facebook

facebook

Join Strategies in Hive

Liyin Tang, Namit Jain

Software Engineer

Facebook

Agenda

1 Common Join

2 Map Join

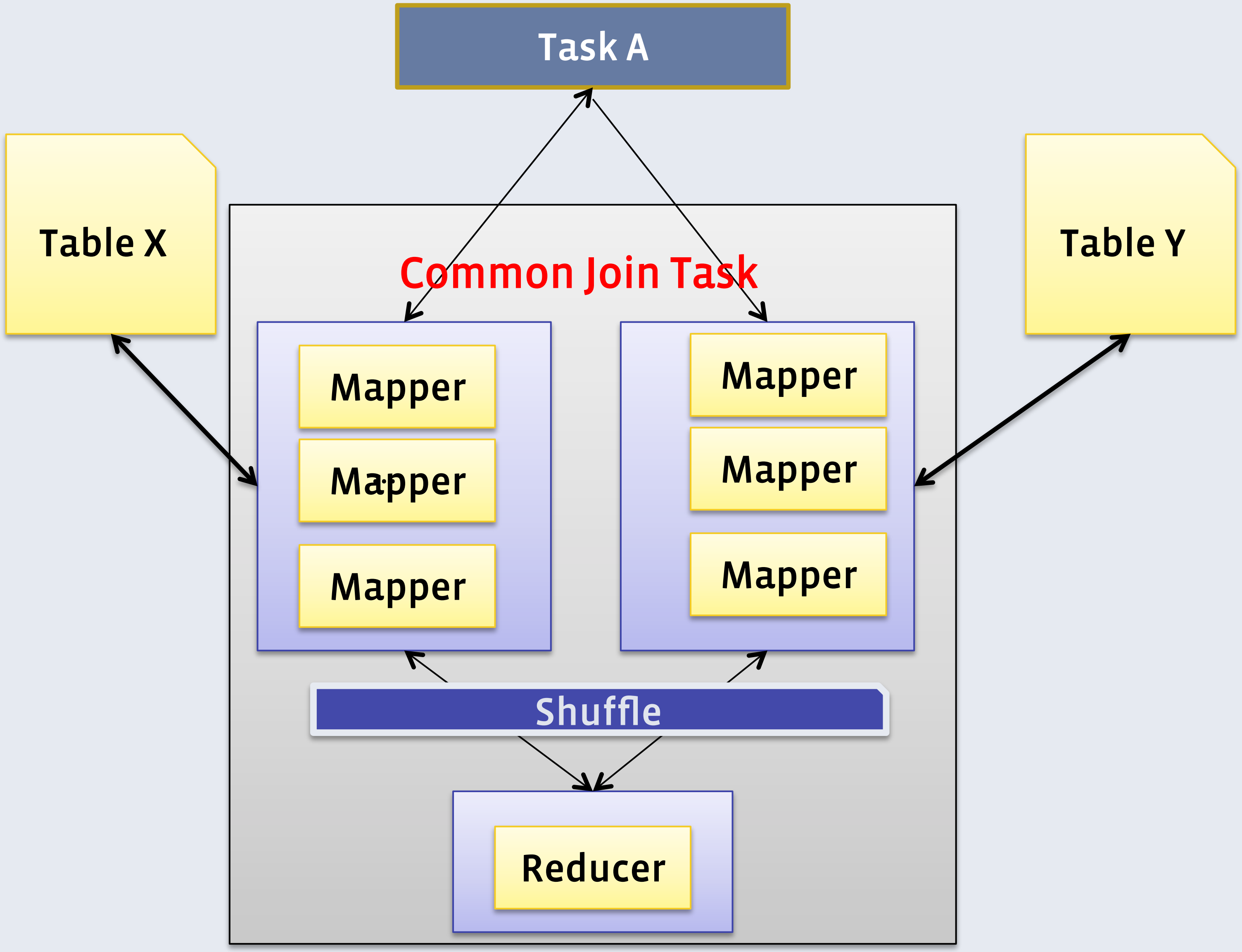
3 Auto MapJoin

4 Bucket Map Join

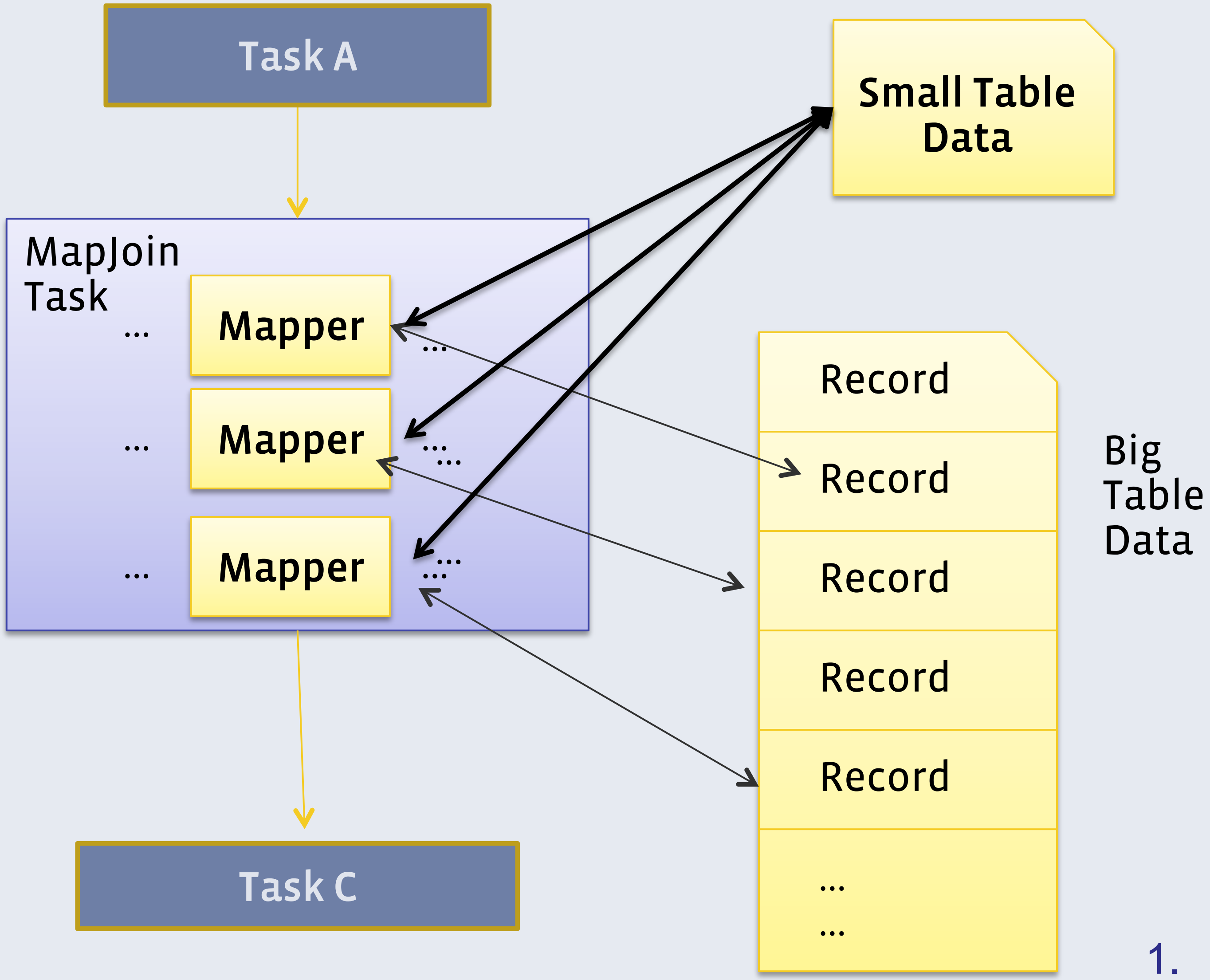
5 Bucket Sort Merge Map Join

6 Skew Join

Common Join

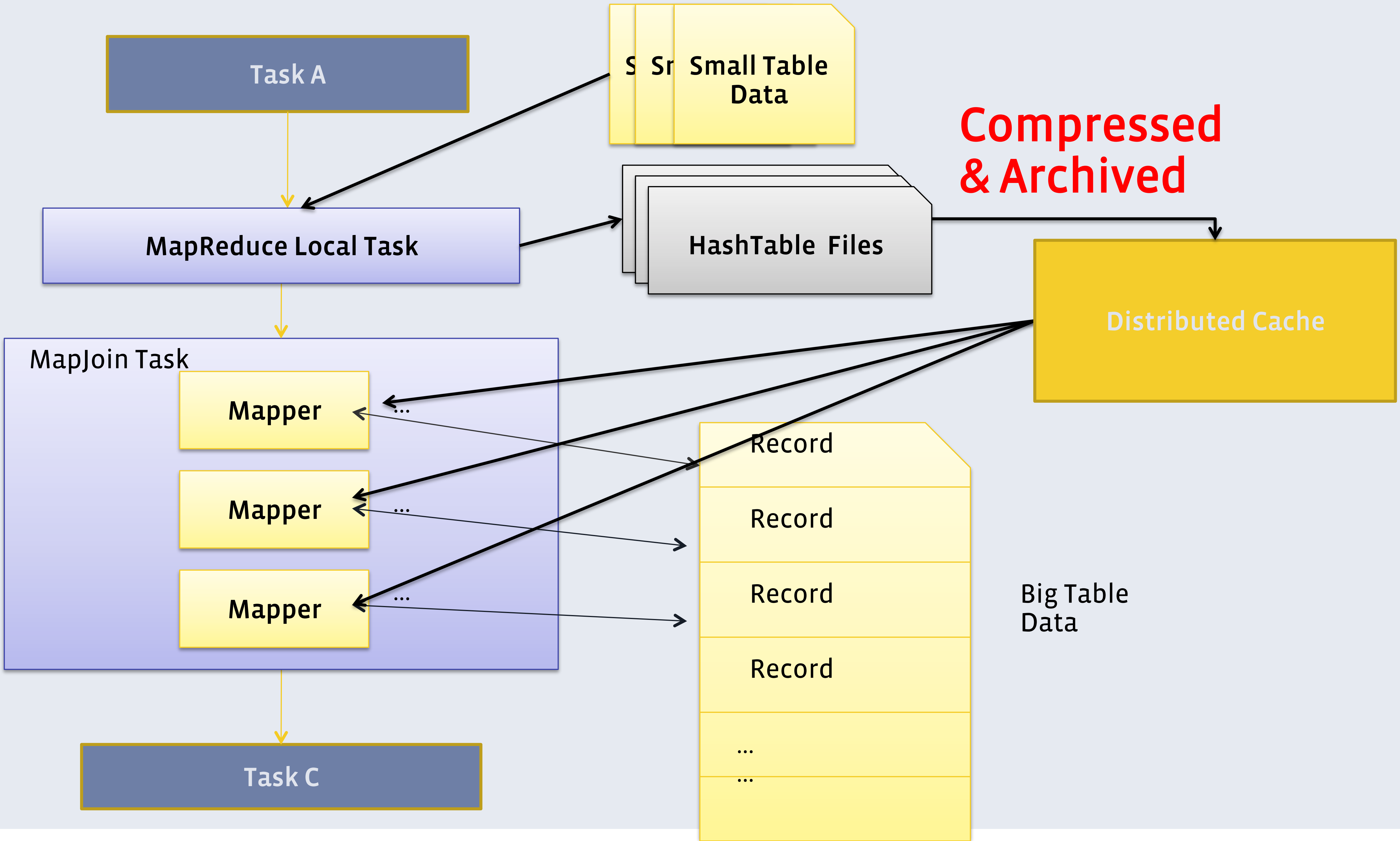

Task A
Table X
Common Join Task
Table Y
Mapper
Mapper
Mapper
Mapper
Mapper
Mapper
Shuffle
Reducer

MapJoin

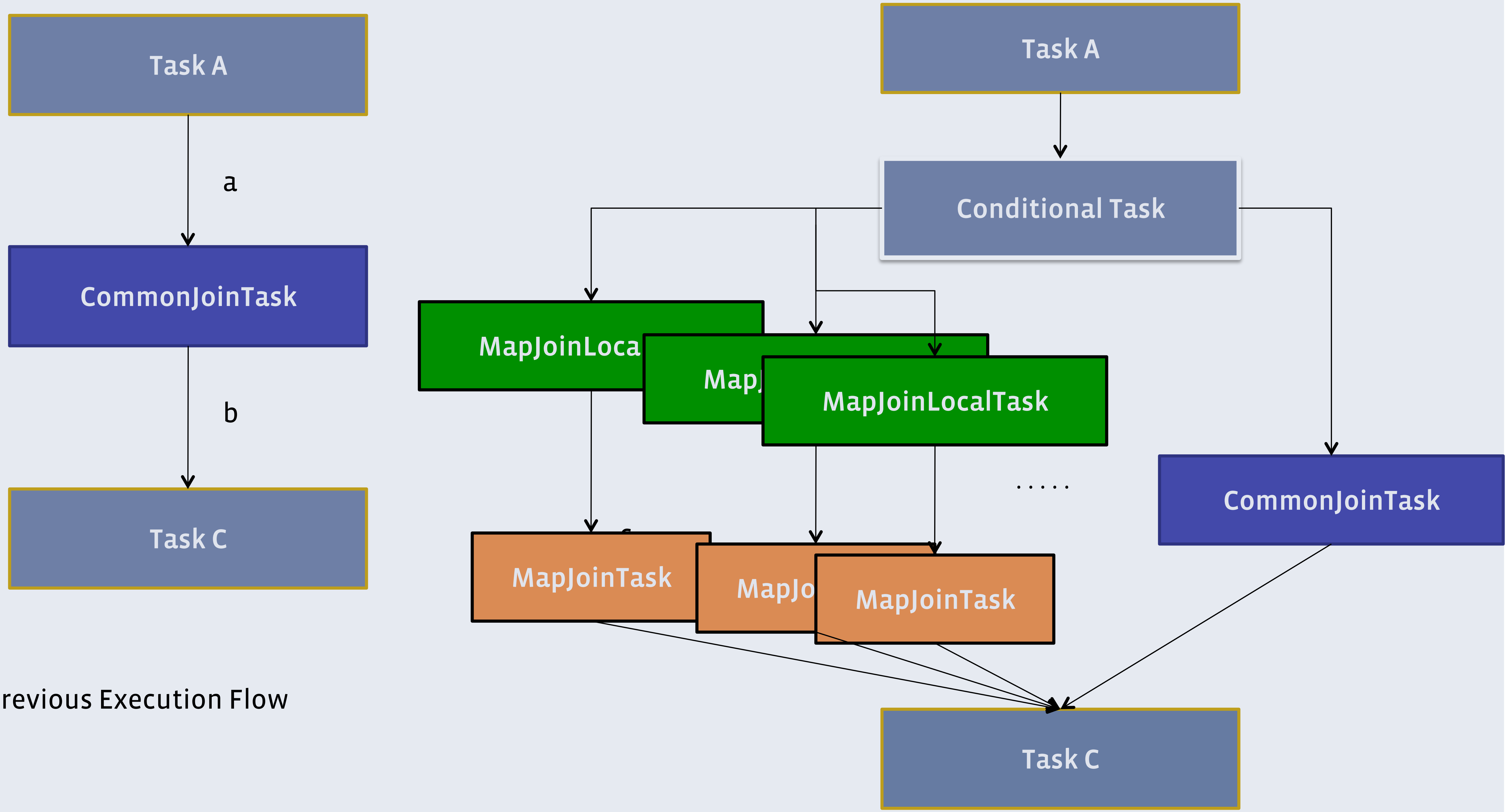


1. Spawn mapper based on the big table
2. All files of all small tables are replicated onto each mapper

Optimized Map Join



Converting Common Join into Map Join

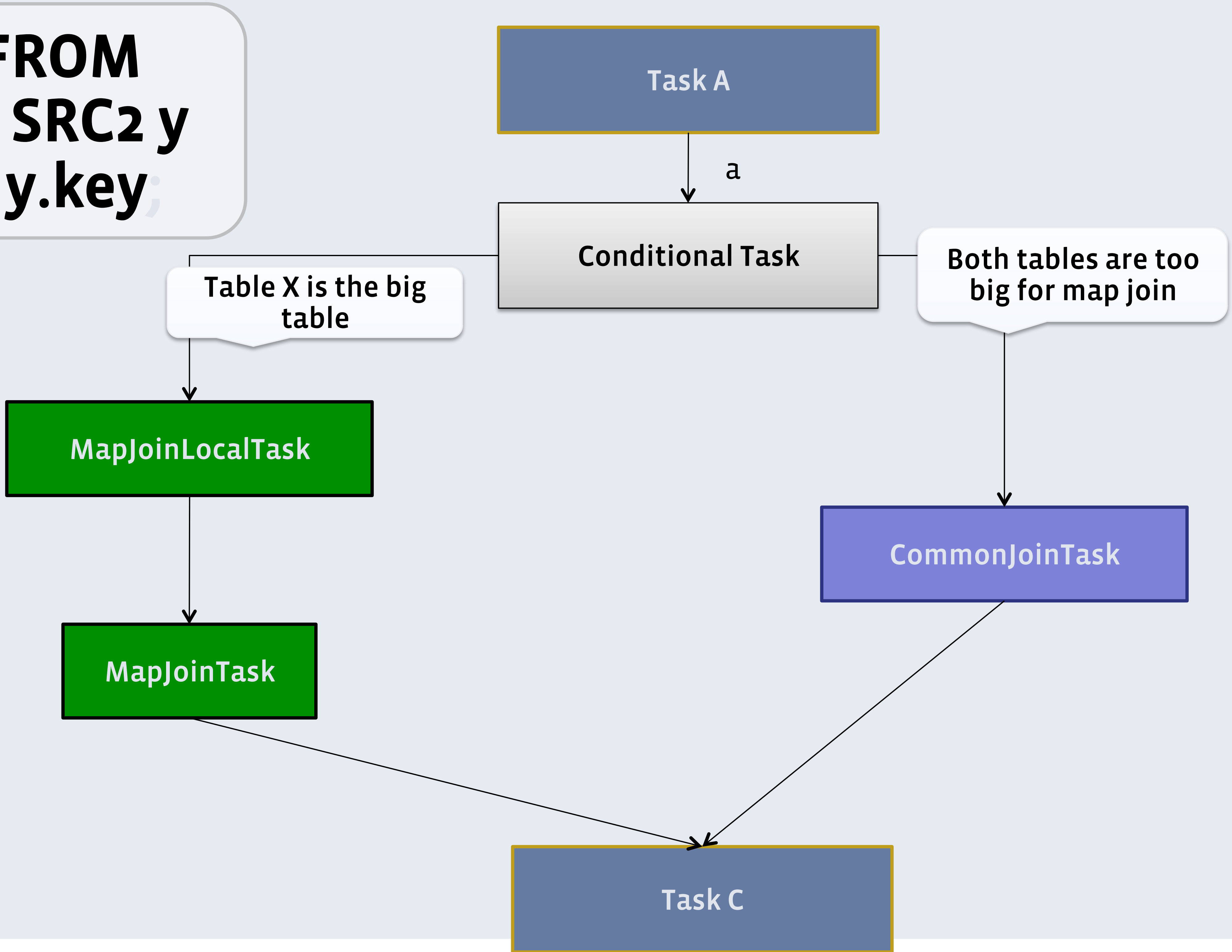


Previous Execution Flow

Optimized Execution Flow

Execution Time

```
SELECT * FROM
SRC1 x JOIN SRC2 y
ON x.key = y.key;
```



Task A

a

Conditional Task

Table X is the big table

Both tables are too big for map join

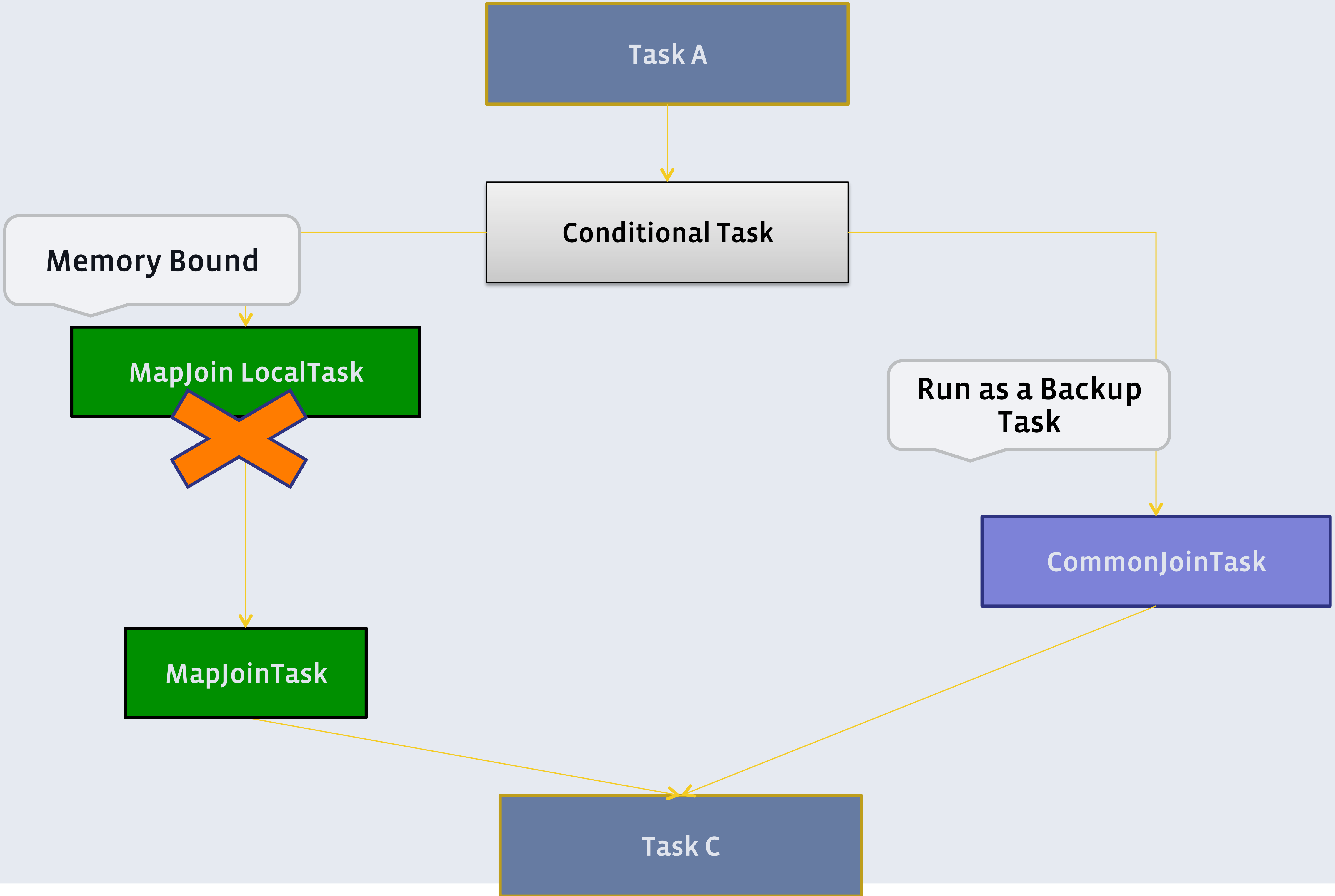
MapJoinLocalTask

MapJoinTask

CommonJoinTask

Task C

Backup Task



Task A

Conditional Task

Memory Bound

MapJoin LocalTask

MapJoinTask

Run as a Backup Task

CommonJoinTask

Task C

Performance Bottleneck

Distributed Cache is the potential performance bottleneck

- Large hashtable file will slow down the propagation of Distributed Cache
- Mappers are waiting for the hashtables file from Distributed Cache

Compress and archive all the hashtable file into a tar file.

Bucket Map Join

Why:

Total table/partition size is big, not good for mapjoin.

How:

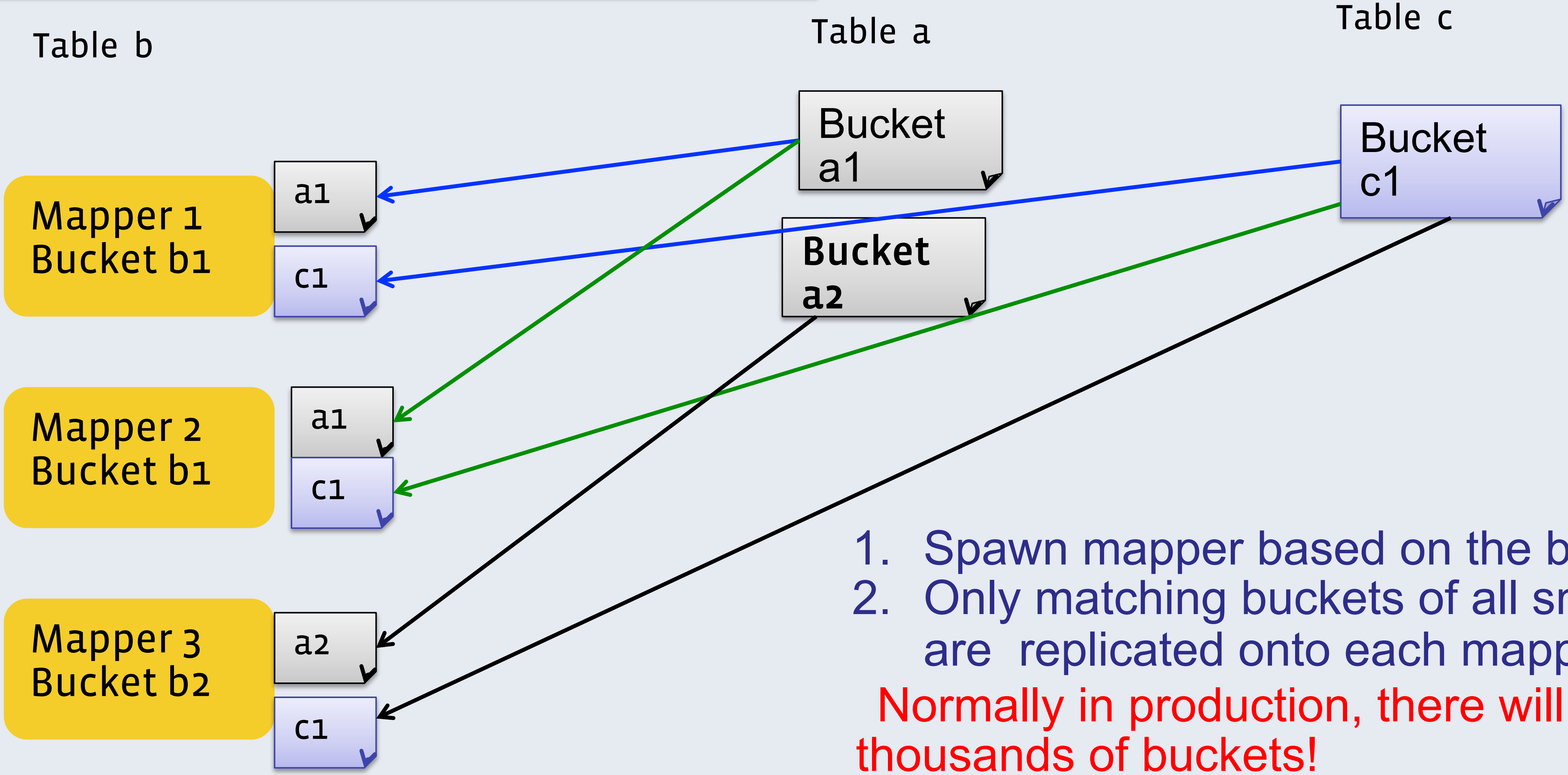
set hive.optimize.bucketmapjoin = true;

1. Work together with map join
2. All join tables are bucketized, and each small table’s bucket number can be divided by big table’s bucket number.
3. Bucket columns == Join columns

Bucket Map Join

SELECT /*+MAPJOIN(a,c)*/ a.*, b.*, c.*
a join b on a.key = b.key
join c on a.key=c.key;

Table a,b,c all bucketized by ‘key’
a has 2 buckets, b has 2, and c has 1



1. Spawn mapper based on the big table
2. Only matching buckets of all small tables are replicated onto each mapper

Normally in production, there will be thousands of buckets!

Sort Merge Bucket Map Join

Why:

No limit on file/partition/table size.

How:

```
set hive.optimize.bucketmapjoin = true;
set hive.optimize.bucketmapjoin.sortedmerge = true;
set hive.input.format=org.apache.hadoop.hive.ql.io.BucketizedHiveInputFormat;
```

1.Work together with bucket map join

2.Bucket columns == Join columns == sort columns

Sort Merge Bucket Map Join

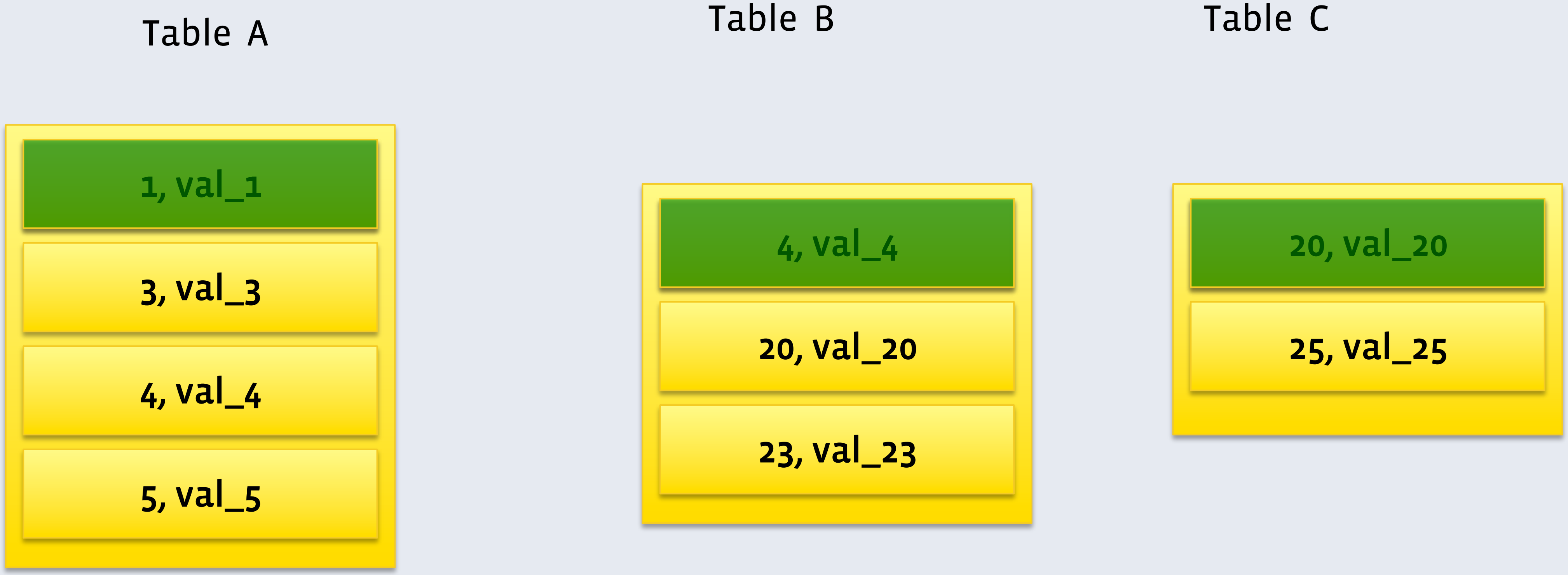


Table A

1, val_1

3, val_3

4, val_4

5, val_5

Table B

4, val_4

20, val_20

23, val_23

Table C

20, val_20

25, val_25

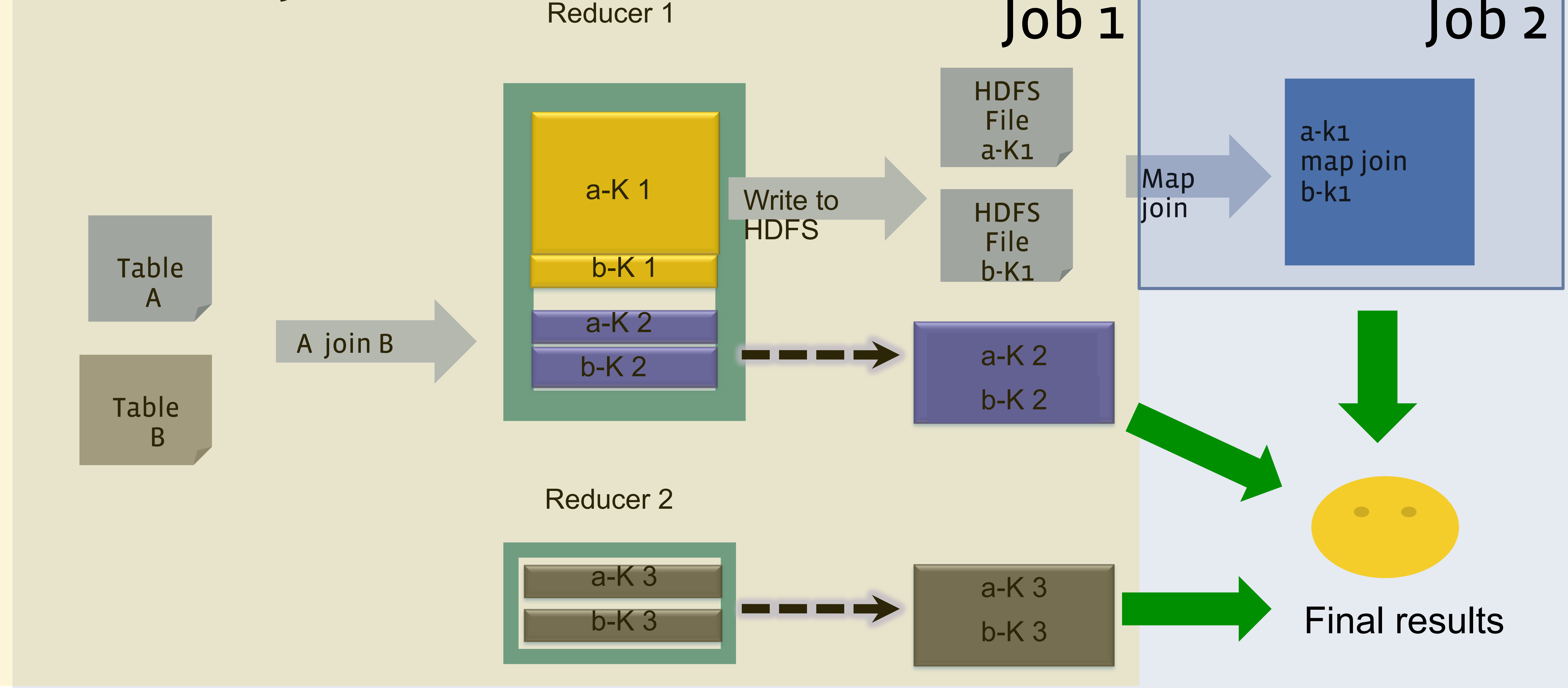
Small tables are read on demand
NOT hold entire small tables in memory
Can perform outer join

Skew Join

Join bottlenecked on the reducer who gets the skewed key

```
set hive.optimize.skewjoin = true;
set hive.skewjoin.key = skew_key_threshold
set hive.optimize.skewjoin = true;
set hive.skewjoin.key = Skew_Key_Threshold;
```

Skew Join



Clusters

2000 live nodes cluster

Commodity machines

- CPU: 2 Intel@Xeon X5650
- Memory: 48G
- Disk: 2 TP*12 Disks
- CentOS

Performance Evaluation I

Small Table	Big Table	Join Condition	Average Map Join Execution time	Average New Optimized Map Join Execution time	Performance Improvement

75 K rows;
383K file size
	130 M rows;
	3.5G file size;
		1 join key,
		2 join value
			1032 sec	79 sec	+ 1206%

500 K rows; 2.6M file size	130 M rows; 3.5G file size	1 join key, 2 join value	3991 sec	144 sec	+2671 %
75 K rows; 383K file size	16.7 B rows; 459 G file size	1 join key, 2 join value	4801 sec	325 sec	+ 1377 %

Performance Evaluation II

Small Table	Big Table	Join Condition	Average Join Execution Time Without Compression	Average Join Execution Time With Compression	Performance Improvement

75 K rows;
383K file size
	130 M rows;
	3.5G file size;
		1 join key,
		2 join value
			106 sec	73 sec	+ 45%

500 K rows; 2.6M file size	130 M rows; 3.5G file size	1 join key, 2 join value	129 sec	106 sec	+21 %
75 K rows; 383K file size	16.7 B rows; 459 G file size	1 join key, 2 join value	441 sec	326 sec	+ 35 %
500 K rows; 2.6M file size	16.7 B rows; 459 G file size	1 join key, 2 join value	326 sec	251 sec	+30 %
1M rows; 10M file size	16.7 B rows; 459 G file size	1 join key, 3 join value	495 sec	266sec	+86 %
1M rows; 10M file size	16.7 B rows; 459 G file size	2 join key, 2 join value	425 sec	255 sec	+67%

Performance Evaluation III

Small Table	Big Table	Join Condition	Previous Common Join	Optimized Common Join	Performance Improvement

75 K rows;
383K file size
	130 M rows;
	3.5G file size;
		1 join key,
		2 join value
			169 sec	**79 sec**	**+ 114%**

500 K rows; 2.6M file size	130 M rows; 3.5G file size	1 join key, 2 join value	**246 sec**	**144 sec**	**+71 %**
75 K rows; 383K file size	16.7 B rows; 459 G file size	1 join key, 2 join value	**511 sec**	**325 sec**	**+ 57 %**
500 K rows; 2.6M file size	16.7 B rows; 459 G file size	1 join key, 2 join value	**502 sec**	**305 sec**	**+64 %**
1M rows; 10M file size	16.7 B rows; 459 G file size	1 join key, 3 join value	**653 sec**	**248 sec**	**+163 %**
1M rows; 10M file size	16.7 B rows; 459 G file size	2 join key, 2 join value	**1117sec**	**536 sec**	**+108%**

Summary & Future Work

Mapjoin supported since Hive 0.5

New map join Launched @Facebook since Jan,2011

Set hashtable file replica number based on the number of Mappers

Tune the limit of small table data size by sampling

Memory efficient hashtable