60))	c
bcs_gp_1_1_prt_4	((id >= 61) AND (id < 80))	x

Execution Plan

```
QUERY PLAN
Gather Motion 4:1  (slice1; segments: 4)  (cost=0.00..14098.75 rows=346634 width=52)
  Rows out:  11 rows at destination with 30 ms to first row, 42 ms to end, start offset by 15 ms.
  ->  Append  (cost=0.00..14098.75 rows=86659 width=52)
        Rows out:  Avg 2.8 rows x 4 workers.  Max 7 rows (seg2) with 0.134 ms to first row, 38 ms to end, start offset by 19 ms.
        ->  Seq Scan on bcs_gp_1_1_prt_2 bcs_gp_1  (cost=0.00..598.75 rows=3325 width=52)
              Filter: id < 40
              Rows out:  Avg 1.2 rows x 4 workers.  Max 2 rows (seg3) with 24 ms to first row, 38 ms to end, start offset by 19 ms.
        ->  External Scan on bcs_gp_1_1_prt_1 bcs_gp_1  (cost=0.00..13500.00 rows=83334 width=52)
              Filter: id < 40
              Rows out:  6 rows (seg2) with 0.098 ms to first row, 0.107 ms to end, start offset by 19 ms.
Slice statistics:
  (slice0)    Executor memory: 386K bytes.
  (slice1)    Executor memory: 247K bytes avg x 4 workers, 253K bytes max (seg2).
Statement statistics:
  Memory used: 128000K bytes
Optimizer status: legacy query optimizer
Total runtime: 58.697 ms
```

prt2: Greenplum Heap Partition

prt1: Ignite Cache Partition

ARENADATA


Thank you!
FIFA WORLD CUP
RUSSIA 2018
ODPi
RUNTIME
COMPLIANT
odpi.org
Questions?
ARENADATA