清華大學
Tsinghua University

A PRACTICE OF TPC-DS MULTIDIMENSIONAL IMPLEMENTATION ON NOSQL DATABASE SYSTEMS

HONGWEI ZHAO AND XIAOJUN YE

HWZHAO73@GMAIL.COM, YEXJ@TSINGHUA.EDU.CN

SCHOOL OF SOFTWARE, TSINGHUA UNIVERSITY

BEIJING 100084,CHINA

I.S.E. School of Software Tsinghua

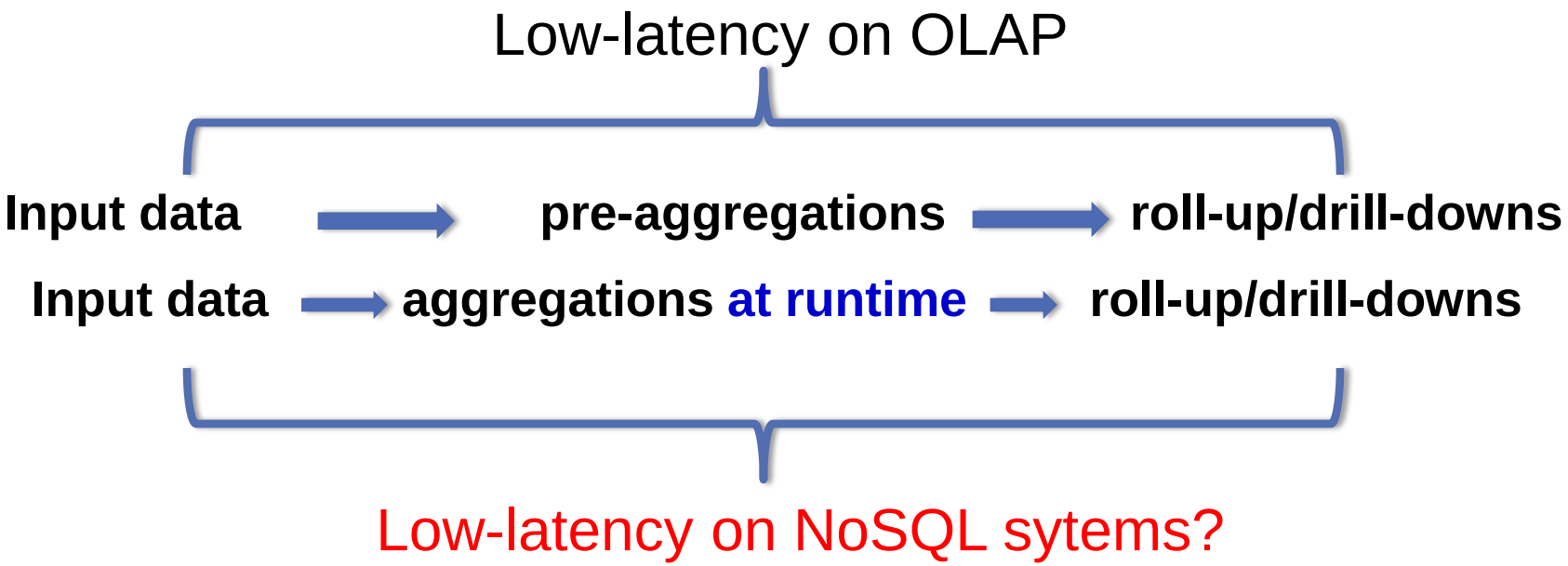
DB Test Group

OUTLINE

- **Motivation**
- Methodology for MOLAP
- Description for MOALP engine
- Experimenting
- Conclusion

MOTIVATION

- **Practice MOLAP cube operations on NoSQL Databases:**
  - OLAP operation implementation techniques
  - Interactive queries experiments and analysis



WHY MOLAP

- **MOLAP is online analytical processing that indexes directly into a multidimensional database**
  - User can be able to view different aspects or facets of data aggregates stored in a multidimensional array
- **The limitations in MOLAP are that it is not very scalable and can only handle limited amounts of data since calculations are predefined (storage and cache) in the cube**
  - Not all dimensions are used in a query
  - Not all queries are used with the same frequency
  - ... …
- **OLAP engine practice on NoSQL systems for low-latency?**
  - Space & Efficiency
  - Better scalability

WHEN AGGREGATING

Aggregate at runtime

Advantage

- Most flexible
- Fast – scatter gather
- Space efficient

Disadvantage

- I/O, CPU intensive
- Slow for larger data
- Low throughput

Pre-aggregate

Advantage

- Fast
- Efficient – O(1)
- High throughput

Disadvantage

- More effort to process (latency)
- Combinatorial explosion (space)
- No flexibility

Cosmin Lehene Low Latency "*OLAP" with HBase* - HBaseCon 2012

BALANCE FOR AGGREGATION

- **Our solution:**
  - Pre-aggregate base cuboid based on data model, Aggregate other cuboids at runtime according to user queries
    - Space efficient
    - Efficient O(1) after first query: high throughput
    - More flexible for user queries
- **Latency balanced in basic cuboid building and user querying**

OUTLINE

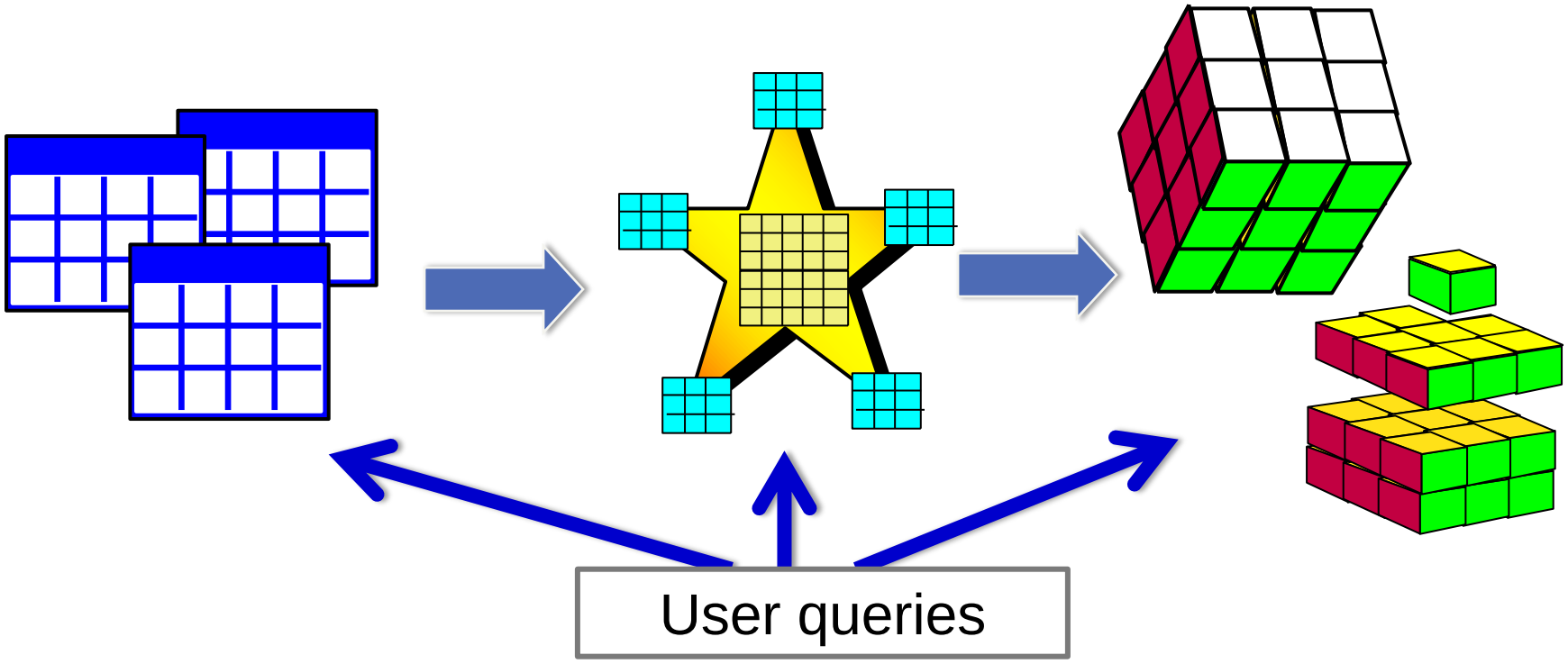
- Motivation
- ➢ **Methodology for MOLAP**
- Description for MOALP engine
- Experimenting
- Conclusion

ETL FOR CUBE BUILDING

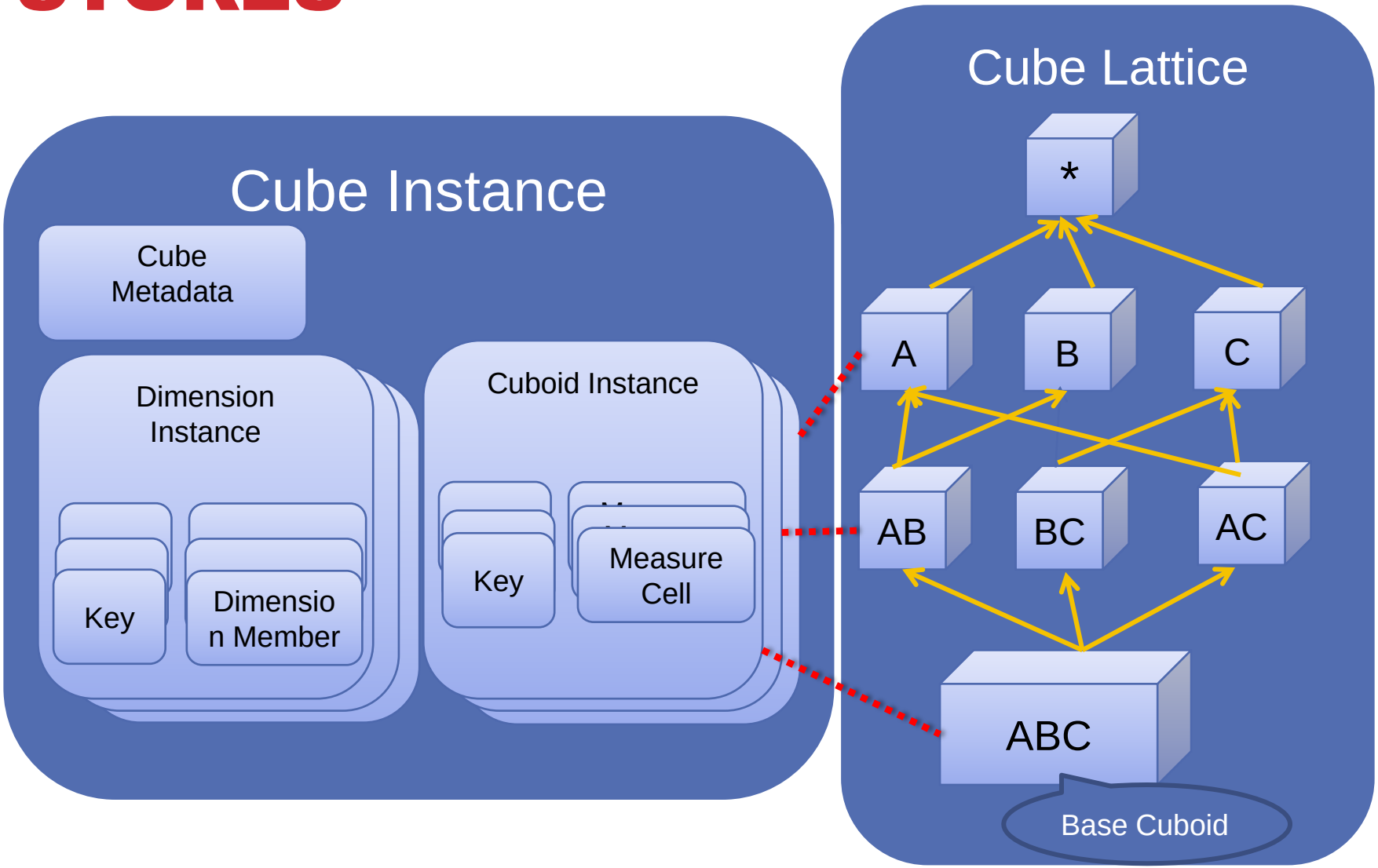
TPC-DS data files

Star schema

Cube data



CUBE MODEL ON KEY-VALUE STORES



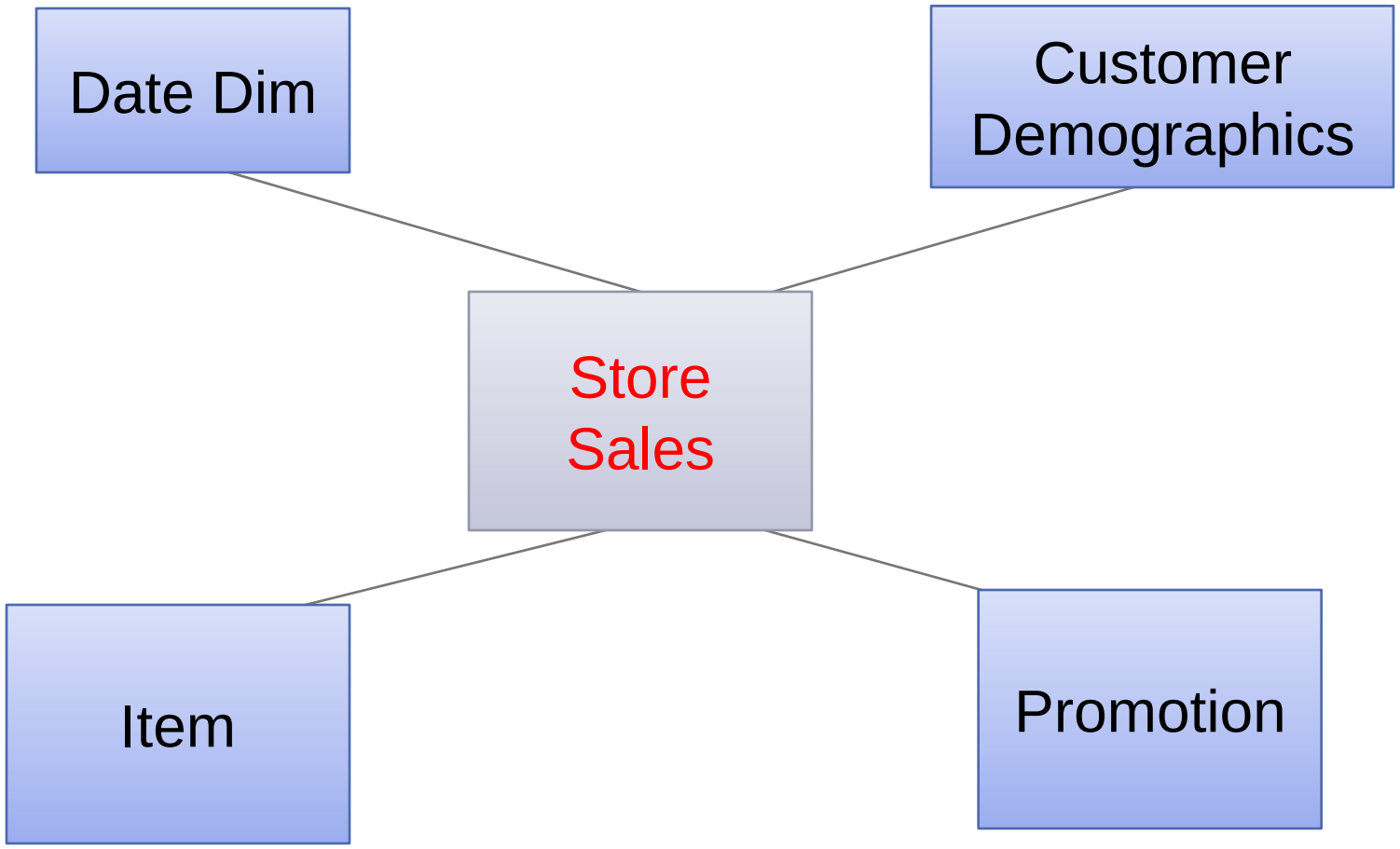
EXAMPLE: TPC-DS QUERY 7

```
select i_item_id, avg(ss_quantity) agg1, avg(ss_list_price) agg2,
avg(ss_coupon_amt) agg3, avg(ss_sales_price) agg4
from store_sales, customer_demographics, date_dim, item, promotion
where ss_sold_date_sk = d_date_sk and
      ss_item_sk = i_item_sk and
      ss_cdemo_sk = cd_demo_sk and
      ss_promo_sk = p_promo_sk and
      cd_gender = '[GEN]' and
      cd_marital_status = '[MS]' and
      cd_education_status = '[ES]' and
      (p_channel_email = 'N' or p_channel_event = 'N') and
      d_year = [YEAR]
 group by i_item_id
 order by i_item_id
```

MDX FOR QUERY 7

```
select { i_item_id } on rows,
        { avg(ss_quantity), avg(ss_list_price),
        avg(ss_coupon_amt),
avg(ss_sales_price) }
        on columns
from store_sales_cube
where (cd_gender .[Male],
      cd_marital_status .[Single],
      cd_education_status .[College],
        d_year.[2000])
```

STAR SCHEMA FOR QUERY 7, 42, 52, 55



date_sk	year	moy	dom
3428	2001	12	21
3617	2003	8	15

cdemo_sk	gender	marital	education
172	M	single	4-years
280	F	married	master

date_sk	cdemo_sk	..	price
3617	280		46.03
3428	172		99.54
…	…	..	…

a) Fact table in star schema

year_key	year
001	2001
011	2003

mon_key	mon
1000	8
1100	12

day_key	day
01111	15
10101	21

bitmap key		avg(price)
011100001111	10011100	46.03
001110010101	01001010	99.54
…		…

b) Cuboid cells from decomposed fact table

gen_key	gen
01	M
10	F

mar_key	mar
001	single
011	married

edu_key	edu
010	4-years
100	master

CUBOID KEY CONSTRUCTION

CUBE DATA STORAGE

Table

Region

ColumnFamily

Row

Column

Version

Value

Cuboid Cell

One table for dimension instances storage:	
Row Key	Dimension Name
Column Family	Default
Column	Member BitKey
Value	Member Value
Multiple tables for cuboids instances	
Table Name	Cuboid Name
Row Key	Cell BitKey
Column Family	Default
Column	Measure Name
Value	Measure Value

CUBE DATA STORAGE FOR EXAMPLE

Table: Dimension

Row Key	Column Family: default			
Dimension A	Mask	000001	001000	001001
	001001	A_1	A_2	A_3
Dimension B	Mask	000010	100000	
	100010	B_1	B_2	

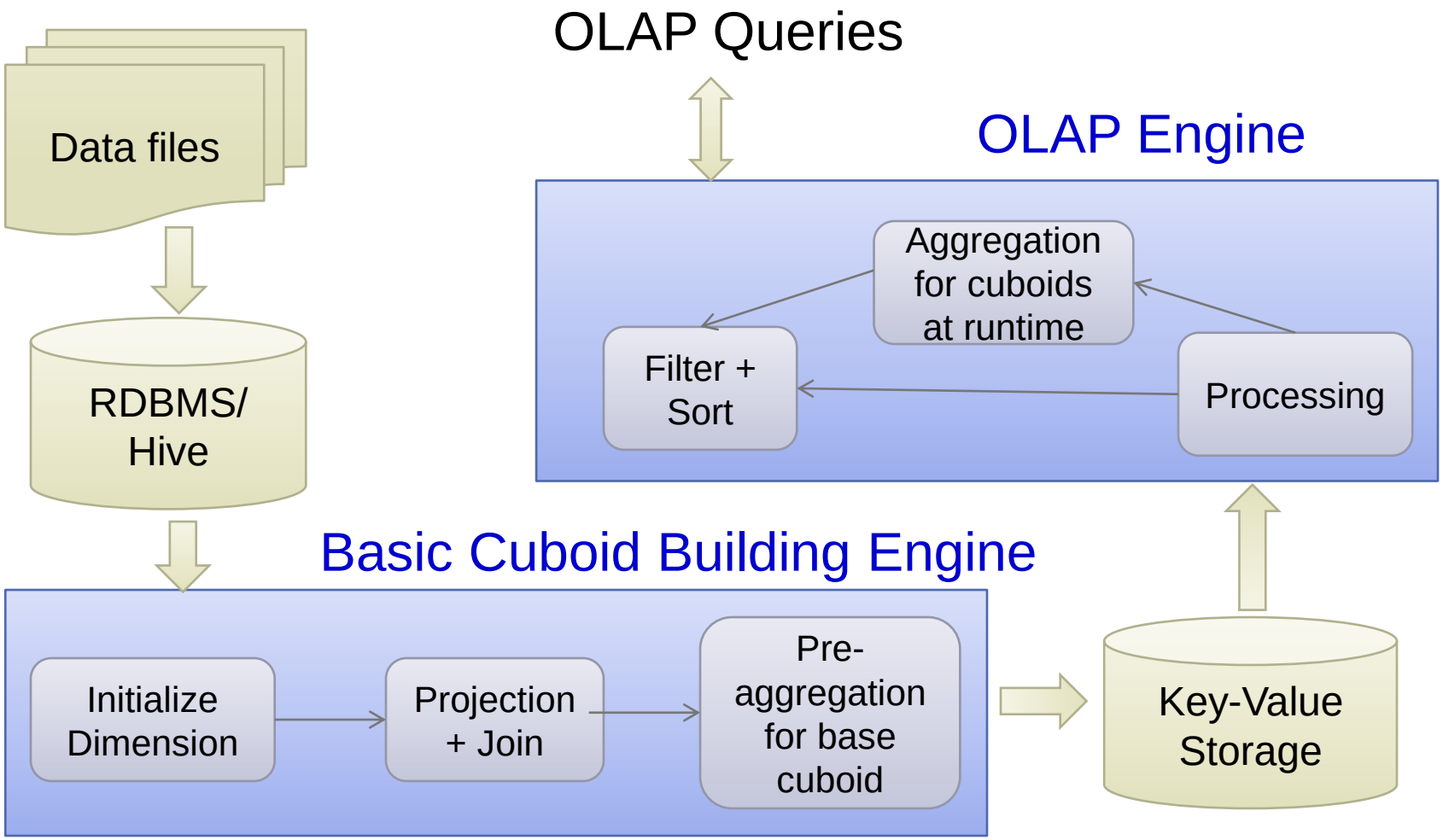
Table: Cuboid_ABC

Row Key	Column Family: default		
000111	Mea_count	Mea_sum	
	1	M_1	
011010	Mea_count	Mea_sum	
	1	M_2	

OUTLINE

- Motivation
- Methodology for MOLAP
- ➢ **Description for MOALP engine**
- Experimenting
- Conclusion

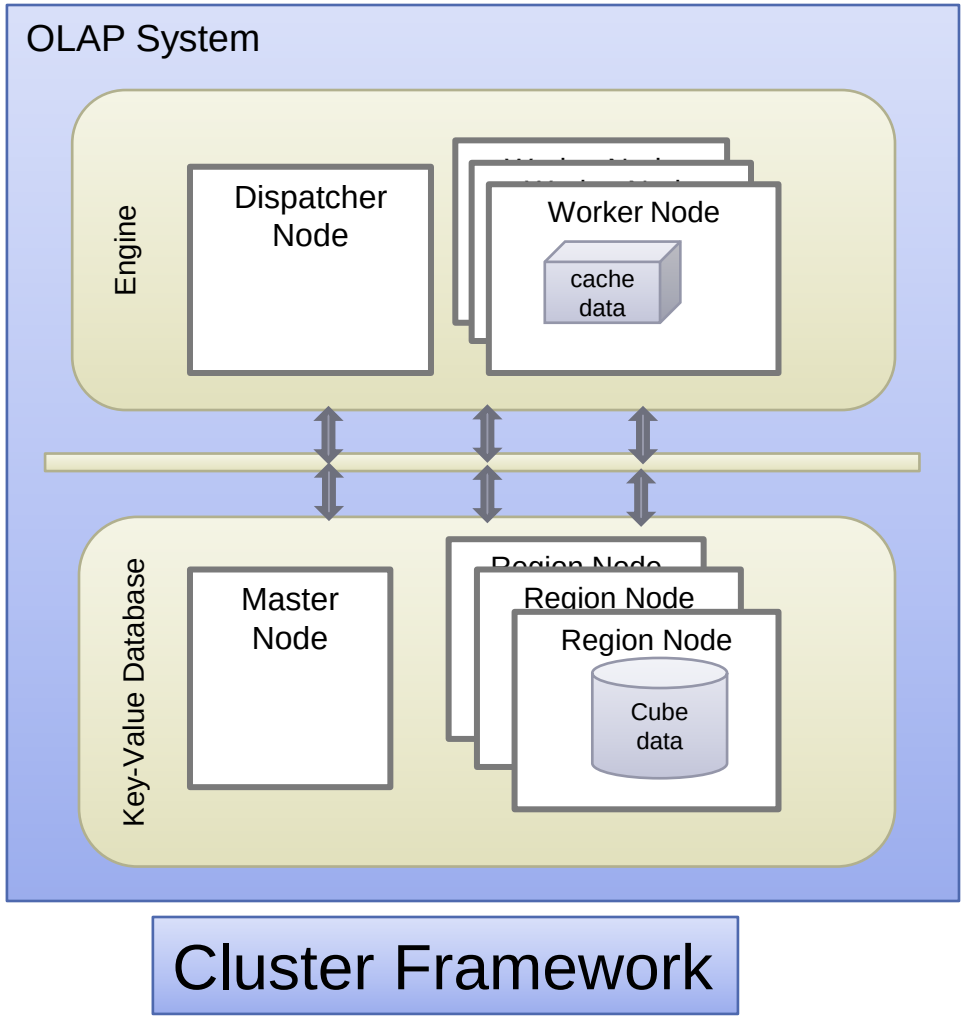
ARCHITECTURE OF PROTOTYPE



ARCHITECTURE OF PROTOTYPE

- Dispatcher Node
- Worker Nodes

- Distribute dynamically cubes data onto worker nodes
- Parallelize OLAP operations into a concurrent model



IMPLEMENTATION STEPS

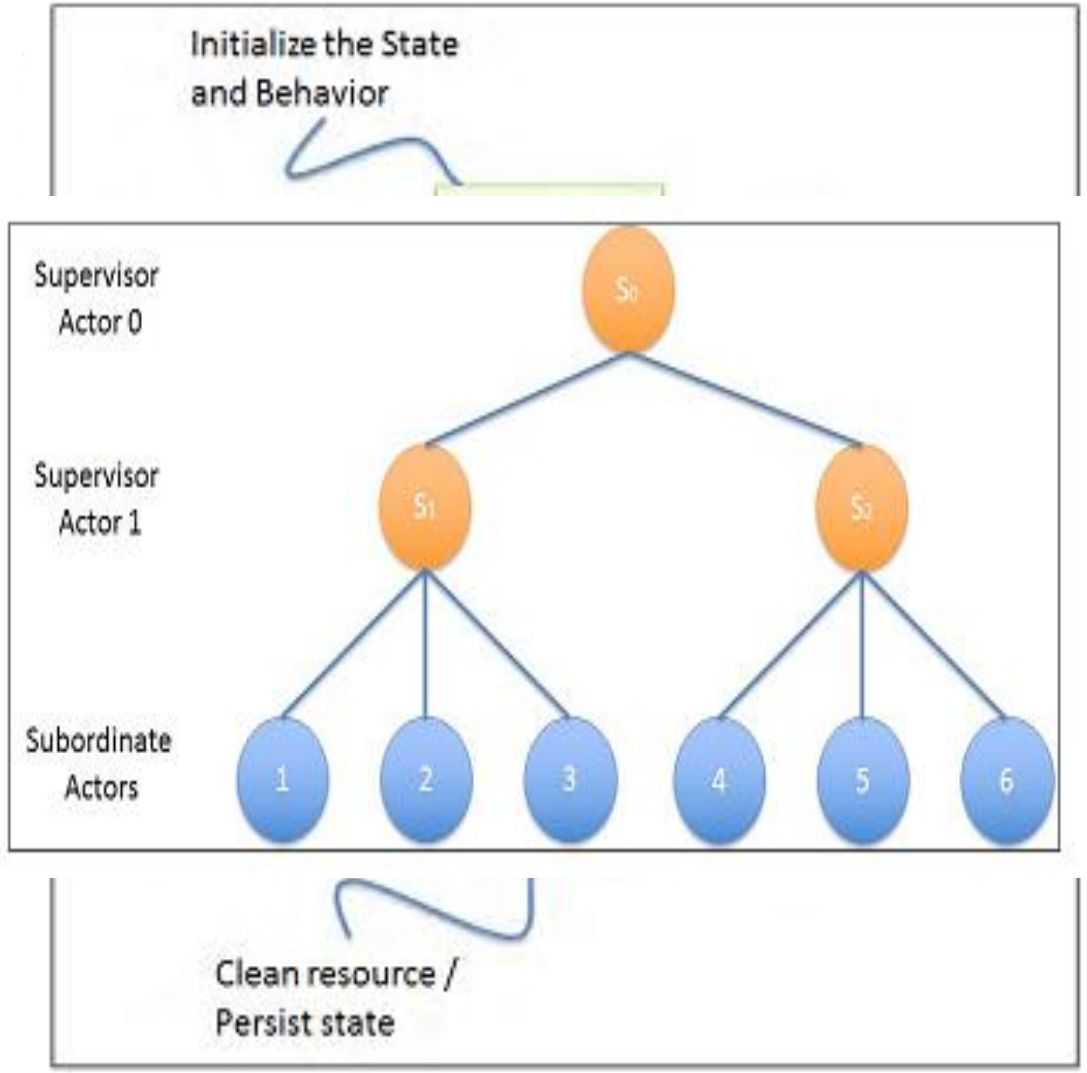
- Base cuboid building with 4 stages:
  - Dimension constructing
  - Hive query
  - Aggregation
  - Saving
- OLAP Query execution with 4 stages:
  - Loading dimension
  - Other cuboid constructing
  - Mapping
  - Reducing

ACTORS OF AKKA FRAMEWORK

State
Behavior
Mailbox

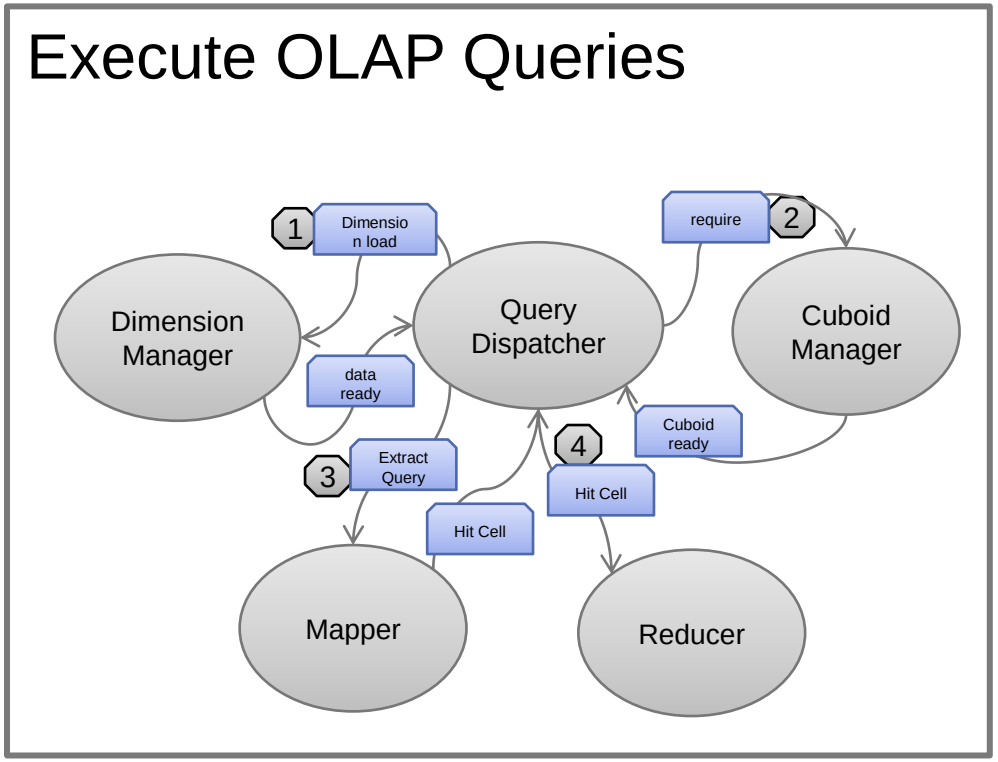
Lifecycle

Fault tolerance

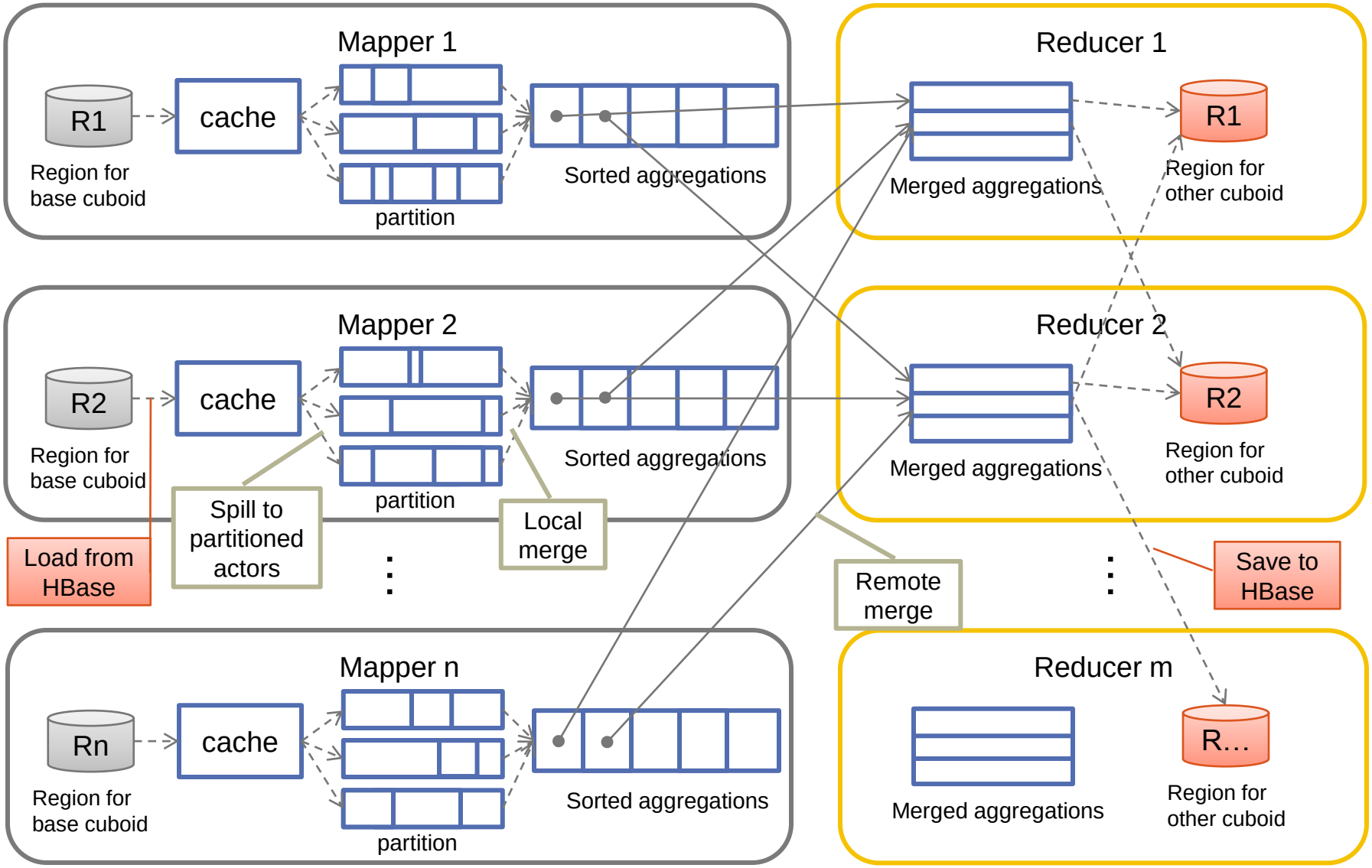


ACTORS FOR OLAP QUERIES

- Load dimension members
- Build other cuboids
- Mapping
- Reducing



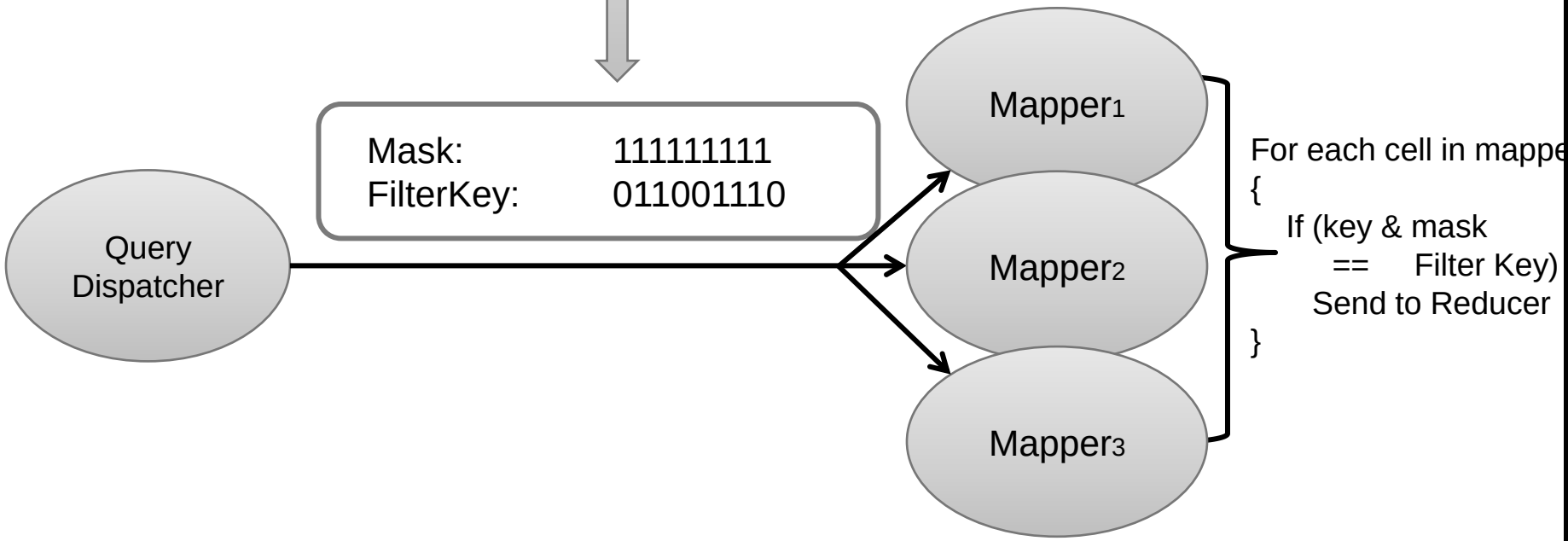
DATA FLOW FOR OTHER CUBOID



COMPILING & MAPPING

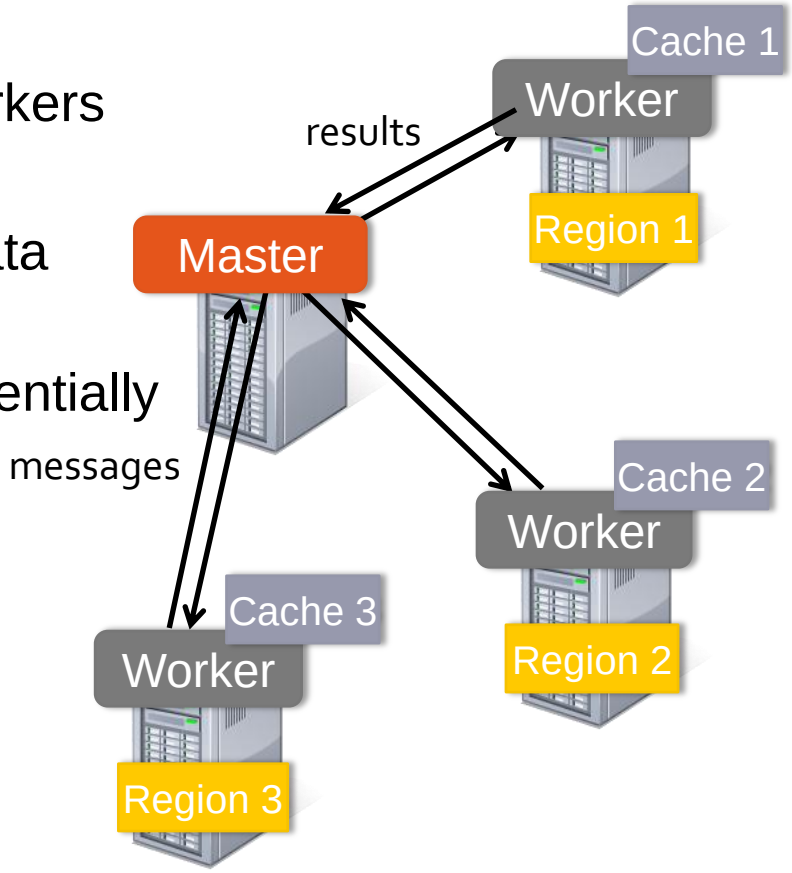
Query 7 Condition:
GEN=M and MS=S and ES=College and YEAR=2000

GEN Mask:	**000000011**	**Male**	**000000010**
MS Mask:	**000011100**	**Single :**	**000001100**
ES Mask:	**001100000**	**College:**	**001000000**
YEAR Mask:	**110000000**	**2000:**	**010000000**



OLAP QUERY EXECUTION

- Master sends task messages to workers
- Each worker caches local region data
- Queries reuse the cache data sequentially



Matei Zaharia: **Overview of Spark**

OUTLINE

- Motivation
- Methodology for MOLAP
- Description for MOALP engine
- ➢ **Experimenting**
- Conclusion

EXPERIMENTS ON TPC-DS

4 Queries:

- Query 7
- Query 42
- Query 52
- Query 55

3 nodes:

- 2*Intel Xeon CPU E5-2630
- 4*600G 15000r/s SAS Raid 1+0
- 256G RAM
- 10Gb Network

Dimensions:

1. "i_item_id",
2. "i_category“
3. "i_manager_id“
4. "i_brand",
5. "cd_gender",
6. "cd_marital_status",
7. "cd_education_status",
8. "p_channel_email",
9. "p_channel_event",
10. "d_year“
11. “d_moy”

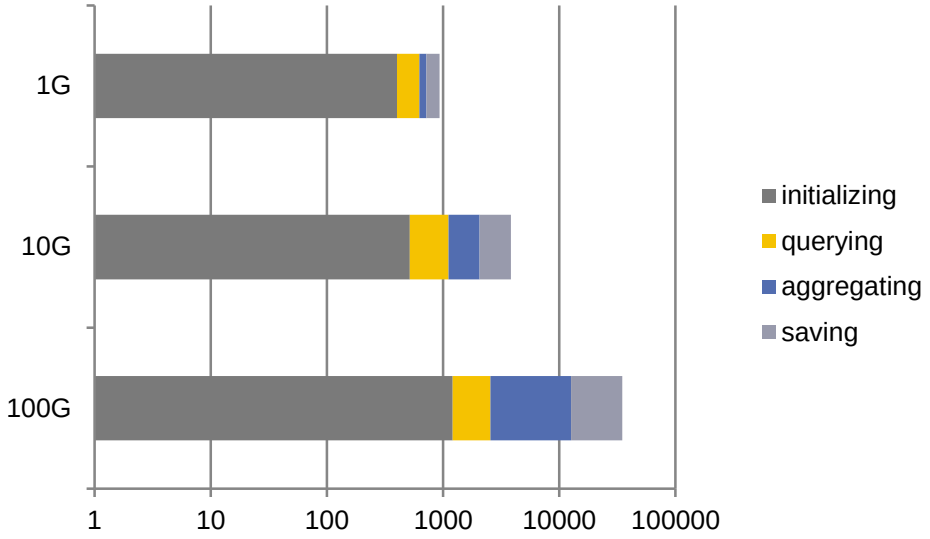
Measures:

ss_quantity, ss_list_price, ss_coupon_amt, ss_sales_price, ss_ext_sales_price

	1G	10G	100G
records number	2,653,108	26,532,571	265,325,821
cube cell number	2,543,842	24,639,263	189,298,704
Storage In HBase	4*64M	64*64M	256*64M

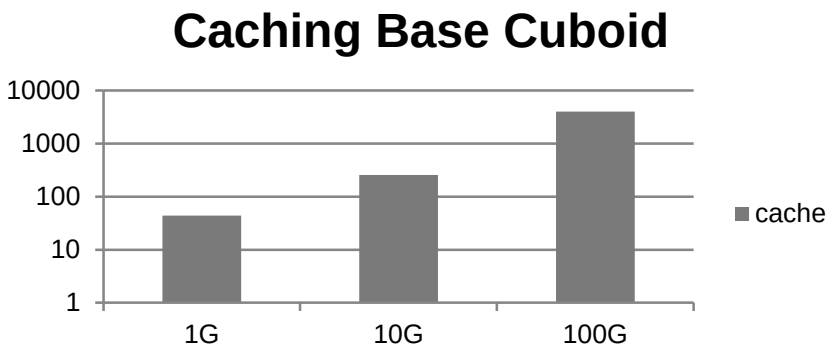
BUILD CUBE FOR QUERIES

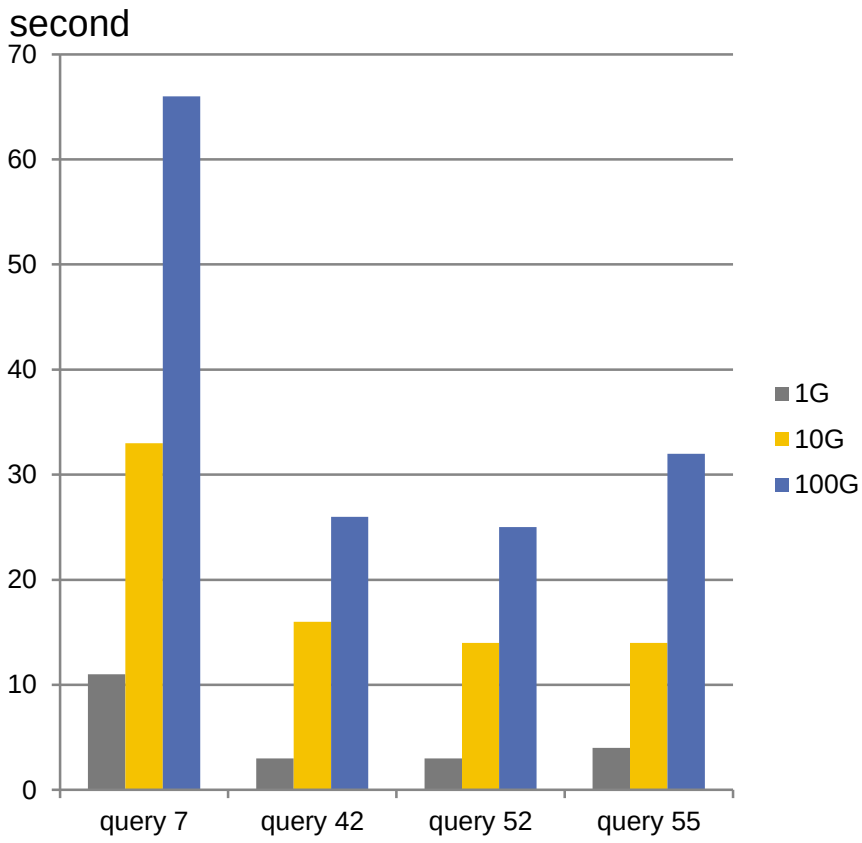
- Partition by the largest dimension(i_item_id)
- In-memory aggregation
- Saving stage can be ignore(cache)
- Logarithmic scale



EXECUTE QUERY 7, 42, 52, 55

- Stages for first query executing
- Dimension loading
- Caching
- Mapping
- Reducing



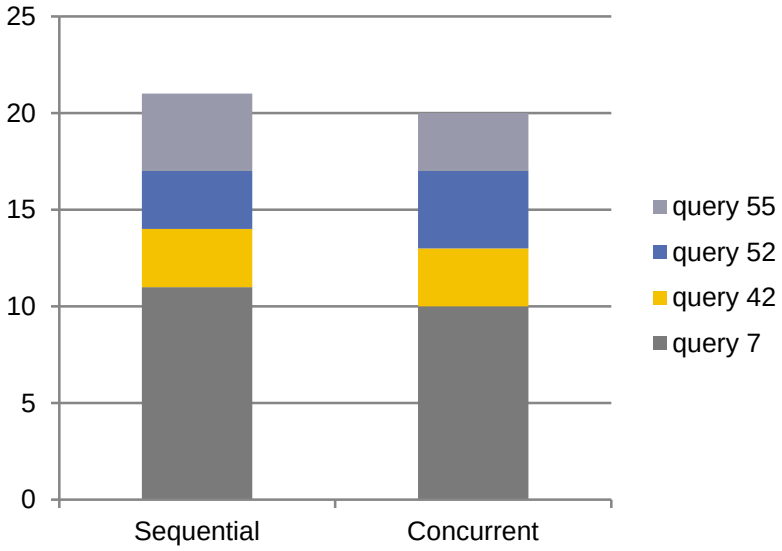


- Stages for later queries executing:
- Mapping
- Reducing

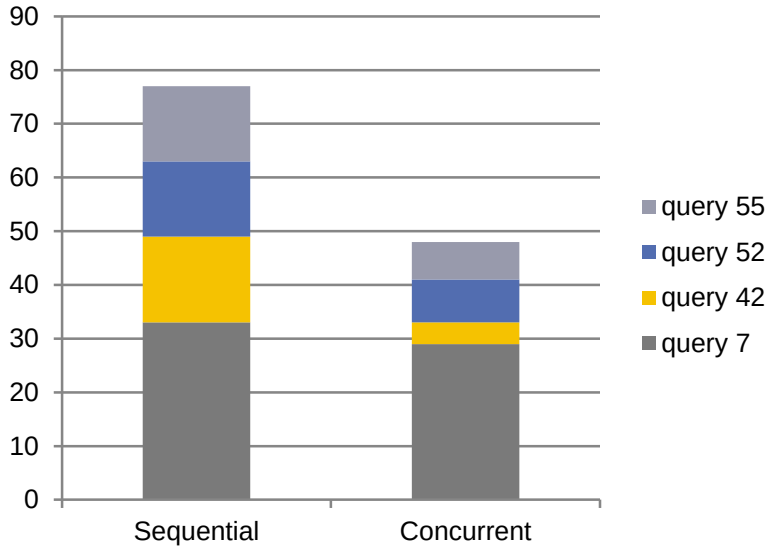
CONCURRENT QUERIES EXECUTING

- Sequence VS concurrency

Results on 1G data



Results on 10G data



COMPARE WITH ROLAP

execution time in out-of-the-box setting:

14-56X

	1G	10G	100G
query 7	14X	24X	19X
query 42	53X	49X	48X
query 52	53X	56X	50X
query 55	40X	56X	39X

CONCLUSIONS

A MOLAP prototype on NoSQL databases:

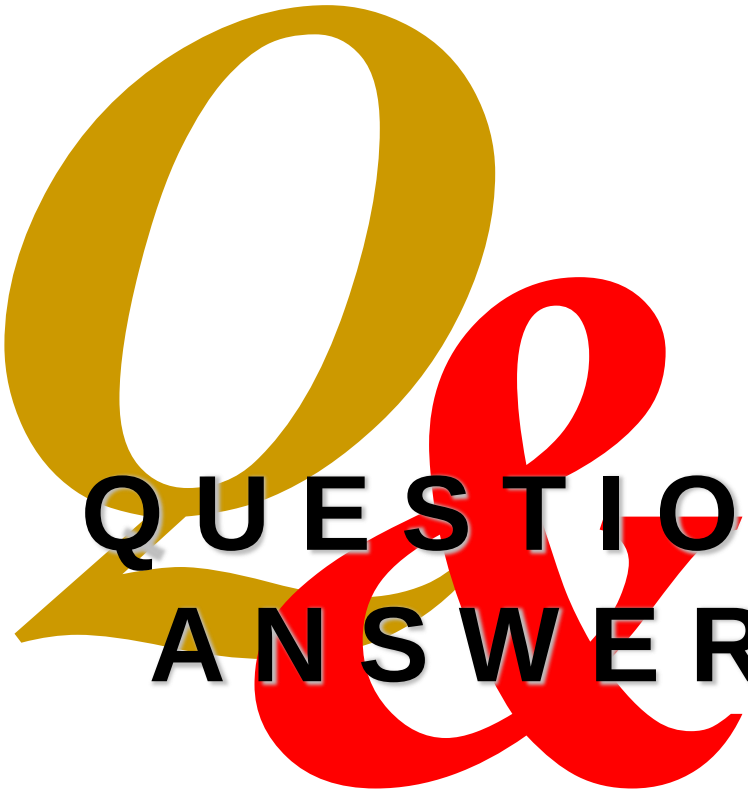
- Basic OLAP operation implementation
- Some queries experiments and analysis

➢ **Other experiments on TPC-DS queries**

- Report, ad hoc, iterative, data mining, … …

More work on multidimensional benchmarking

- **Choice of cube model :**
  - Demand-driven & data-driven
- **Generation for cube data:**
  - Model-driven & requirement-driven



QUESTIONS ANSWERS