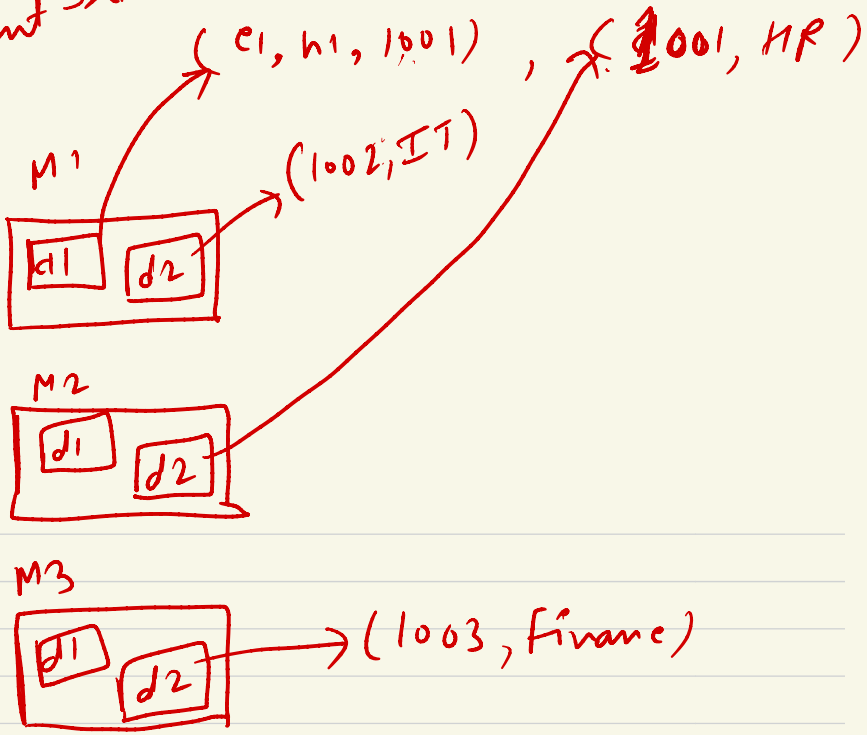
Bucketing in Hive

employee → d1 } in HDFS
department → d2

Join



What is Bucketing

Basically, decomposing table data sets into more mangable parts.

employee Tables

emp_id	emp_name	dept_id
1	S1	100
2	S2	103
3	S3	101
4	S5	102

bucket column (dept_id) → cluster_id

buckets = 5

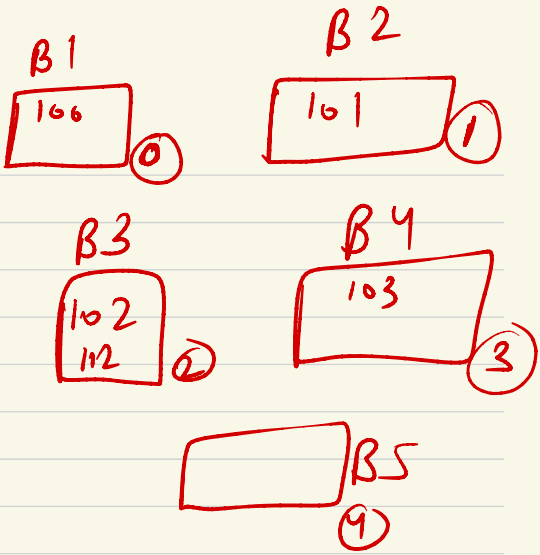
hash_funct(key) = ? (key % num_of_buckets)

i) hash(100) = 100 % 5
= 0

ii) hash(103) = 103 % 5
= 3

iii) hash(101) = 101 % 5
= 1

iv) hash(102) = 102 % 5



Country is partition Column

How it looks in warehouse?

hdfs_path/Country=INDIA

hdf_path/country=USA

*) Bucket representation for a table without partition?

Bucket num = 5

hdfs_path_table/ Part_01, Part_02, Part_03, Part_04, Part_05 } file

*) Bucket representation if a partitioned table!

hdf_path_table/country=INDIA/ P0 P1 P2 P3 P4

hdfs_path_table/country=USA/ P0 P1 P2 P3 P4

Optimized joins in Hive

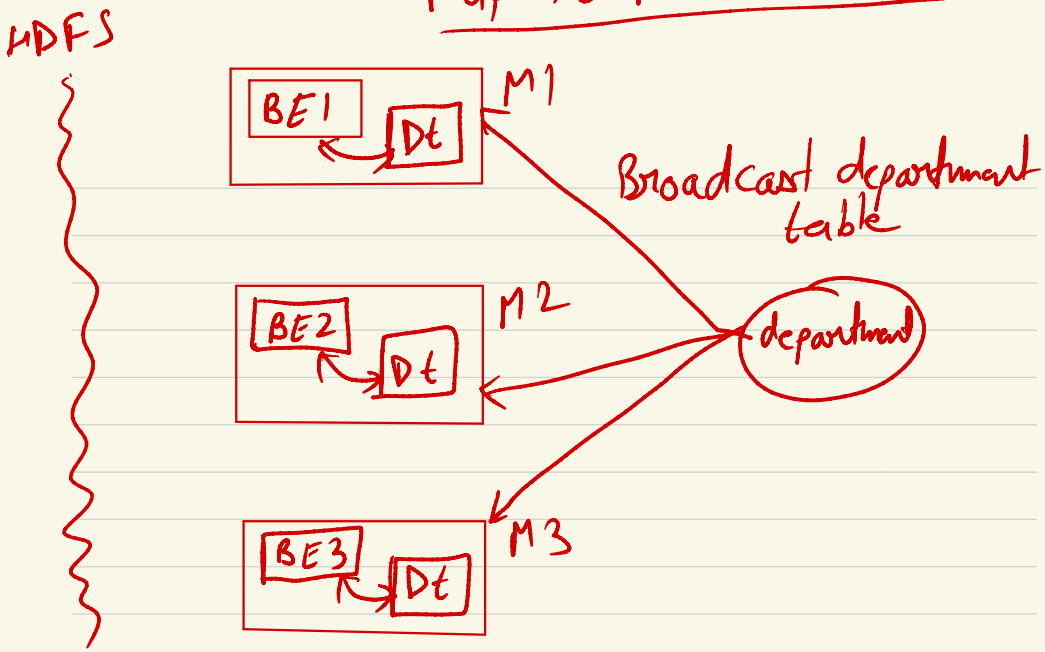
Map Join (Broadcast Join)

employee.Join(department)

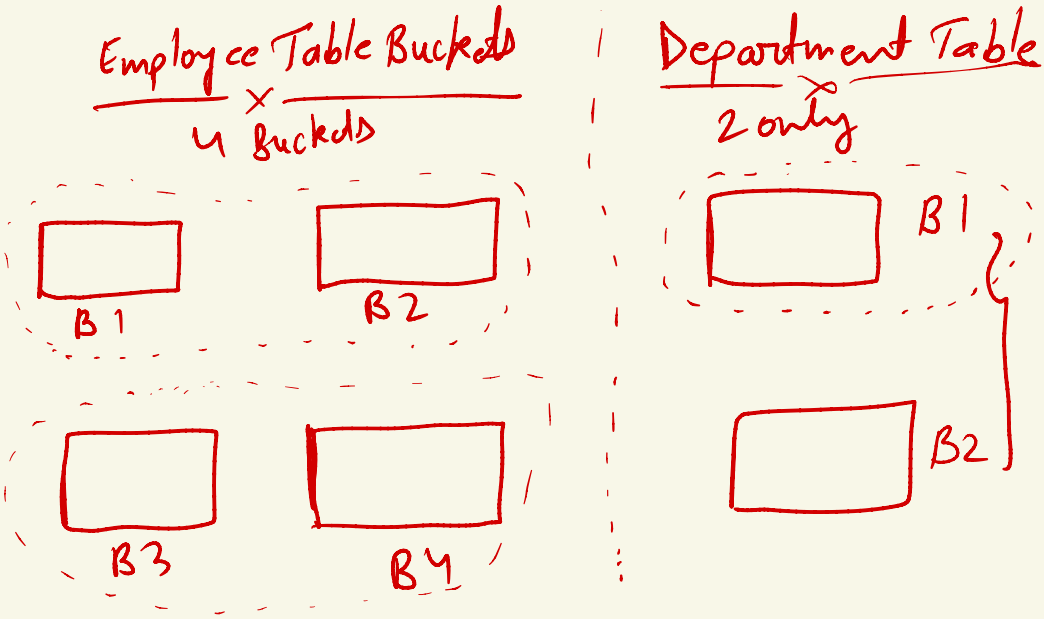
employee = 20K records

department = 10 records

Map Task Execution



Bucket Map Join



Condition for Bucket Map Join

1. Tables are bigger in size
2. Both tables are bucketized on the same join column.
3. Number of buckets in both of the table are multiple of each other

Ex

City Table (2 B) (cityid) (Join) Sales_Table (2 B)

B1

cityid	cityN
3	NY
1	SJ

B1

cityid	Sales
1	500
3	6000
1	400

B2

cityid	name
2	SF
4	LA

B2

cityid	Sales
4	50
2	200
4	45

Sorted - Merge - Bucked Join

(M) arr1 = [(1), 5, 9, 11] x

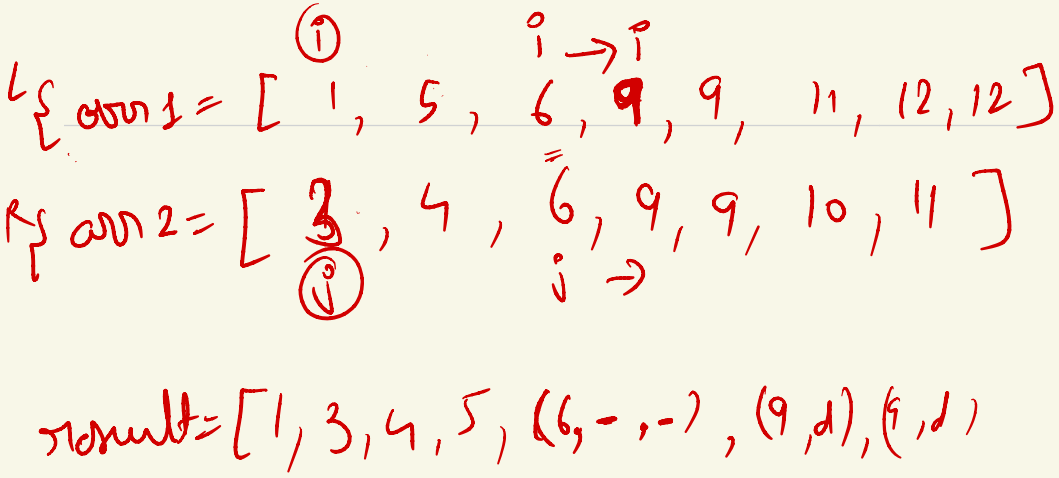
i → (i) → (i) → (i)

(M) arr2 = [(2), 4, 7, 12, 20, 21]

j → j → j → j

result = [1, 2, 4, 5, 7, 9, 11, 12, 20, 21]

O(n+m) → linear Complexity


arr1 = [1, 5, 6, 9, 9, 11, 12, 12]
arr2 = [3, 4, 6, 9, 9, 10, 11]
i →
j →
result = [1, 3, 4, 5, (6, -, -), (9, d), (9, d)