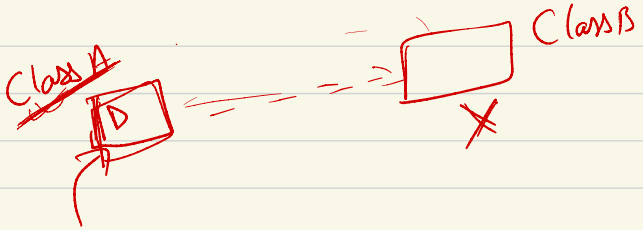
Serialization → Converting Objects into bytes

Deserialization → Converting bytes into objects

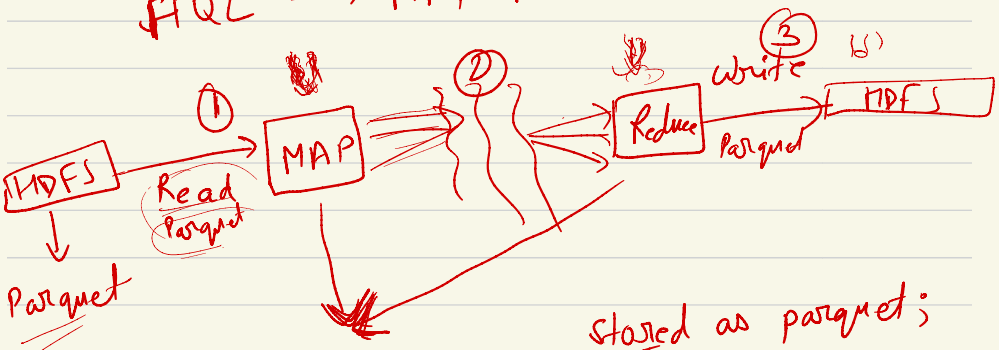
Input format → Parquet SerDe

Output format → Parquet SerDE

SerDe library → Parquet Hive SerDe



HQL → MAP-Reduce



stored as parquet;

stored as ORC;

Row format delimited ; .csv

Row format serDe "OpenCSVSerde";

.csv
↓
quotes
Nested delimitor

Row
select * from employee
name, salary

SerDe library → [Parse thor raw data]

Deserialized object (ROW)

" " "

" " "

" \ " "

" Shashank " "

" Shashank , Mishra " "

quoteChar = " "

quoteChar = " " "

escapChar = " \ "

" Shashank, Mishra "

" Shashank, Mishra " DE "

" Shashank, Mishra " DE " "

Raw data → " Shashank, Mishra \" DE \" "

output → Shashank , Mishra " DE "

a = "shashank"

a = 'shashank'

"shashank, Mishra"

name, location

shash, "Bangalore, India"

quoteChar

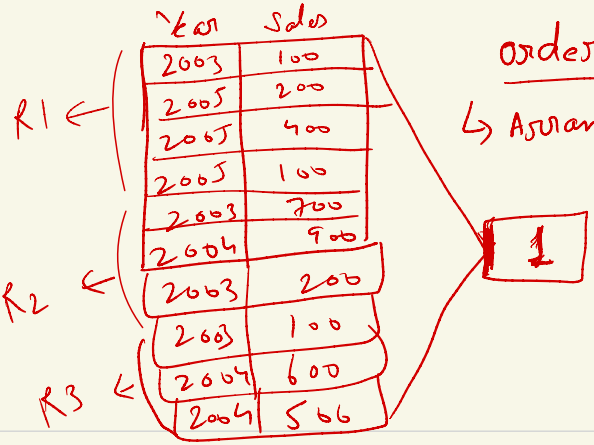
" "

"shashank"

Sales	Year_ID	Qual
200	2003	1
500	2003	2
700	2004	4
100	2004	5
300	2005	6
200	2005	7

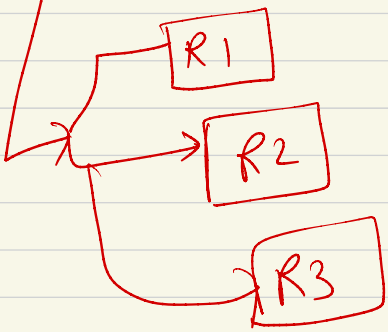
select year_id, sum(sales) as total_sales
from sales_data group by year_id;

2003, 700
2004, 800
2005, 500



order by →
↳ Arrange data globally

2005
2005
2005
2004
2004
2004
2003
2003
2003



Sort by
↳ local sorting

2005
2004
2003
→ R1

2005
2004
2003
→ R2

2005
2004
2003
→ R3

Partitioning -

R1 → Shashank, IT
R2 → Rahul, QA
R3 → Amit, IT

IM → Record

employee

name	salary	Country
Shashank	1	India
Abdul	1000	UK
Amit	500	India
Rahul	400	China
Surbhi	800	USA
Kapil	900	USA
Hulk	1000	Russia

→ which column should be used for partitioning?

→ Columns used to filter the data frequently

→ Always avoid Primary key type of columns for partitioning
↳ IM (emp_id)

IM (Super Bad)

↳ loss Uniqueness (Repeated multiple time or used)

any hive table with partition

normal Internal hive table

↳ /user/hive/warehouse/db_name/tb_name/

↳ /user/hive/warehouse/db_name/tb_name/country=IND/
/country=USA/
/country=China/
/country=Russia/
/country=UK/

- Static Partitioning → we know in which partion data will get loaded
- Dynamic Partitioning → Hive will scan the partion column and will create partion in hdfs for each distinct value

Q) Dynamic & Static Partitioning
which one perform slow?

Create dynamic table

{

=

=

}

Partitioned by (country, year_id)

hdfs-path — /country=USA/year_id=2005

/ country=USA/year_id=2004