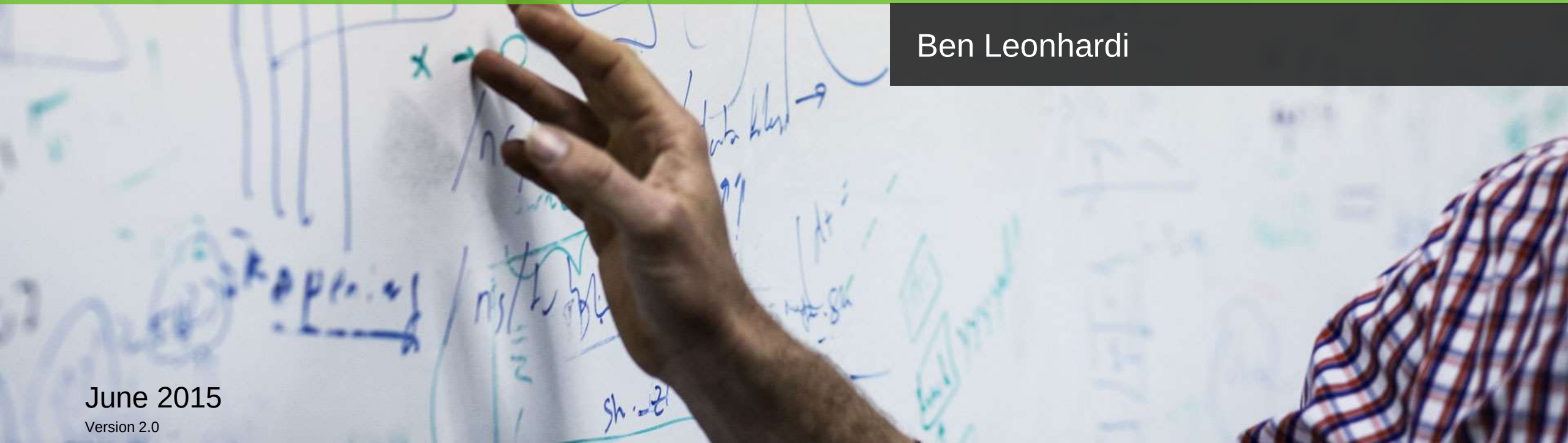
Hortonworks

Hive: Loading Data



Ben Leonhardi

June 2015

Version 2.0

Agenda

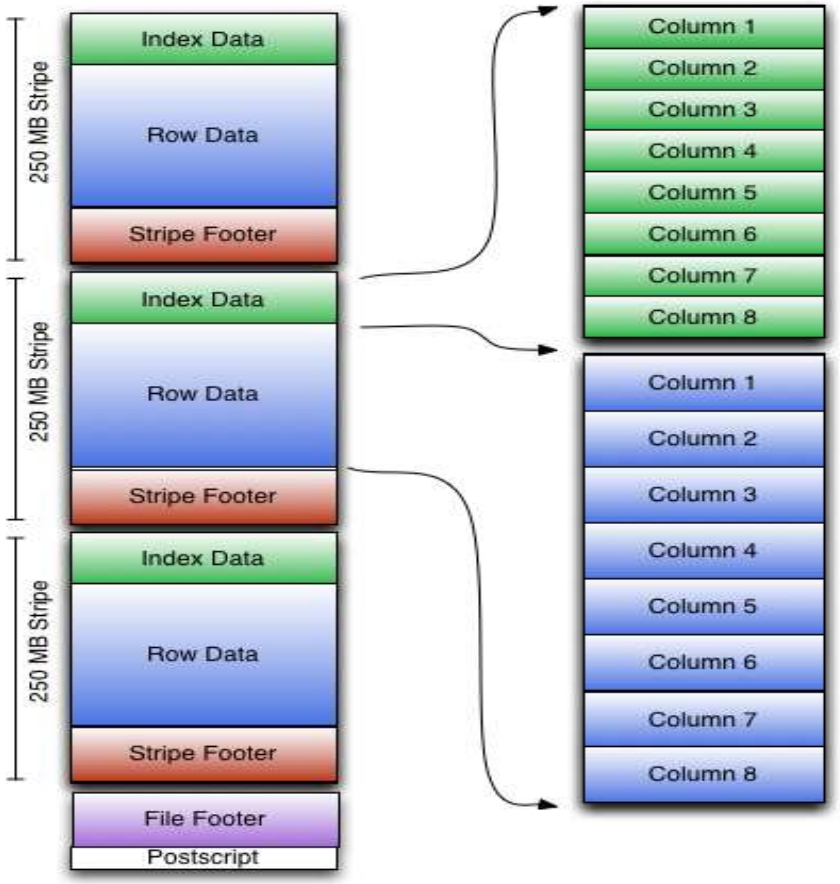
- **Introduction**
  - ORC files
  - Partitioning vs. Predicate Pushdown
- Loading data
  - Dynamic Partitioning
  - Bucketing
  - Optimize Sort Dynamic Partitioning
  - Manual Distribution
- Miscellaneous
  - Sorting and Predicate pushdown
  - Debugging
  - Bloom Filters

Introduction

- **Effectively storing data in Hive**
- **Reducing IO**
  - Partitioning
  - ORC files with predicate pushdown
- **Partitioned tables**
  - Static partition loading
    - One partition is loaded at a time
    - Good for continuous operation
    - Not suitable for initial loads
  - Dynamic partition loading
    - Data is distributed between partitions dynamically
- **Data Sorting for better predicate pushdown**

ORCFile – Columnar Storage for Hive

- **ORC is an optimized, compressed, columnar storage format**
  - Only needed columns are read
  - Blocks of data can be skipped using indexes and predicate pushdown



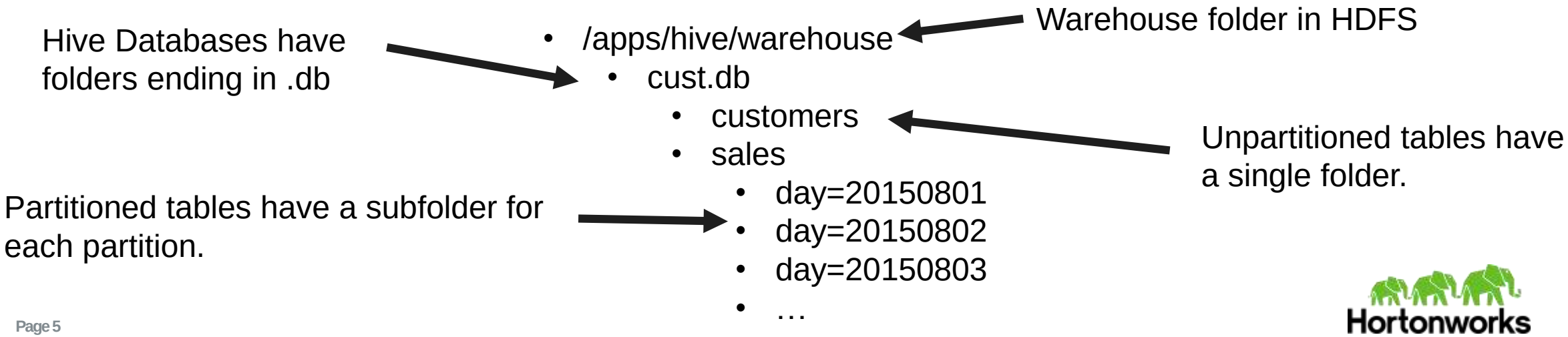
Columnar format enables high compression and high performance.

Hortonworks

Partitioning Hive

- **Hive tables can be value partitioned**
  - Each partition is associated with a folder in HDFS
  - All partitions have an entry in the Hive Catalog
  - The Hive optimizer will parse the query for filter conditions and skip unneeded partitions

- **Usage consideration**
  - Too many partitions can lead to bad performance in the Hive Catalog and Optimizer
  - No range partitioning / no continuous values
  - Normally date partitioned by data load



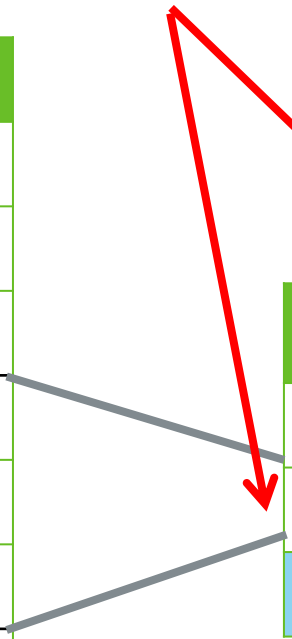
Page 5

Predicate Pushdown

- **ORC (and other storage formats) support predicate pushdown**
  - Query filters are pushed down into the storage handler
  - Blocks of data can be skipped without reading them from HDFS based on ORC index

```
SELECT SUM (PROFIT) FROM SALES WHERE DAY = 03
```

DAY	CUST	PROFIT
01	Klaus	35
01	Max	30
01	John	20
02	John	34
03	Max	10
04	Klaus	20
04	Max	45
05	Mark	20



Only Block 2 can contain rows with DAY 02.
Block 1 and 3 can be skipped

DAY_MIN	DAY_MAX	PROFIT_MIN	PROFIT_MAX
01	01	20	35
02	04	10	34
04	05	20	45

Page 6

Partitioning vs. Predicate Pushdown

- **Both reduce the data that needs to be read**
  - Partitioning works at split generation, no need to start containers
  - Predicate pushdown is applied during file reads

- **Partitioning is applied in the split generation/optimizer**
  - Impact on Optimizer and HCatalog for large number of partitions
  - **Thousands of partitions will result in performance problems**

- **Predicate Pushdown needs to read the file footers**
  - Container are allocated even though they can run very quickly
  - No overhead in Optimizer/Catalog

- **Newest Hive build 1.2 can apply PP at split generation time**
  - hive.exec.orc.split.strategy=BI, means never read footers (& fire jobs fast)
  - hive.exec.orc.split.strategy=ETL, always read footers and split as fine as you want

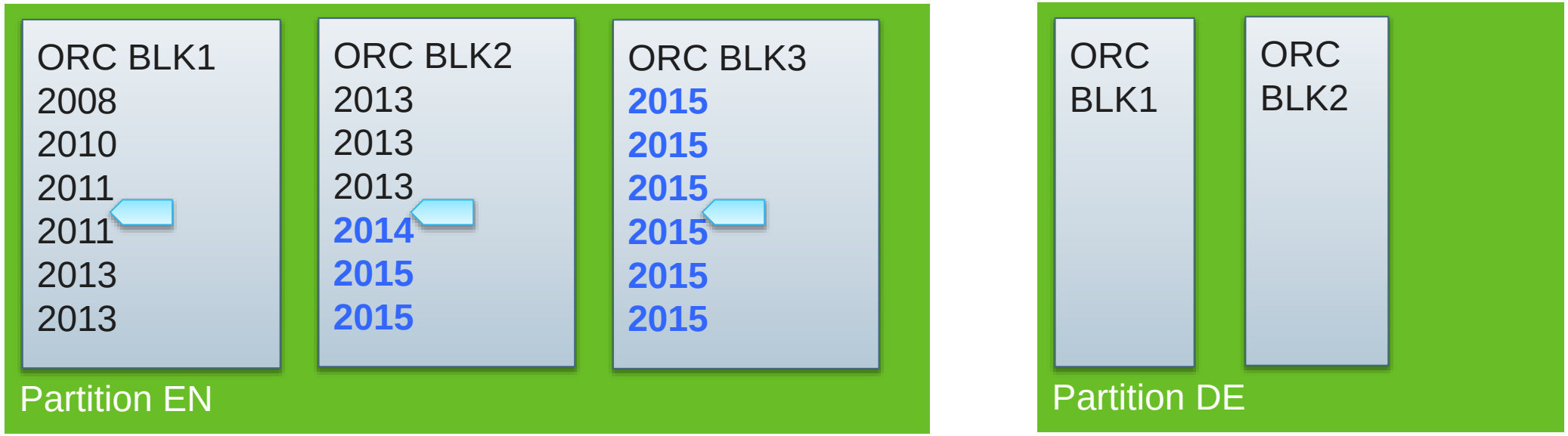
Partitioning and Predicate Pushdown

Table partitioned on Country, only folder for “EN” is read

SELECT * FROM TABLE WHERE COUNTRY = “EN” and DATE = 2015

ORC files keep index information on content, blocks can be skipped based on index

Map1 Map2 Map3



Hortonworks

Agenda

- Introduction
  - ORC files
  - Partitioning vs. Predicate Pushdown
- **Loading data**
  - Dynamic Partitioning
  - Bucketing
  - Optimize Sort Dynamic Partitioning
  - Manual Distribution
- Miscellaneous
  - Sorting and Predicate pushdown
  - Debugging
  - Bloom Filters

Loading Data with Dynamic Partitioning

```
CREATE TABLE ORC_SALES
    ( CLIENTID INT, DT DATE, REV DOUBLE, PROFIT DOUBLE, COMMENT STRING )
    PARTITIONED BY ( COUNTRY STRING )
    STORED AS ORC;
INSERT INTO TABLE ORC_SALES PARTITION (COUNTRY) SELECT * FROM DEL_SALES;
```

Dynamic partition columns need to be the last columns in your dataset
Change order in SELECT list if necessary

- Dynamic partitioning could create millions of partitions for bad partition keys
  - Parameters exist that restrict the creation of dynamic partitions

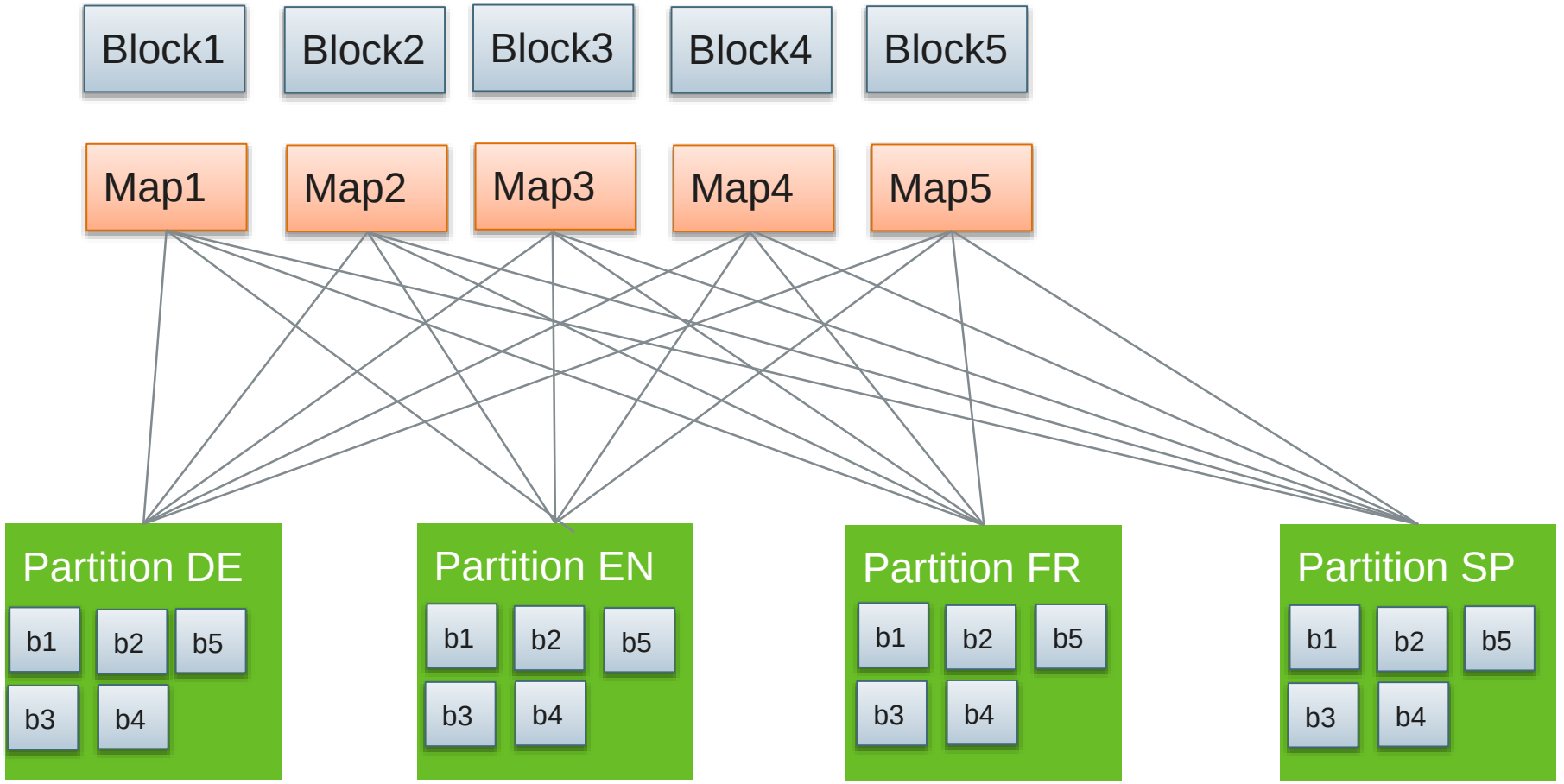
```
set hive.exec.dynamic.partition=true;
set hive.exec.dynamic.partition.mode = nonstrict;
set hive.exec.max.dynamic.partitions.pernode=100000;
set hive.exec.max.dynamic.partitions=100000;
set hive.exec.max.created.files=100000;
```

Most of these settings are already enabled with good values in HDP 2.2+

Hortonworks

Dynamic Partition Loading

- **One file per Reducer/Mapper**
- **Standard Load will use Map tasks to write data. One map task per input block/split**



Hortonworks

Small files

- **Large number of writers with large number of partitions results in small files**
  - Files with 1-10 blocks of data are more efficient for HDFS
  - ORC compression is not very efficient on small files

- **ORC Writer will keep one Writer object open for each partition he encounters.**
  - RAM needed for one stripe in every file / column
  - Too many Writers results in small stripes (down to 5000 rows)

- **If you run into memory problems you can increase the task RAM or increase the ORC memory pool percentage**

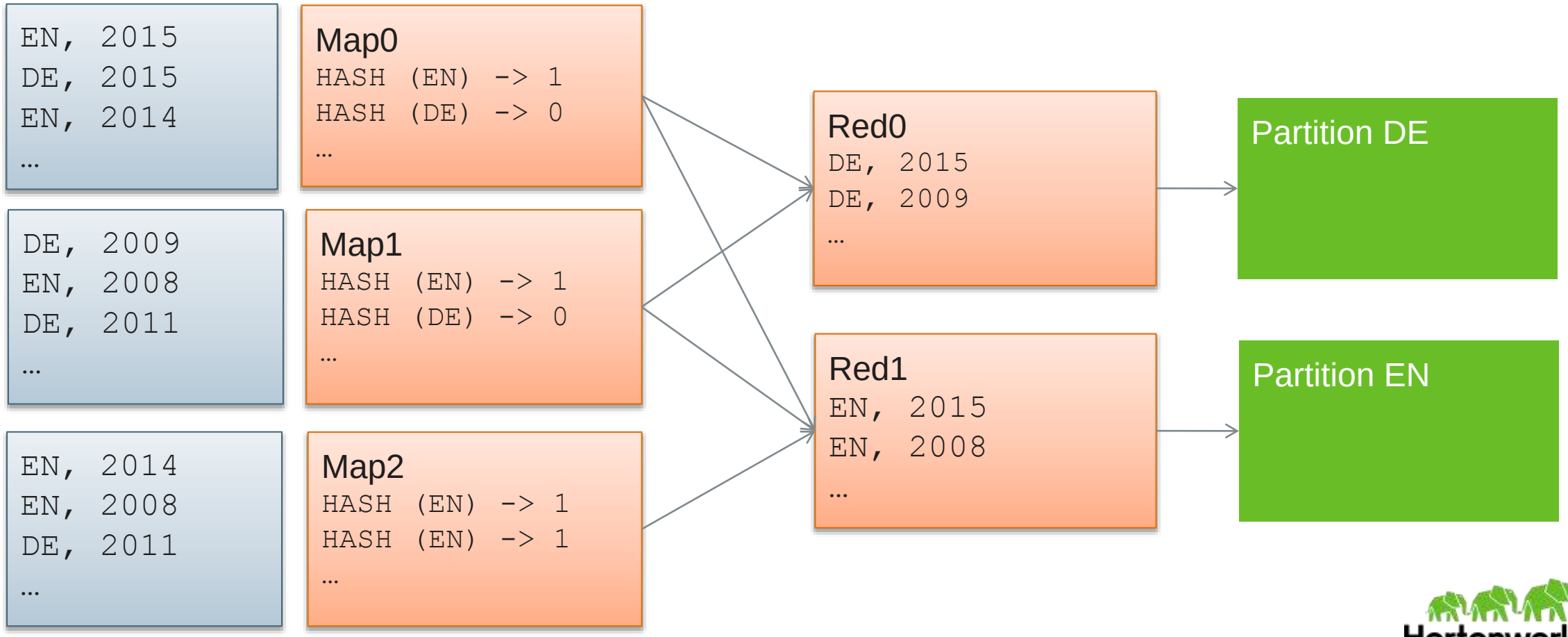
```
set hive.tez.java.opts="-Xmx3400m";

set hive.tez.container.size = 4096;

set  hive.exec.orc.memory.pool = 1.0;
```

Loading Data Using Distribution

- **For large number of partitions, load data through reducers.**
  - One or more reducers associated with a partition through data distribution
  - Beware of Hash conflicts (two partitions being mapped to the same reducer by the hash function)



Hortonworks

Bucketing

- **Hive tables can be bucketed using the `CLUSTERED BY` keyword**
  - One file/reducer per bucket
  - Buckets can be sorted
  - Additional advantages like bucket joins and sampling

- **Per default one reducer for each bucket across all partitions**
  - Performance problems for large loads with dynamic partitioning
  - ORC Writer memory issues

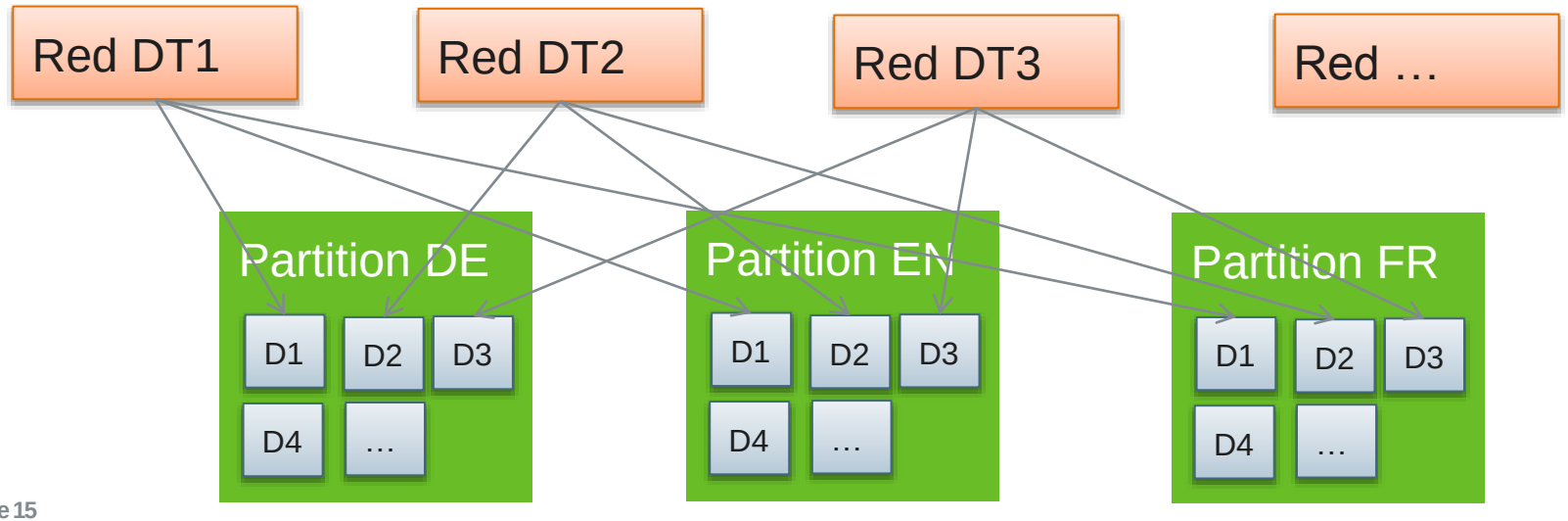
- **Enforce Bucketing and Sorting in Hive**

```
set hive.enforce.sorting=true;

set hive.enforce.bucketing=true;
```

Bucketing Example

```
CREATE TABLE ORC_SALES
    ( CLIENTID INT, DT DATE, REV DOUBLE, PROFIT DOUBLE, COMMENT STRING )
    PARTITIONED BY ( COUNTRY STRING )
    CLUSTERED BY DT SORT BY ( DT ) INTO 31 BUCKETS;
INSERT INTO TABLE ORC_SALES PARTITION (COUNTRY) SELECT * FROM DEL_SALES;
```



Hortonworks

Optimized Dynamic Sorted Partitioning

- **Enable optimized sorted partitioning to fix small file creation**
  - Creates one reducer for each partition AND bucket
  - If you have 5 partitions with 4 buckets you will have 20 reducers

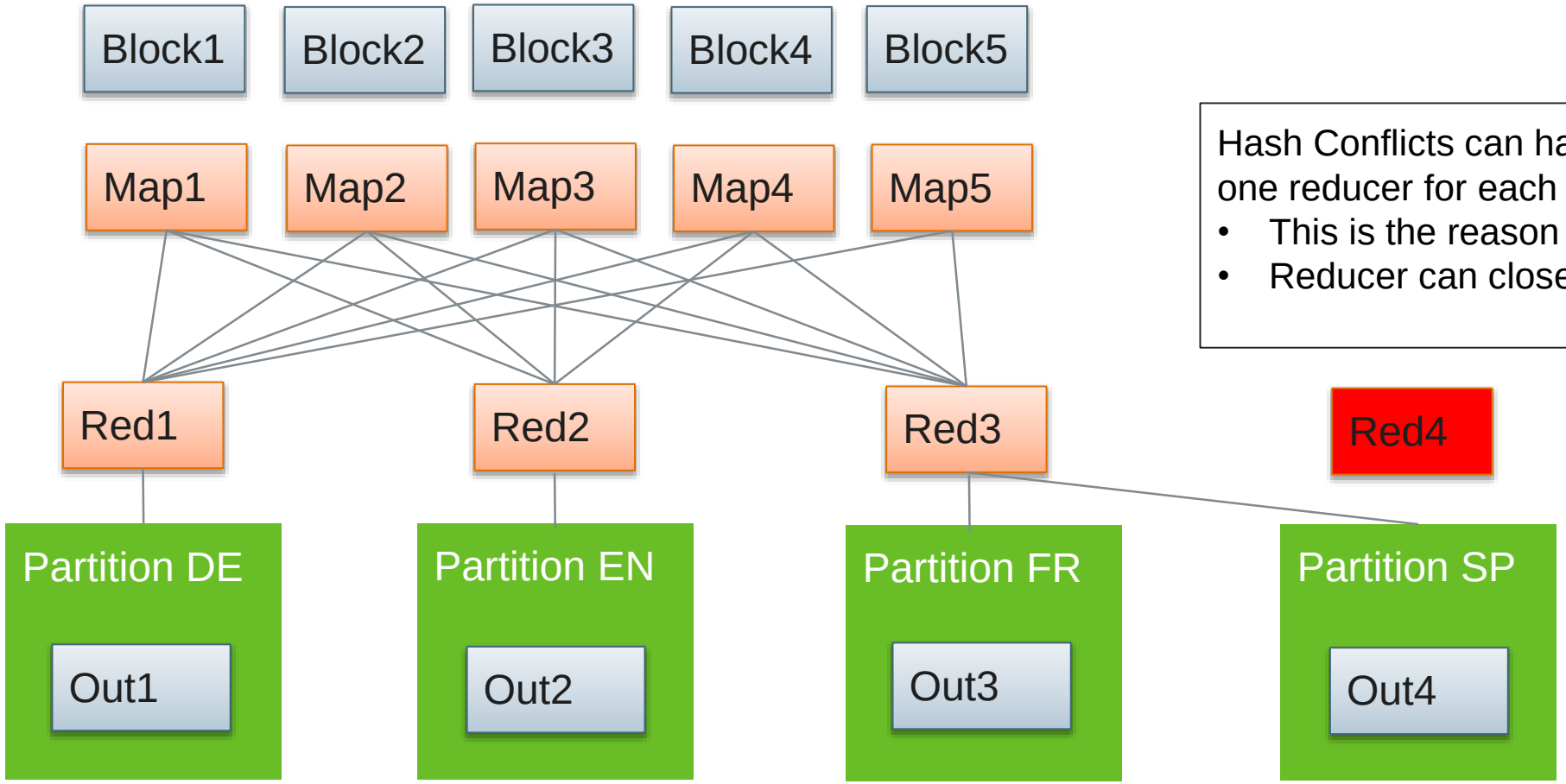
- **Hash conflicts mean that you can still have reducers handling more than one file**
  - Data is sorted by partition/bucket key
  - ORCWriter closes files after encountering new keys
    - only one open file at a time
    - reduced memory needs

- **Can be enabled with**

```
set optimize.sort.dynamic.partitioning=true;
```

Optimized Dynamic Sorted Partitioning

- **Optimized sorted partitioning creates one reducer per partition * bucket**



Hash Conflicts can happen even though there is one reducer for each partition.

- This is the reason data is sorted
- Reducer can close ORC writer after each key

Hortonworks

Miscellaneous

- **Small number of partitions can lead to slow loads**
  - Solution is bucketing, increase the number of reducers
  - This can also help in Predicate pushdown
  - Partition by country, bucket by client id for example.

- **On a big system you may have to increase the max. number of reducers**

```
set hive.exec.reducers.max=1000;
```

Manual Distribution

- **Fine grained control over distribution may be needed**

- **`DISTRIBUTE BY` keyword allows control over the distribution algorithm**
  - For example **`DISTRIBUTE BY GENDER`** will split the data stream into two sub streams
  - Does not define the numbers of reducers
    - Specify a fitting number with

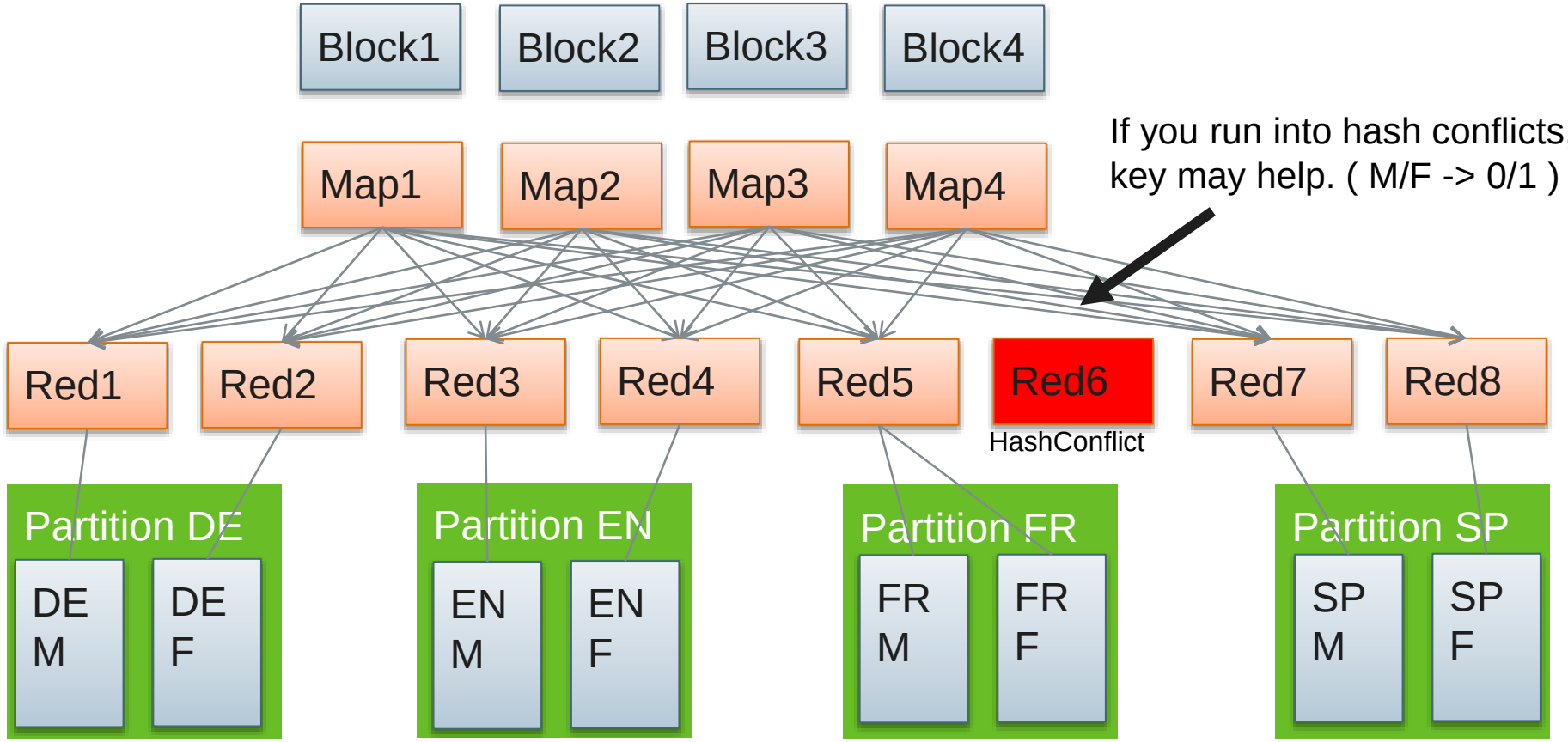
```
set mapred.reduce.tasks=2
```

- **For dynamic partitioning include the partition key in the distributiom**
  - Any additional subkeys result in multiple files per partition folder (not unlike bucketing)
  - For fast load try to maximize number of reducers in cluster

Distribute By

Reducers and number of distribution keys do not have to be identical but it is good best practice

```
SET MAPRED.REDUCE.TASKS = 8;
INSERT INTO ORC_SALES PARTITION ( COUNTRY) SELECT FROM DEL_SALES
    DISTRIBUTE BY COUNTRY, GENDER;
```



Hortonworks

Agenda

- Introduction
  - ORC files
  - Partitioning vs. Predicate Pushdown
- Loading data
  - Dynamic Partitioning
  - Bucketing
  - Optimize Sort Dynamic Partitioning
  - Manual Distribution
- **Miscellaneous**
  - Sorting and Predicate pushdown
  - Debugging
  - Bloom Filters

SORT BY for Predicate Pushdown (PPD)

- **ORC can skip stripes (and 10k sub-blocks) of data based on ORC footers**
  - Data can be skipped based on min/max values and bloom filters

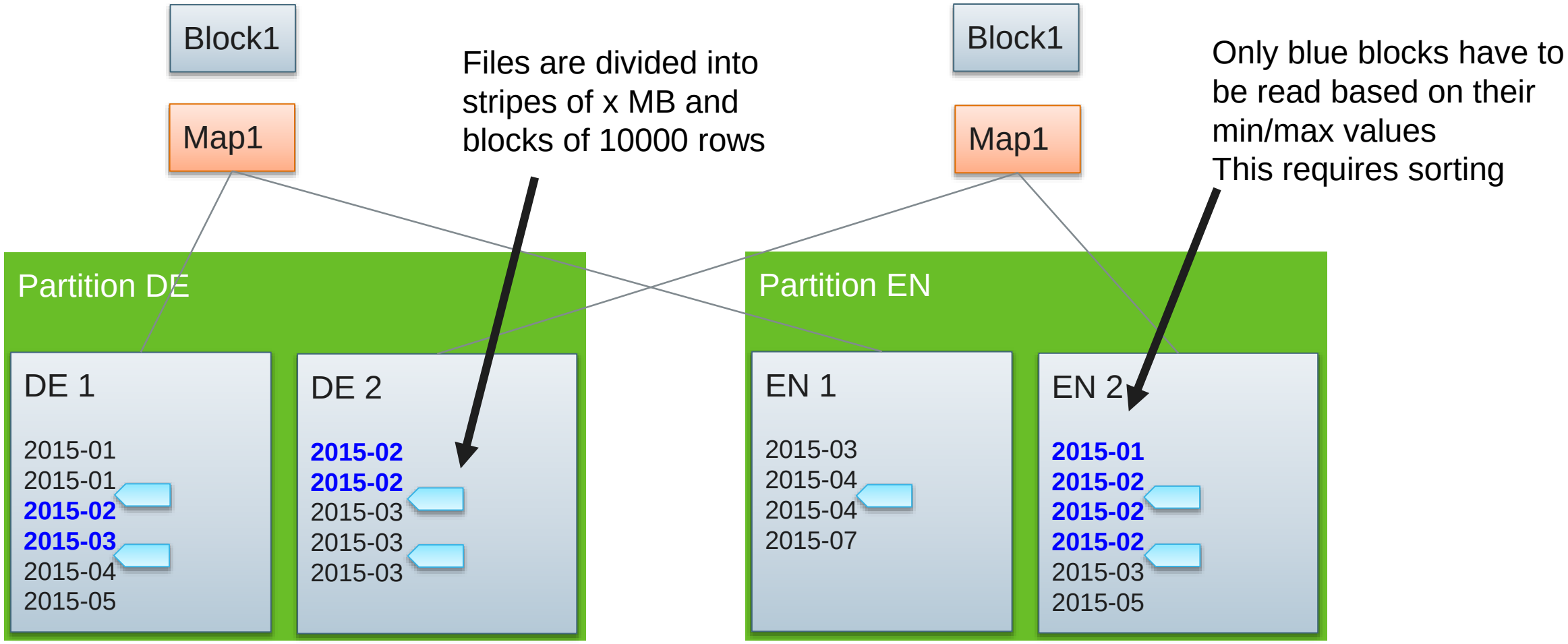
- **In warehouse environments data is normally sorted by date**
  - For initial loads or other predicates data can be sorted during load
  - Two ways to sort data: **`ORDER BY`** (global sort, slow) and **`SORT BY`** (sort by reducer)
    - Use want **`SORT BY`** for PPD: faster and cross-file sorting does not help PPD

- **Can be combined with Distribution, Partitioning, Bucketing to optimize effect**

```
CREATE TABLE ORC_SALES
    ( CLIENTID INT, DT DATE, REV DOUBLE, PROFIT DOUBLE, COMMENT STRING )
    STORED AS ORC;

INSERT INTO TABLE ORC_SALES SELECT * FROM DEL_SALES SORT BY DT;
```

Sorting when Inserting into Table




```
SELECT * FROM DATA_ORC WHERE dt = 2015-02
```

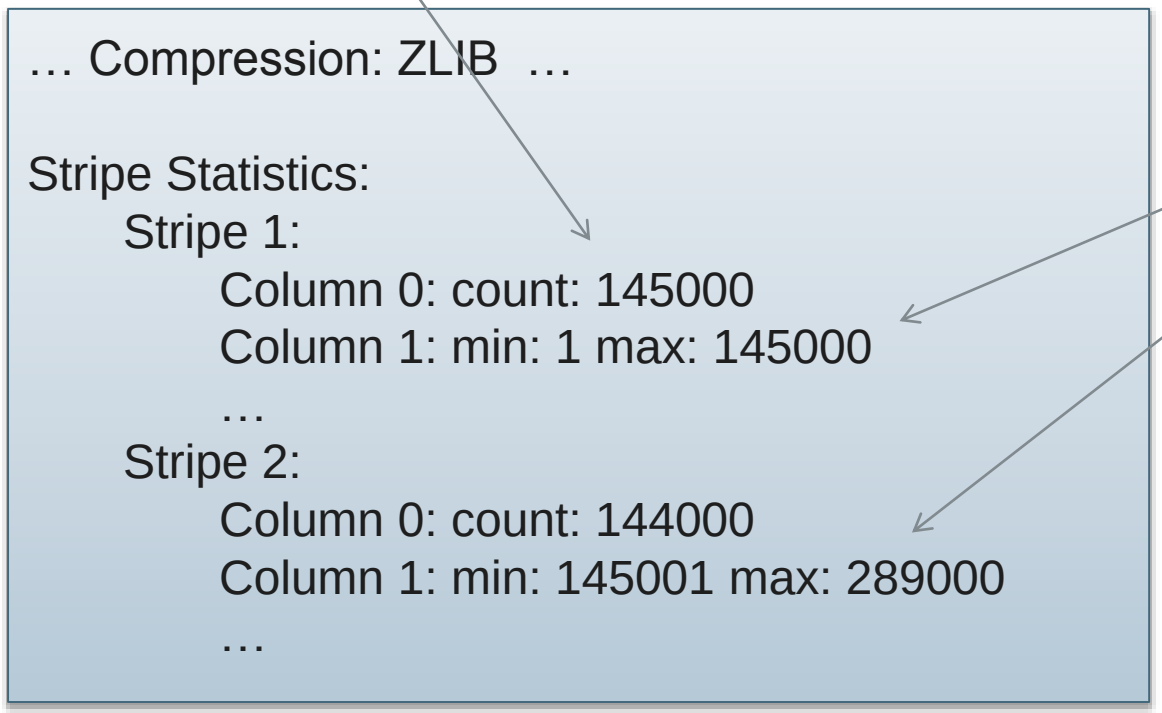

Hortonworks

Checking Results

- **Use hive –orcfiledump to check results in ORC files**

```
hive –orcfiledump /apps/hive/warehouse/table/dt=3/00001_0
```

Check Number of Stripes and number rows
- small stripes (5000 rows) indicate a memory problem during load



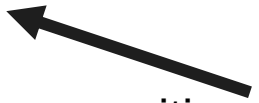
Data should be sorted on your predicate columns

Hortonworks

Bloom Filters

- **New feature in Hive 1.2**
- **A hash index bitmap of values in a column**
  - If the bit for hash(value) is 0, no row in the stripe can be your value
  - If the bit for hash(value) is 1, it is possible that the stripe contains your value
- **Hive can skip stripes without need to sort data**
  - Hard to sort by multiple columns

```
CREATE TABLE ORC_SALES ( ID INT, Client INT, DT INT… );
STORED AS ORC TBLPROPERTIES
("orc.bloom.filter.columns"="Client,DT");
```



Parameter needs case sensitive comma-separated list of columns

Hortonworks

Bloom Filters

- **Bloom Filters are good**
  - If you have multiple predicate columns
  - If your predicate columns are not suitable for sorting (URLs, hash values, …)
  - If you cannot sort the data (daily ingestion, filter by clientid)

- **Bloom Filters are bad**
  - If every stripe contains your value
    - low cardinality fields like country
    - Events that happen regularly (client buys something daily)

- **Check if you successfully created a bloom filter index with orcfiledump**

```
hive --orcfiledump -rowindex 3,4,5 /apps/hive/…
```

You only see bloom filter indexes if you specify the columns you want to see

Verify ORC indexes

- **Switch on additional information like row counts going in/out of Tasks**

```
SET HIVE.TEZ.PRINT.EXEC.SUMMARY = TRUE;
```

- **Run query with/without Predicate Pushdown to compare row counts:**

```
set hive.optimize.index.filter=false;
    // run query
set hive.optimize.index.filter=true;
    // run query
    // compare results
```

Summary

- **Partitioning and Predicate Pushdown can greatly enhance query performance**
  - Predicate Pushdown enhances Partitioning, it does not replace it
  - Too many partitions lead to performance problems

- **Dynamic Partition loading can lead to problems**
  - Normally Optimized Dynamic Sorted Partitioning solves these problems
  - Sometimes manual distribution can be beneficial

- **Carefully design your table layout and data loading**
  - Sorting is critical for effective predicate pushdown
  - If sorting is no option bloom filters can be a solution

- **Verify data layout with orcfiledump and debug information**