

# Apache Kylin on HBase

## Extreme OLAP Engine for Big Data

Shaofeng Shi | 史少锋

Apache Kylin Committer & PMC

August 17, 2018

# Content

## 01 What is Apache Kylin

Apache Kylin introduction, architecture and relationship with others

## 02 Why Apache HBase

Key factors that Kylin selects HBase as the storage engine

## 03 How to use HBase in OLAP

How Kylin works on HBase

## 04 Use Cases

Apache Kylin typical use cases

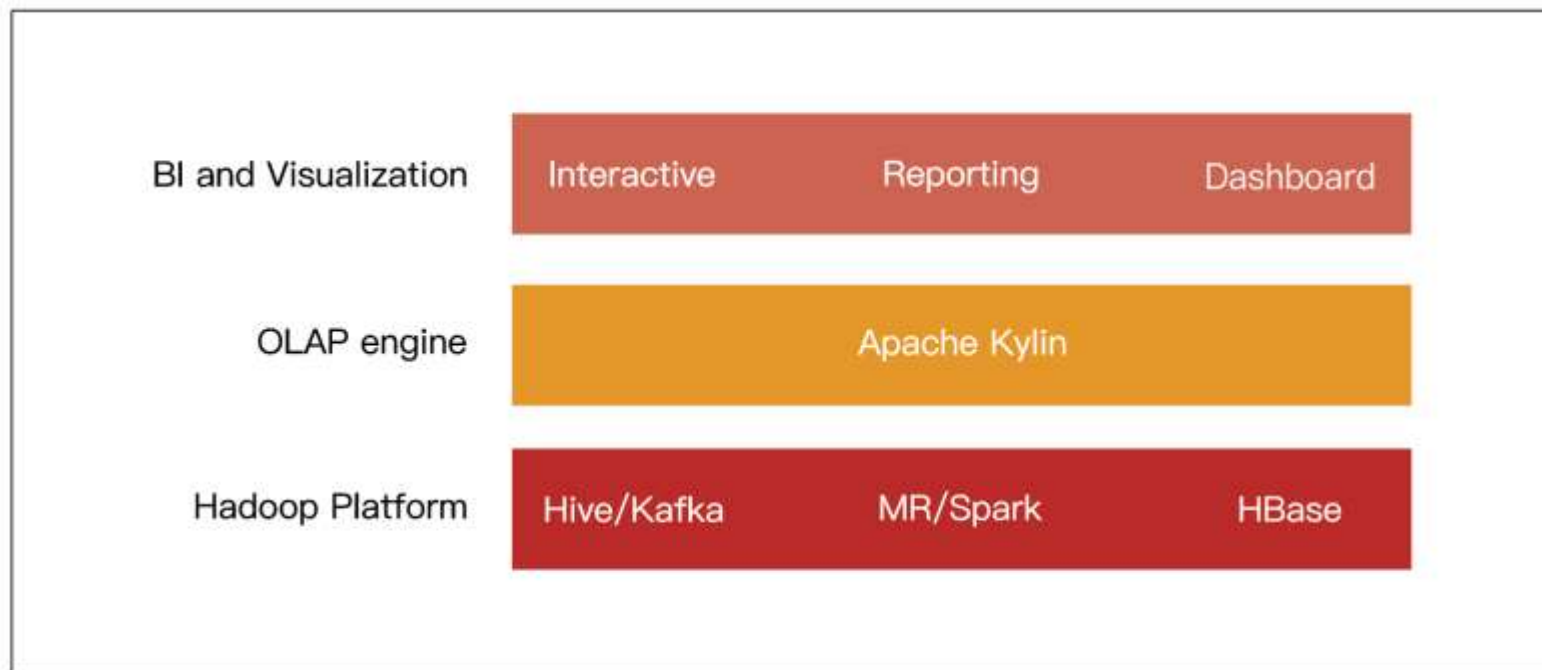
# 01 What is Apache Kylin

# What is Apache Kylin

OLAP engine for Big Data



Apache Kylin is an extreme fast OLAP engine for big data.



# What is Apache Kylin

Key characters

## Real Interactive

Trillion rows data, 99% queries < 1.3 seconds,  
from Meituan.com

## ANSI-SQL

SQL on Hadoop, supports most ANSI SQL  
query functions

## Hadoop Native

Compute and store data with  
MapReduce/Spark/HBase, fully scalable architecture;



## Ease of Use

No programming; User-friendly Web GUI;

## MOLAP Cube

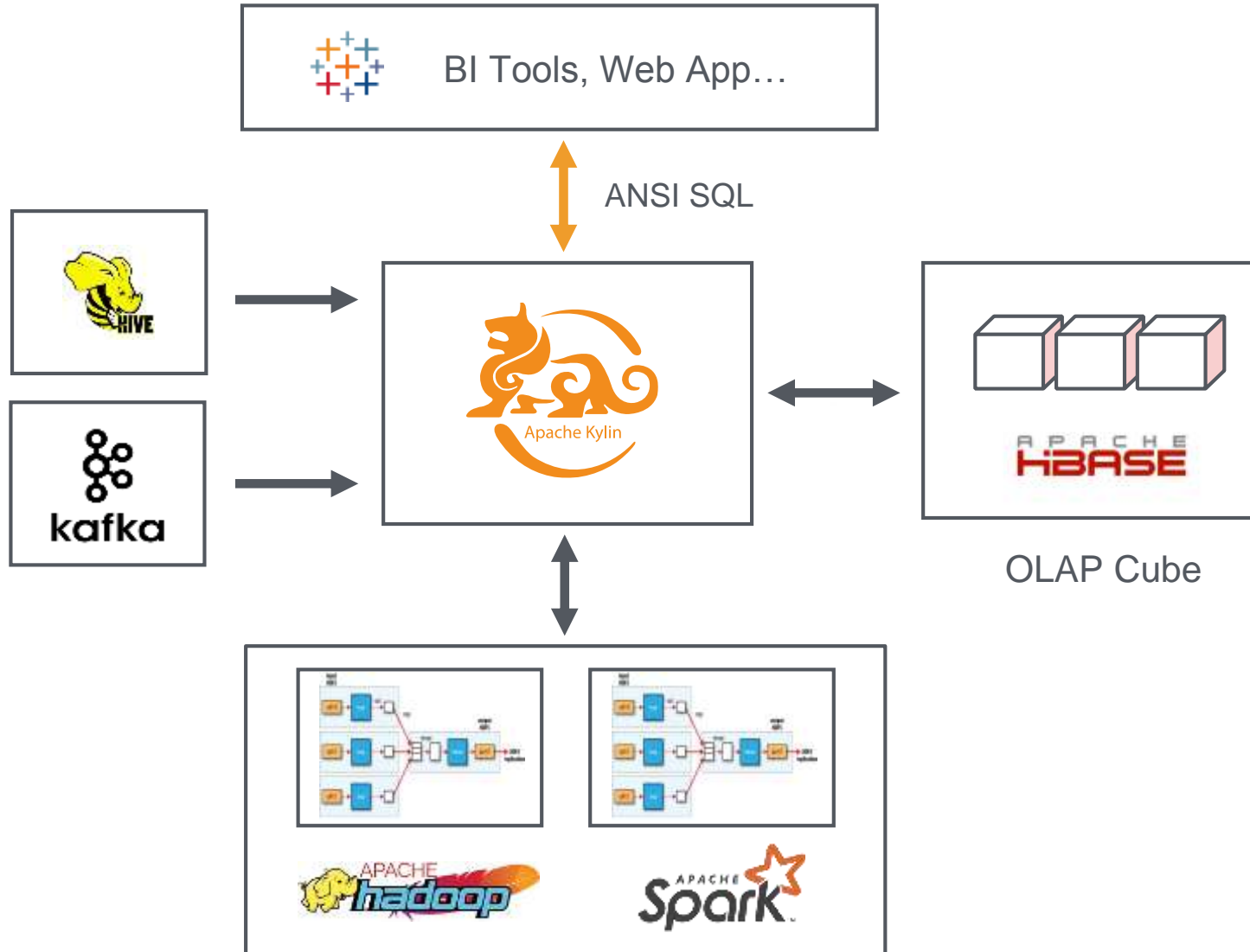
User can define a data model and pre-build in Kylin  
with more than 10+ billions of raw data records

## Seamless BI Integration

JDBC/ODBC/REST API; Supports Tableau, MSTR, Qlik  
Sense, Power BI, Excel and others

# Apache Kylin Architecture

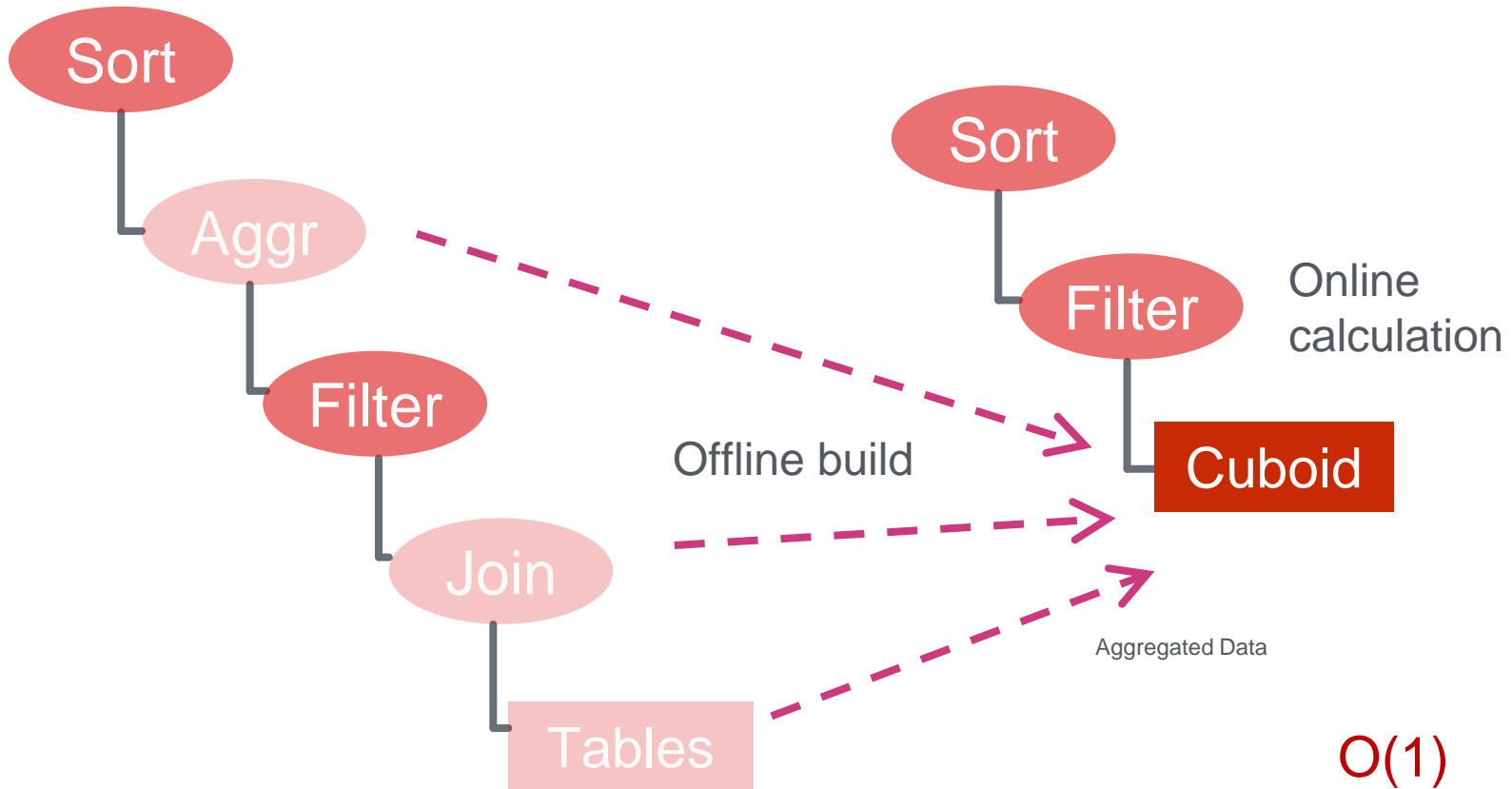
Native on Hadoop, Horizontal Scalable



# Why Kylin is fast

Pre-calculation + Random access

- Join and aggregate data to Cube in offline
- Convert SQL query to Cube visiting
- No join at query time
- Filter with index and do online calculation within memory



# 02 Why Apache HBase



# Criteria of the Storage engine for Kylin

■ Hadoop Native



■ Low Latency



■ High Capacity



■ Wide Adoption



■ Easy to use API



■ Active User Community



# Only HBase Can

## Hadoop Native

HBase is built on Hadoop technologies; It Integrates well with HDFS, MapReduce and other components.

## High Capacity

Supports very large data volume, TB to PB data in one table.

## Easy to use API

## Low Latency

Block cache and Bloom Filters for real-time queries.

## Wide Adoption

Most Hadoop users are running HBase;

## Active User Community

HBase has many active users, which provides many good articles and best practices.

# HBase in Kylin

HBase acts four roles in Kylin

## Massive Storage for Cube

Kylin persists OLAP Cube in HBase, for low latency access.

## MPP for online calculation

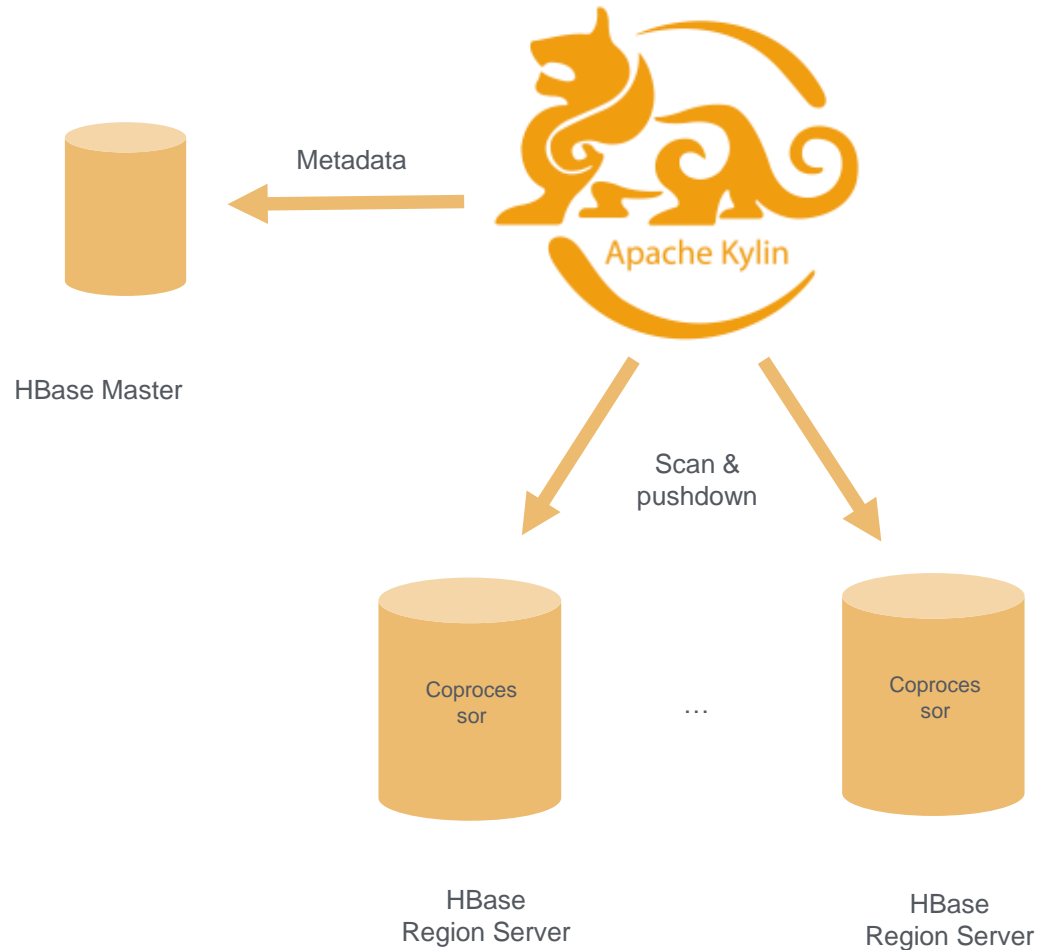
Kylin pushes down calculations to HBase region servers for parallel computing.

## Meta Store

Kylin uses HBase to persist its metadata.

## Cache

Kylin caches big lookup table in HBase.

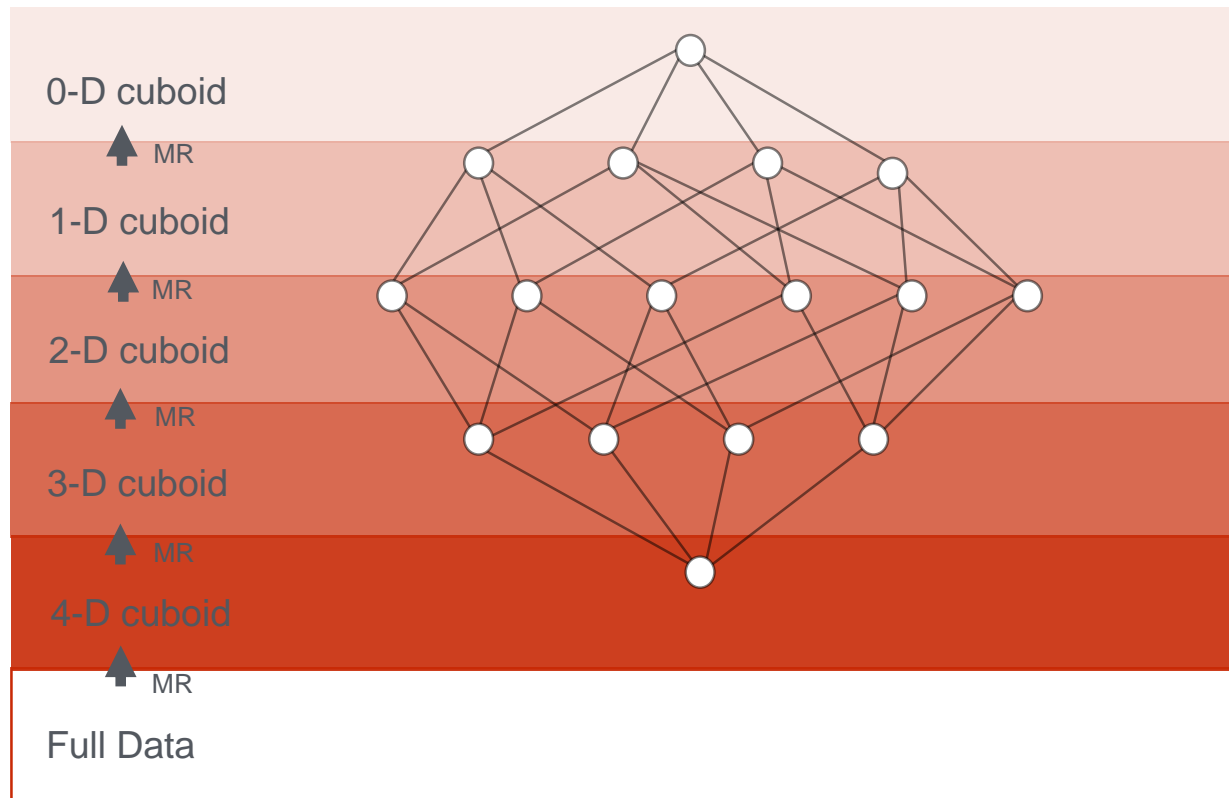


# 03 How to use HBase in OLAP

# How Cube be Built

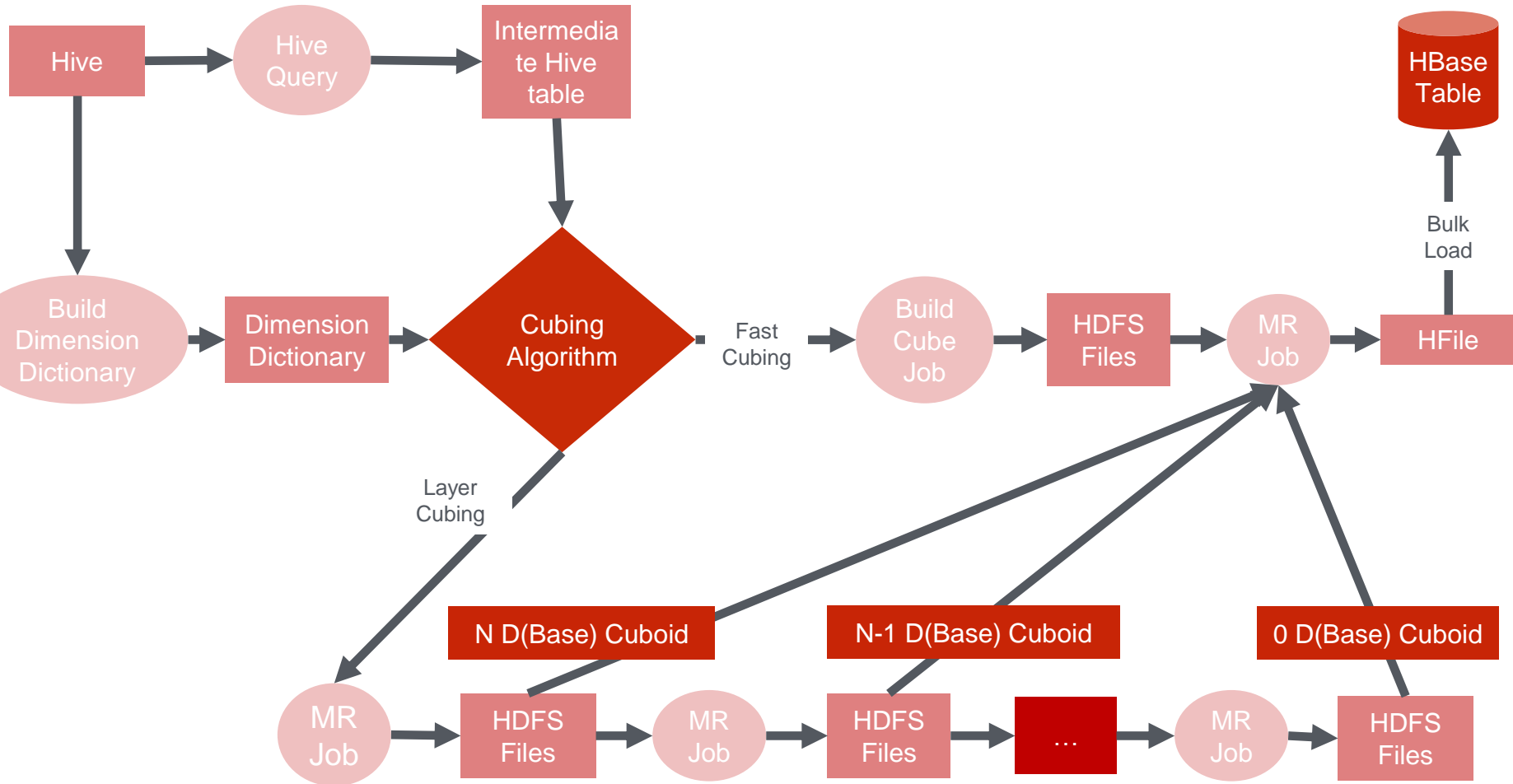
Cube building process

- Kylin uses MR or Spark to aggregate source data into Cube;
- The typical algorithm is by-layer cubing;
- Calculate N-Dimension cuboid first, and then calculate N-1.



# How Cube be Built

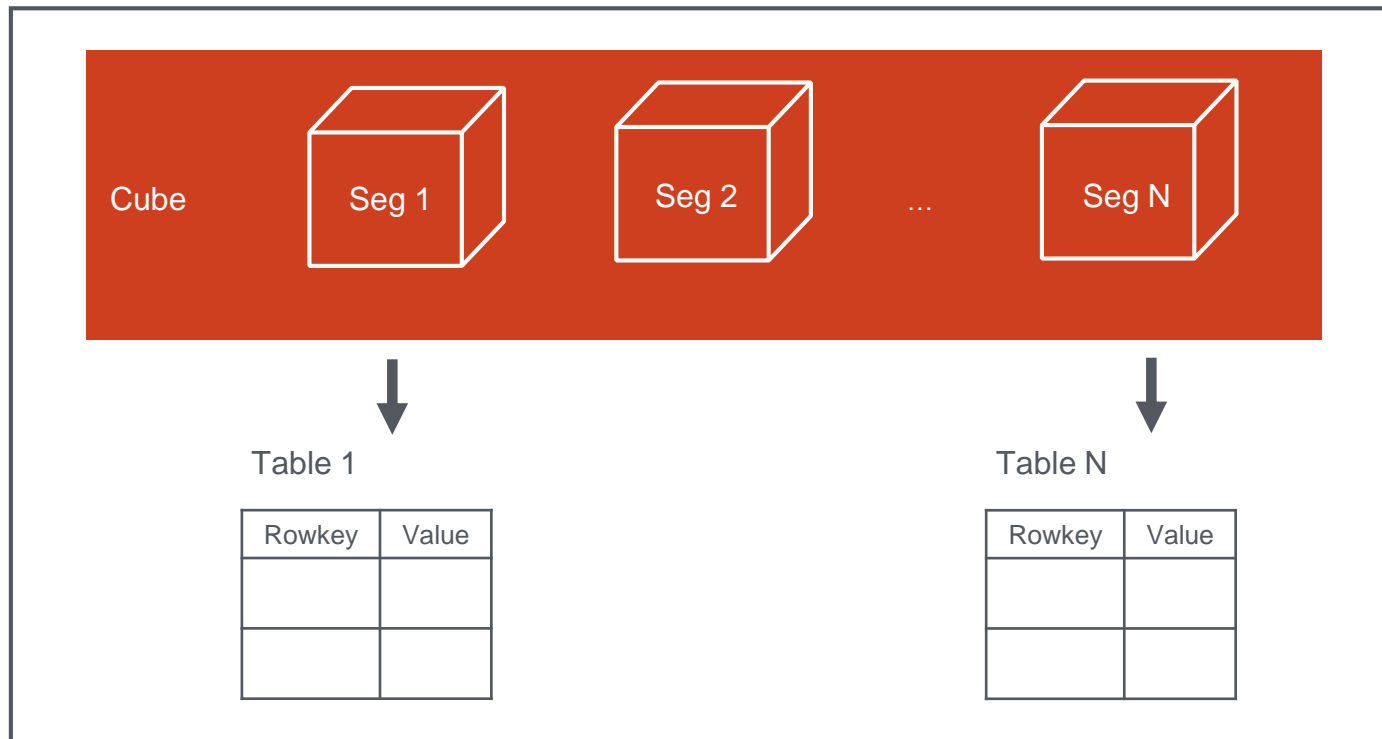
The whole Cube building process



# Cube in HBase

## Cube Structure

- Cube is partitioned into multiple segments by time range
- Each segment is a HBase Table
- Table is pre-split into multiple regions
- Cube is converted into HFile in batch job, and then bulk load to HBase



# HBase Table Format

Rowkey + Value

- Dimension values are encoded to bytes (via dictionary or others)
- Measures are serialized into bytes
- HBase Rowkey format: Shard ID (2 bytes) + Cuboid ID (8 bytes) + Dimensions
- HBase Value: measures serialized bytes
- Table is split into regions by Shard ID
- User can group measures into 1 or multiple column families

Shard ID	Cuboid ID (long)	Dim 1	Dim 2	...	Dim N
----------	---------------------	-------	-------	-----	-------

2 bytes

8 bytes

Rowkey

Measure 1	...	Measure N
-----------	-----	-----------

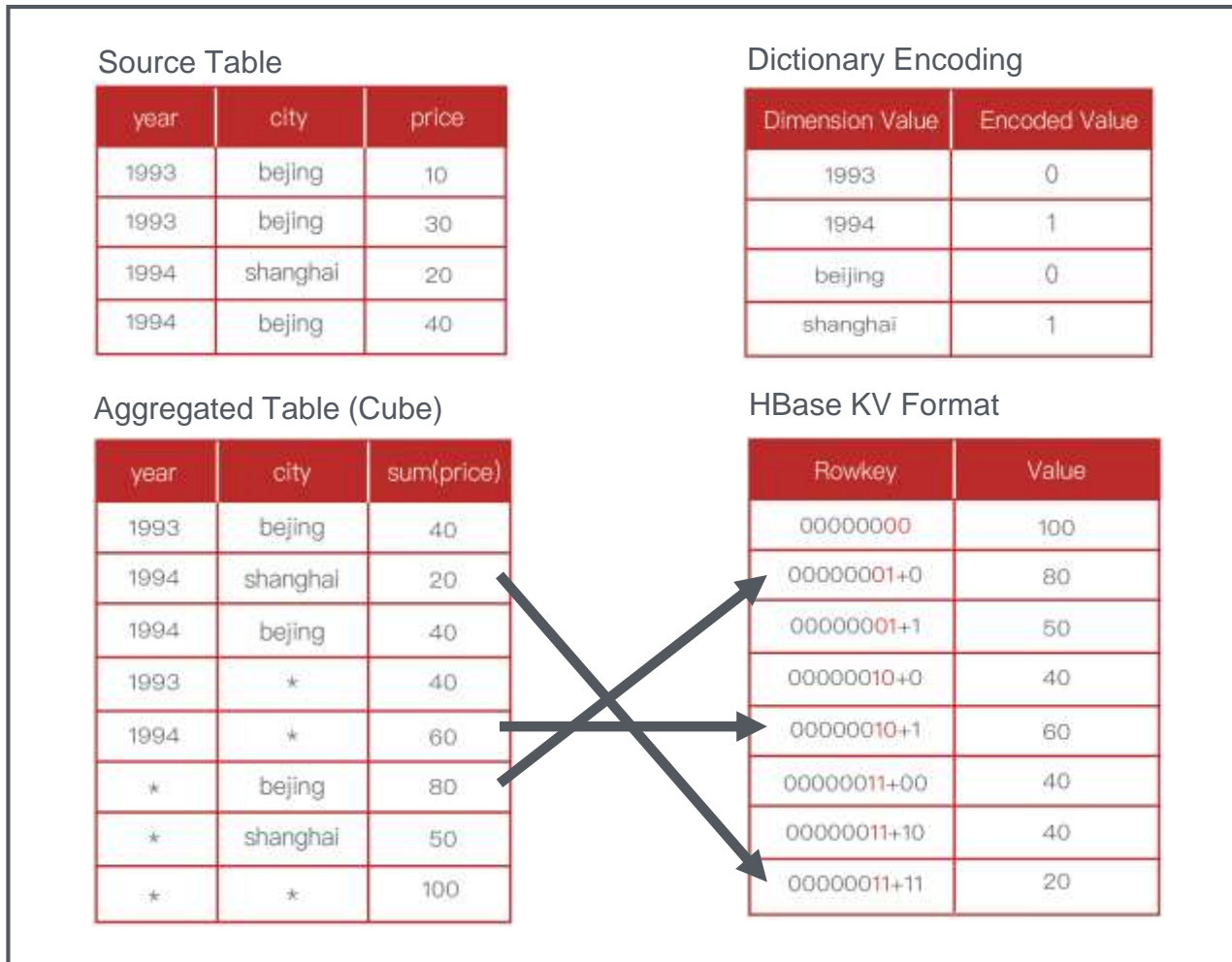
Value



# How Cube be Persisted in HBase

Example: a Tiny Cube

- 2 dimensions and 1 measure; total 4 cuboids: 00,01,10,11
- Please note the Shard ID is not appeared here



# How Cube be Queried

- Kylin parses SQL to get the dimension and measures;
- Identify the Model and Cube;
- Identify the Cuboid to scan;
- Identify the Cube segments;
- Leverage filter condition to narrow down scan range;
- Send aggregation logic to HBase coprocessor, do storage-side filtering and aggregation;
- On each HBase RS returned, de-code and do final processing in Calcite

```
select city , sum(price)
from table where year=
1993 group by year
```



```
Dimension: city, year
Cuboid ID: 00000011
Filter: year=1993 (encoded value: 0)

Scan start: 00000011|0
Scan end: 00000011|1
```

## Good and not-good of HBase for OLAP

HBase is a little complicated for OLAP scenario; HBase supports both massive write + read; while OLAP is read-only.

- ✓ Native on Hadoop.
- ✓ Great performance  
(when search pattern matches  
row key design)
- ✓ High concurrency

- Not a columnar storage
- No secondary index
- Downtime for upgrade  
(update coprocessor need to  
disable table first)

# 04 Use Cases

# 1000+ Global Users

## Internet

- eBay
- Yahoo! Japan
- Baidu 百度
- Meituan 美团
- NetEase 网易
- Expedia
- JD 京东
- VIP 唯品会
- 360
- TOUTIAO 头条
- ...

## FSI

- CCB 建设银行
- CMB 招商银行
- SPDB 浦发银行
- CPIC 太平洋保险
- CITIC BANK 中信银行
- UnionPay 中国银联
- HuaTai 华泰证券
- GuoTaiJunAn 国泰君安证券
- ...

## Telecom

- China Mobile
- China Telecom
- China Unicom
- AT & T
- ...

## Manufacturing

- SAIC 上汽集团
- Huawei 华为
- Lenovo 联想
- OPPO
- XiaoMi 小米
- VIVO
- MeiZu 魅族
- ...

## Others

- MachineZone
- Glispa
- Inovex
- Adobe
- iFLY TEK科大讯飞
- ...

# Use Case – OLAP on Hadoop

Meituan: Top O2O company in China

## Challenge

- Slow performance with previous MySQL option Heavy development efforts with Hive solution
- Huge resources for Hive job
- Analysts can't access directly for data on Hadoop

## Solution

- Apache Kylin as core OLAP on Hadoop solution
- SQL interface for internal users
- Active participate in open source Kylin community

973 Cubes, 8.9+ trillion rows, Cube size 971 TB  
3,800,000 queries/day, TP90 < 1.2 second

(Data in 2018/08)

Supporting all Meituan business lines

# Use Case – Shopping Reporting

Yahoo! Japan: the most visited website in Japan

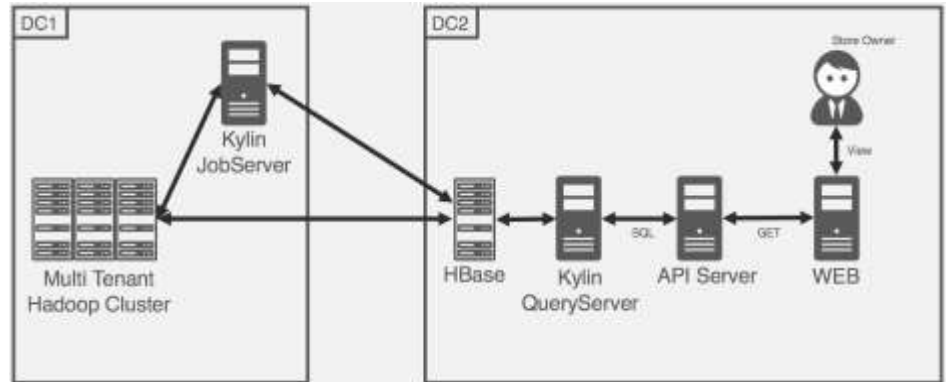
We provide a reporting system that show statistics for store owners.

e. g. impressions, clicks and sales.

Our reporting system used Impala as a backend database previously. It took a long time (about 60 sec) to show Web UI.

In order to lower the latency, we moved to Apache Kylin.

Average latency < 1sec for most cases



Thanks to low latency with Kylin,  
we become possible to focus on  
adding functions for users.

# We Are Hiring



WeChat: Kyligence



WeChat: Apache Kylin



The background features a smooth gradient from dark blue on the left to bright orange on the right. Overlaid on this are several sets of thin, parallel, wavy lines that create a sense of depth and movement, resembling a stylized landscape or a digital wave pattern.

Thanks