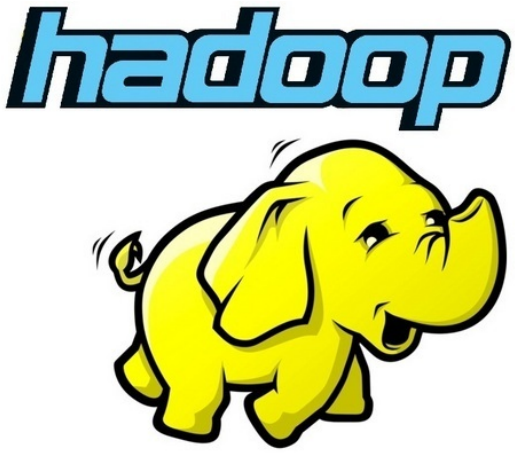
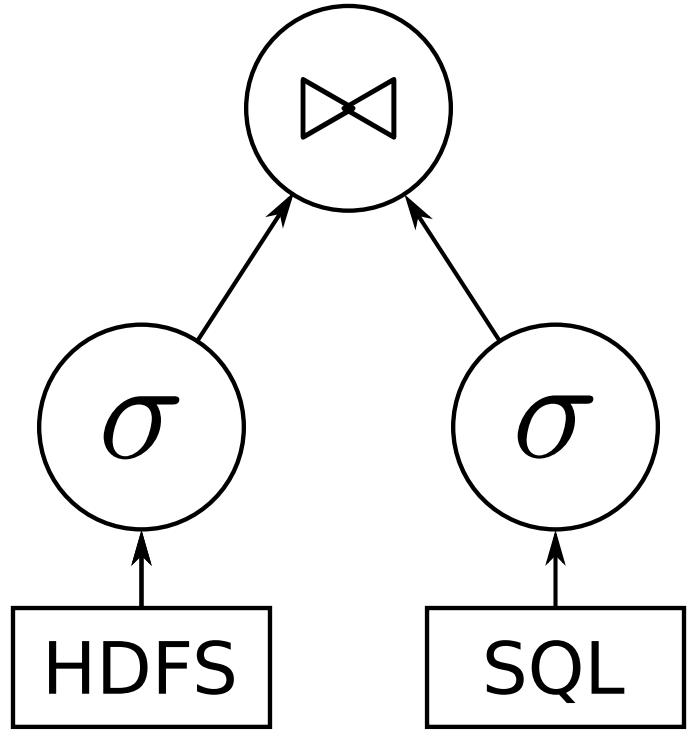
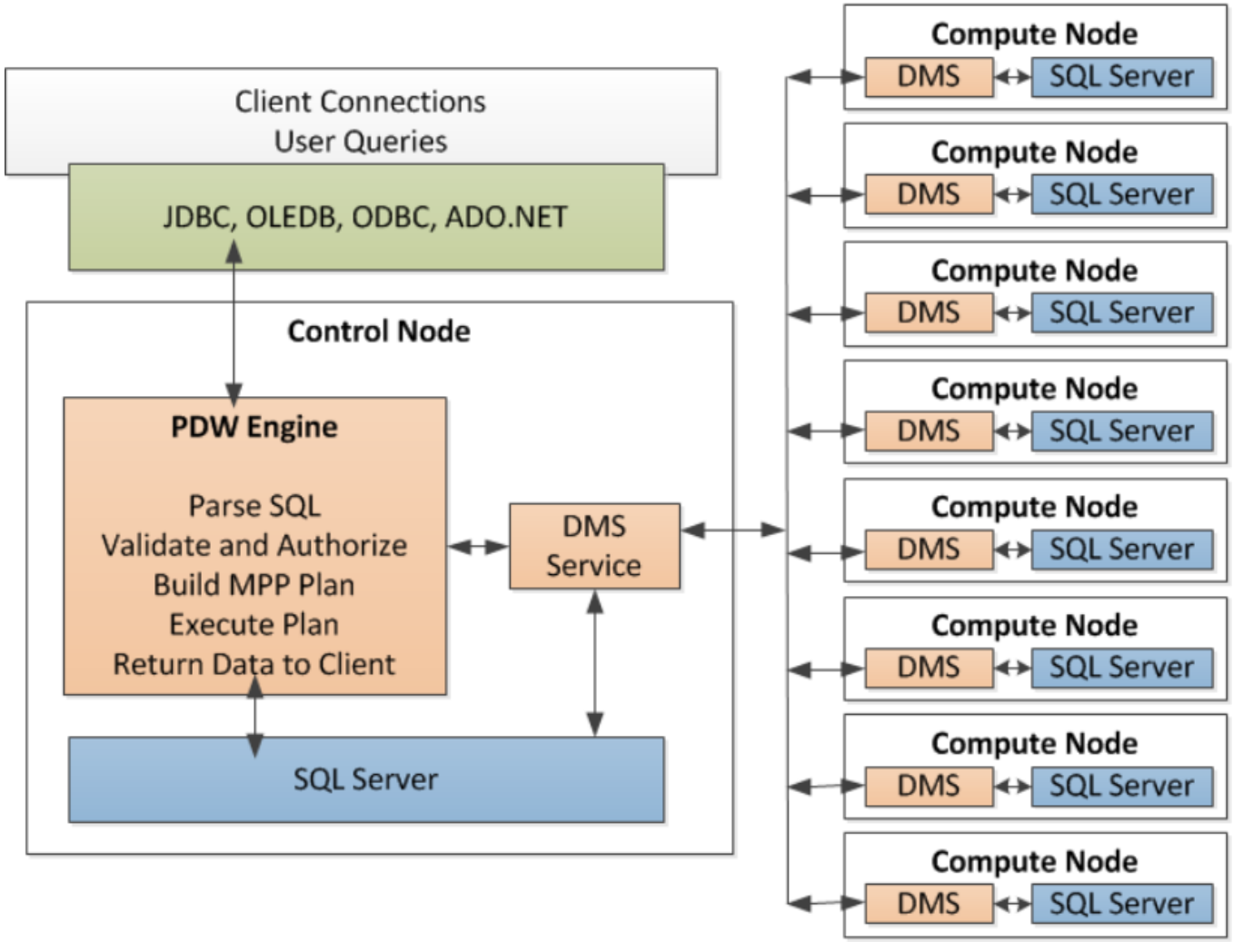
Split Query Processing in Polybase

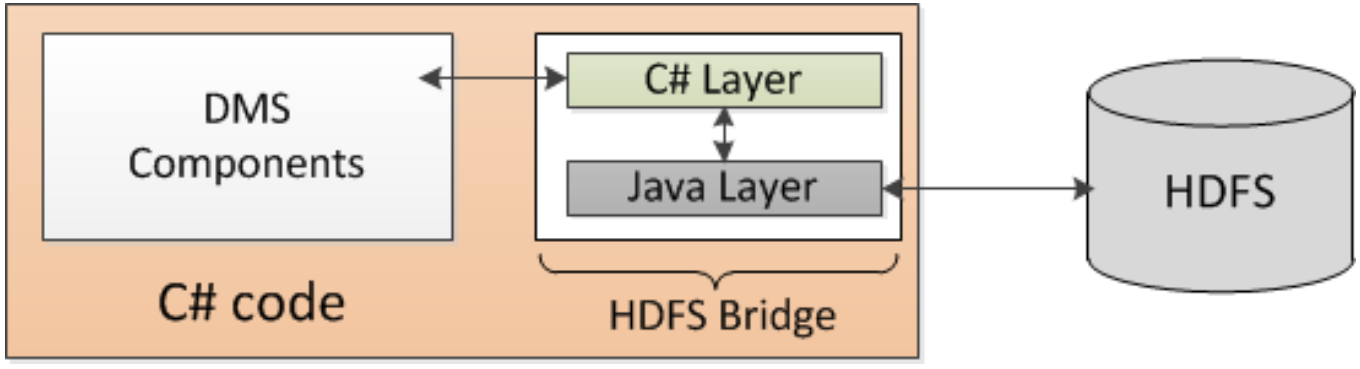


VS

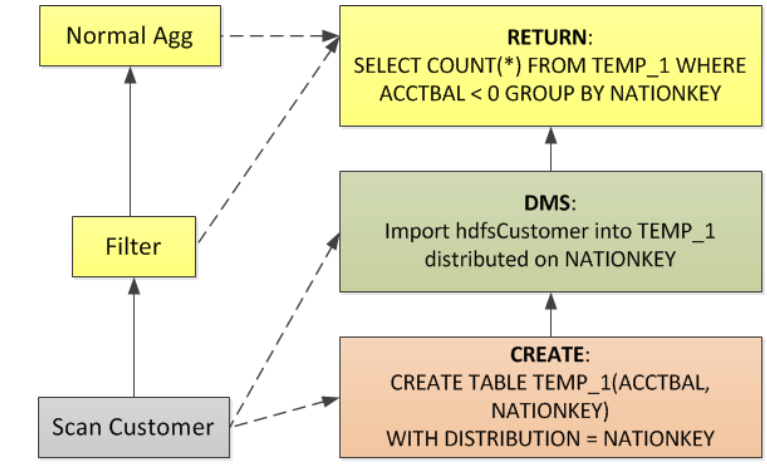

⋈
σ
σ
HDFS
SQL


Client Connections
User Queries
JDBC, OLEDB, ODBC, ADO.NET
Control Node
PDW Engine
Parse SQL
Validate and Authorize
Build MPP Plan
Execute Plan
Return Data to Client
DMS Service
SQL Server
Compute Node
DMS
SQL Server
Compute Node
DMS
SQL Server
Compute Node
DMS
SQL Server
Compute Node
DMS
SQL Server
Compute Node
DMS
SQL Server
Compute Node
DMS
SQL Server
Compute Node
DMS
SQL Server

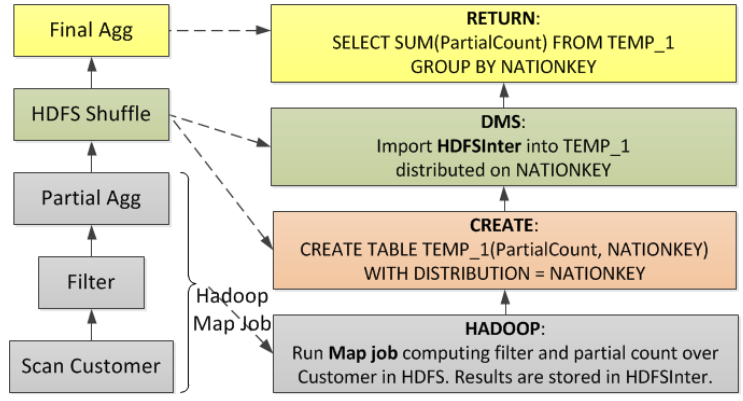
```
CREATE EXTERNAL TABLE hdfsCustomer
   ( c_custkey        bigint not null,
     c_name           varchar(25) not null,
     c_address        varchar(40) not null,
     c_nationkey      integer not null,
     c_phone          char(15) not null,
     c_acctbal        decimal(15,2) not null,
     c_mktsegment     char(10) not null,
     c_comment        varchar(117) not null)
WITH (LOCATION='/tpch1gb/customer.tbl',
FORMAT_OPTIONS (EXTERNAL_CLUSTER = GSL_CLUSTER,
EXTERNAL_FILEFORMAT = TEXT_FORMAT));
```



```
SELECT count (*) from Customer
WHERE acctbal < 0
GROUP BY nationkey
```



(a) Optimized query plan (b) Corresponding DSQL plan



(a) Alternative query plan (b) Corresponding DSQL plan

```
SELECT TOP 10 unique1, unique2, unique3, stringu1,
  stringu2, string4 FROM T1
WHERE (unique1 % 100) < T1-SF
```

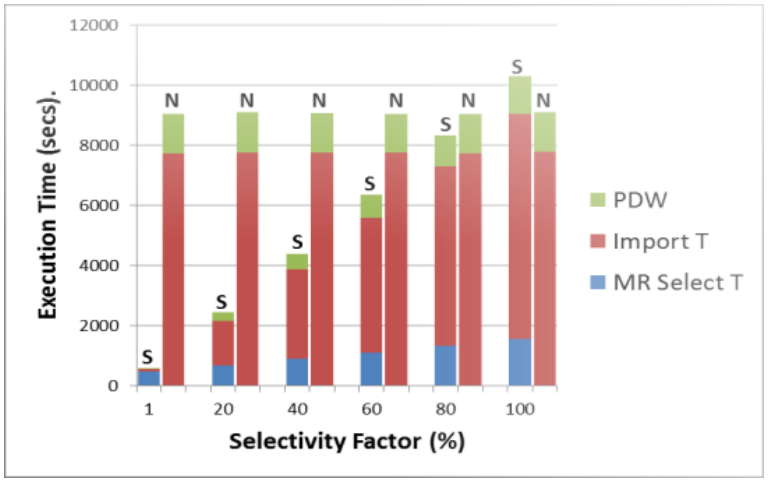


Figure 10: Q1 with T1 in HDFS using C-16/48.

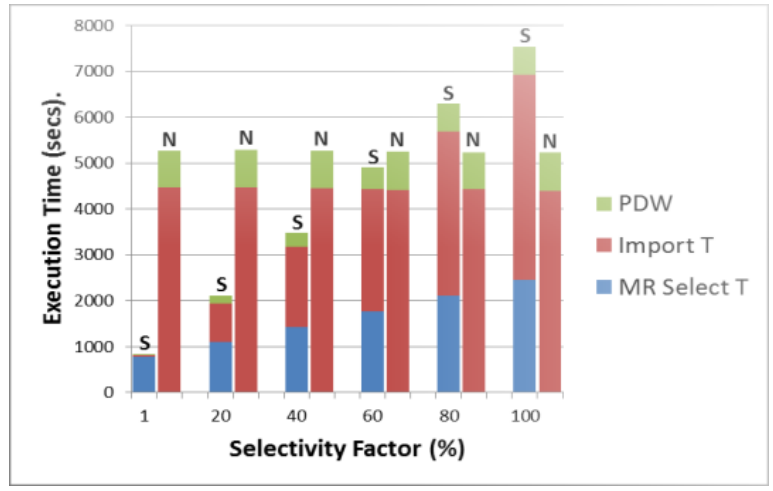


Figure 11: Q1 with T1 in HDFS using C-30/30.

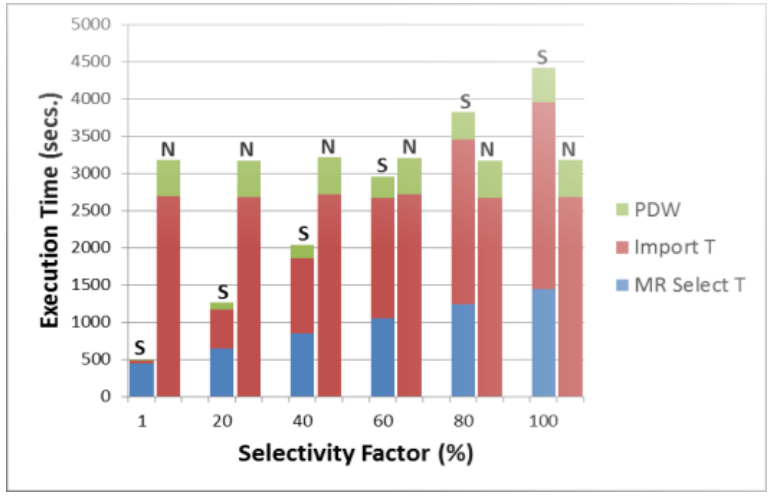
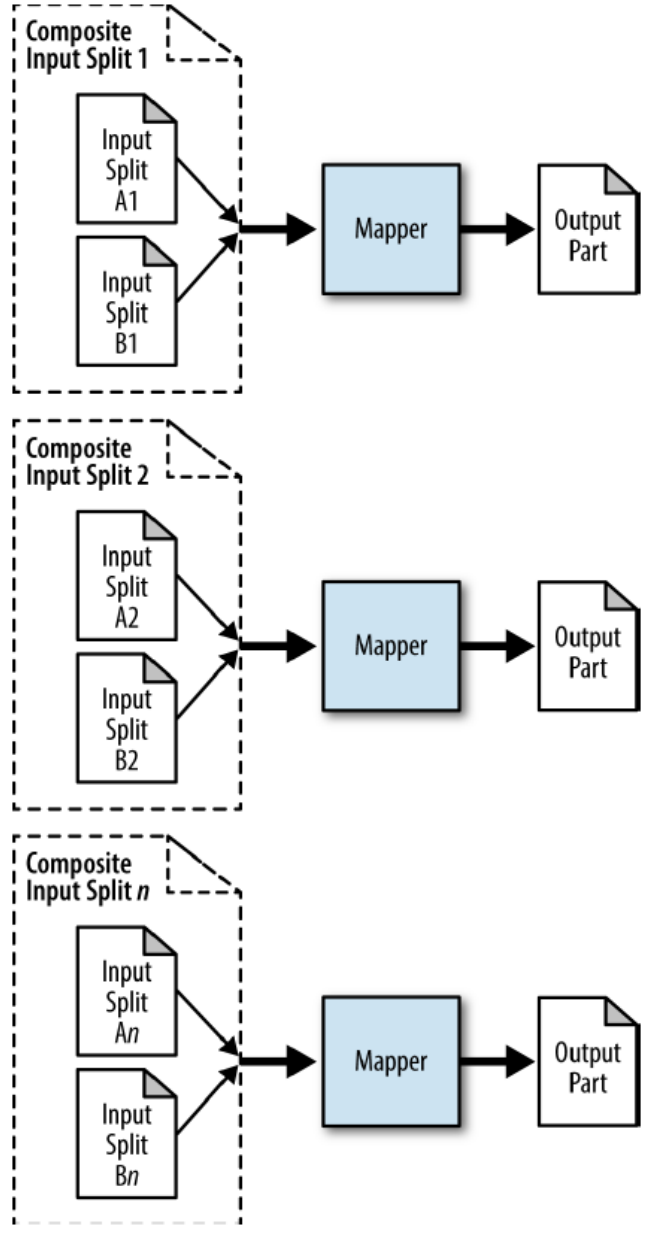
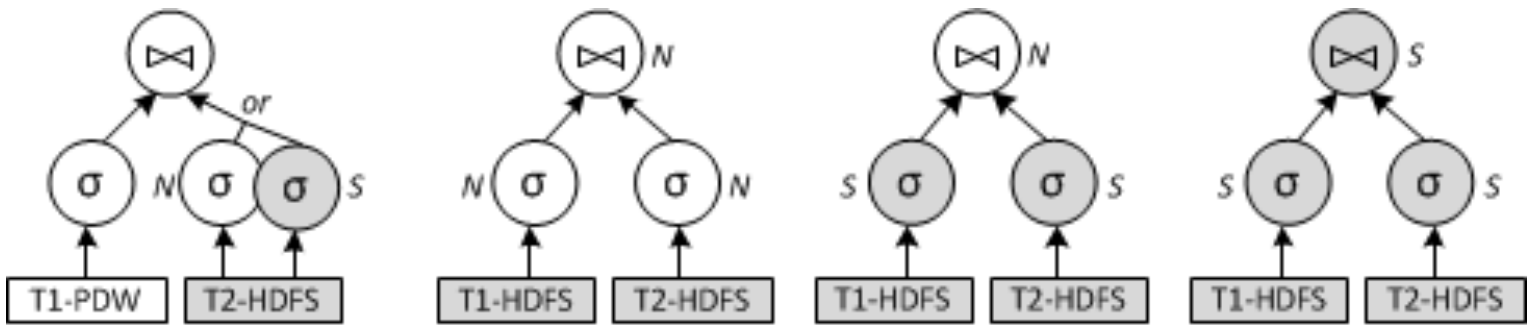


Figure 12: Q1 with T1 in HDFS using C-60.

```
SELECT TOP 10 T1.unique1, T1.unique2, T2.unique3,
  T2.stringu1, T2.stringu2
FROM T1 INNER JOIN T2 ON (T1.unique1 = T2.unique2)
WHERE T1.onePercent < T1-SF AND
  T2.onePercent < T2-SF
ORDER BY T1.unique2
```



```
SELECT TOP 10 T1.unique1, T1.unique2, T2.unique3,
  T2.stringu1, T2.stringu2
FROM T1 INNER JOIN T2 ON (T1.unique1 = T2.unique2)
WHERE T1.onePercent < T1-SF AND
  T2.onePercent < T2-SF
ORDER BY T1.unique2
```



Q2-a Q2-b Q2-c Q2-d

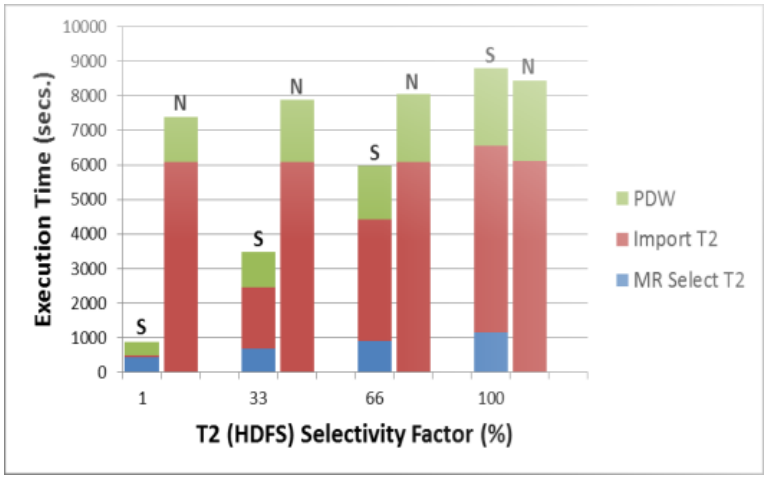


Figure 14: Q2-a (T1 in PDW and T2 in HDFS) Configuration C-16/48. T1-SF fixed at 30%.

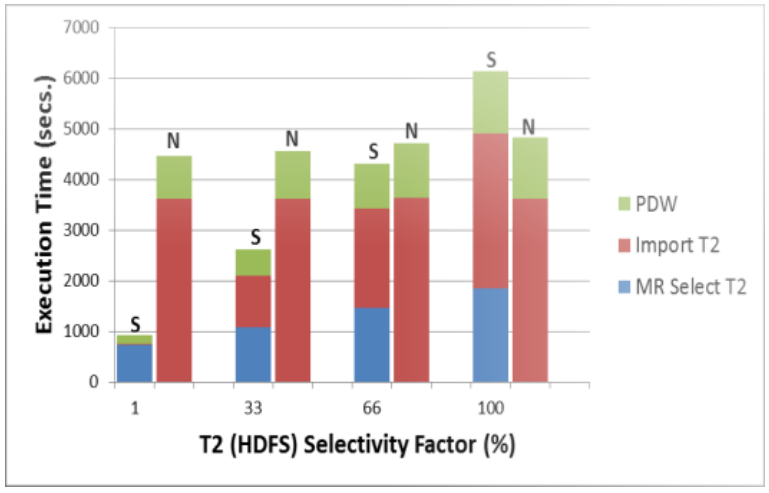
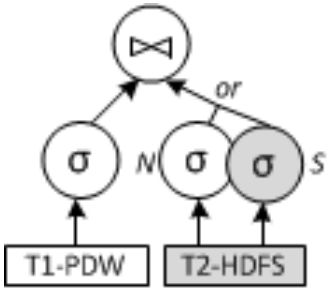


Figure 15: Q2-a (T1 in PDW and T2 in HDFS) Configuration C-30/30. T1-SF fixed at 30%.



Q2-a

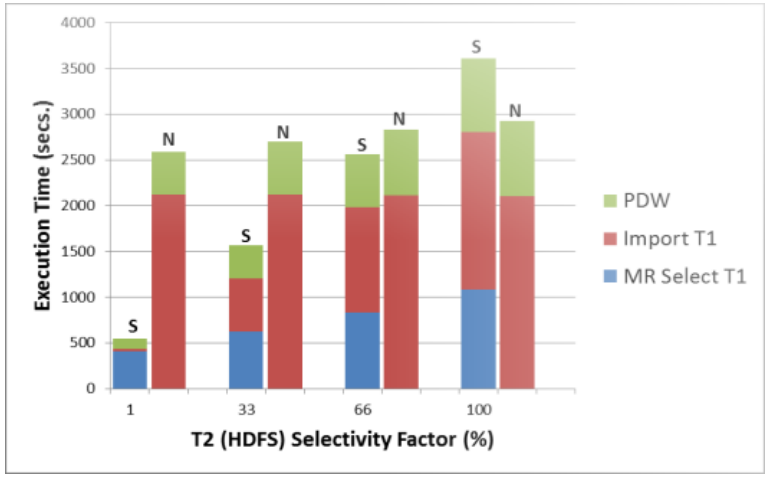
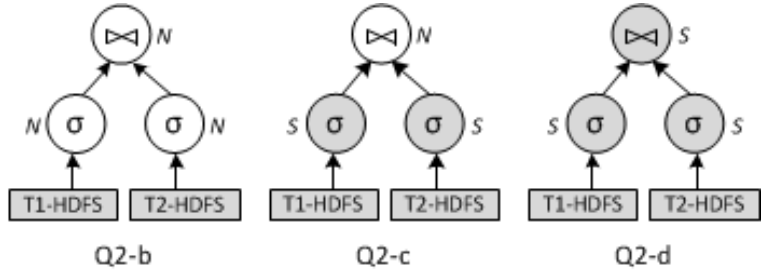


Figure 16: Q2-a (T1 in PDW and T2 in HDFS) Configuration C-60. T1-SF fixed at 30%.



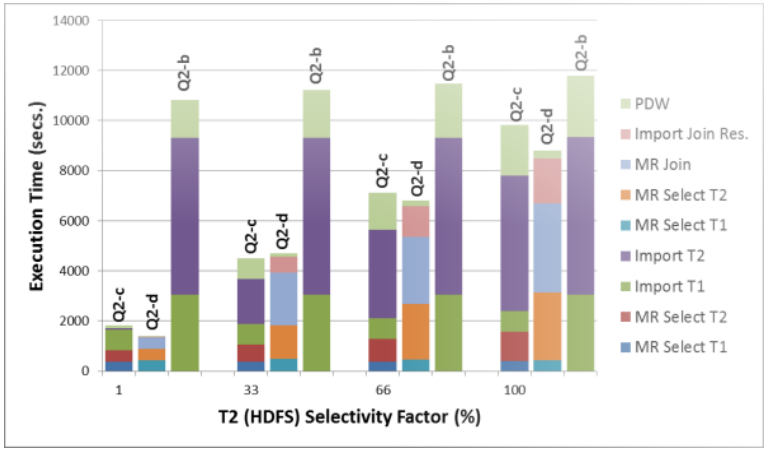


Figure 17: Queries Q2-b, c, & d (T1 and T2 in HDFS) Configuration C-16/48. T1-SF fixed at 30%.

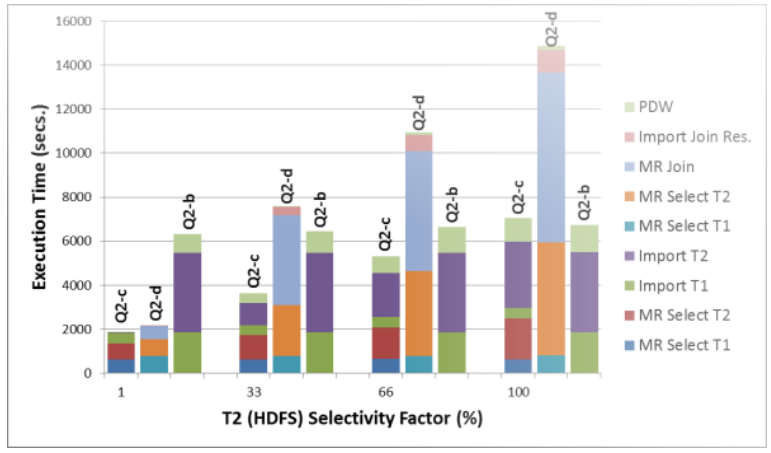


Figure 18: Queries Q2-b, c, & d (T1 and T2 in HDFS) Configuration C-30/30. T1-SF fixed at 30%.

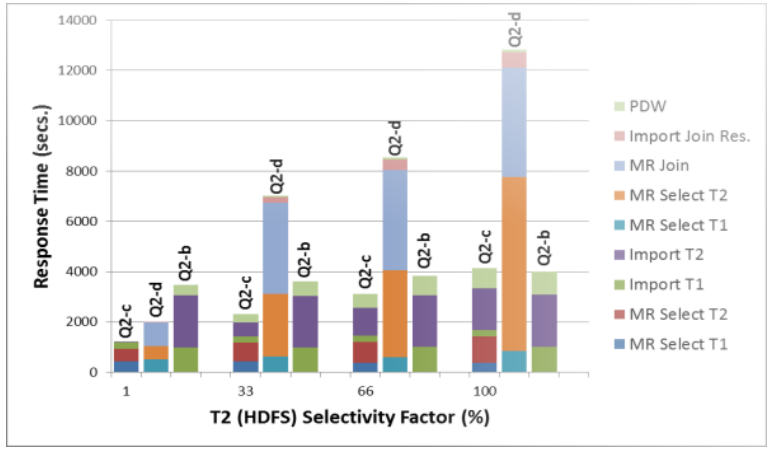


Figure 19: Queries Q2-b, c, & d (T1 and T2 in HDFS) Configuration C-60. T1-SF fixed at 30%.

Fractured Mirrors

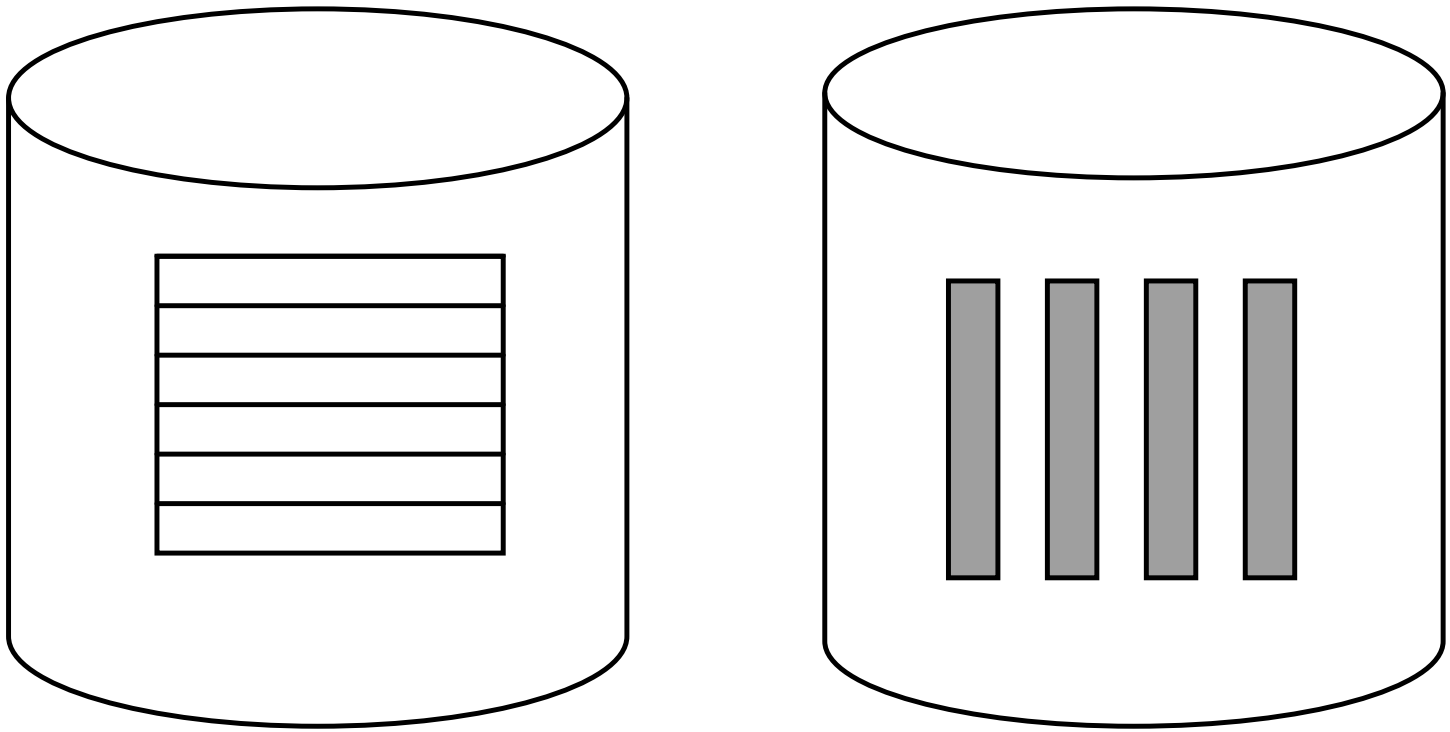


Figure 1: Data placement based on storage model.

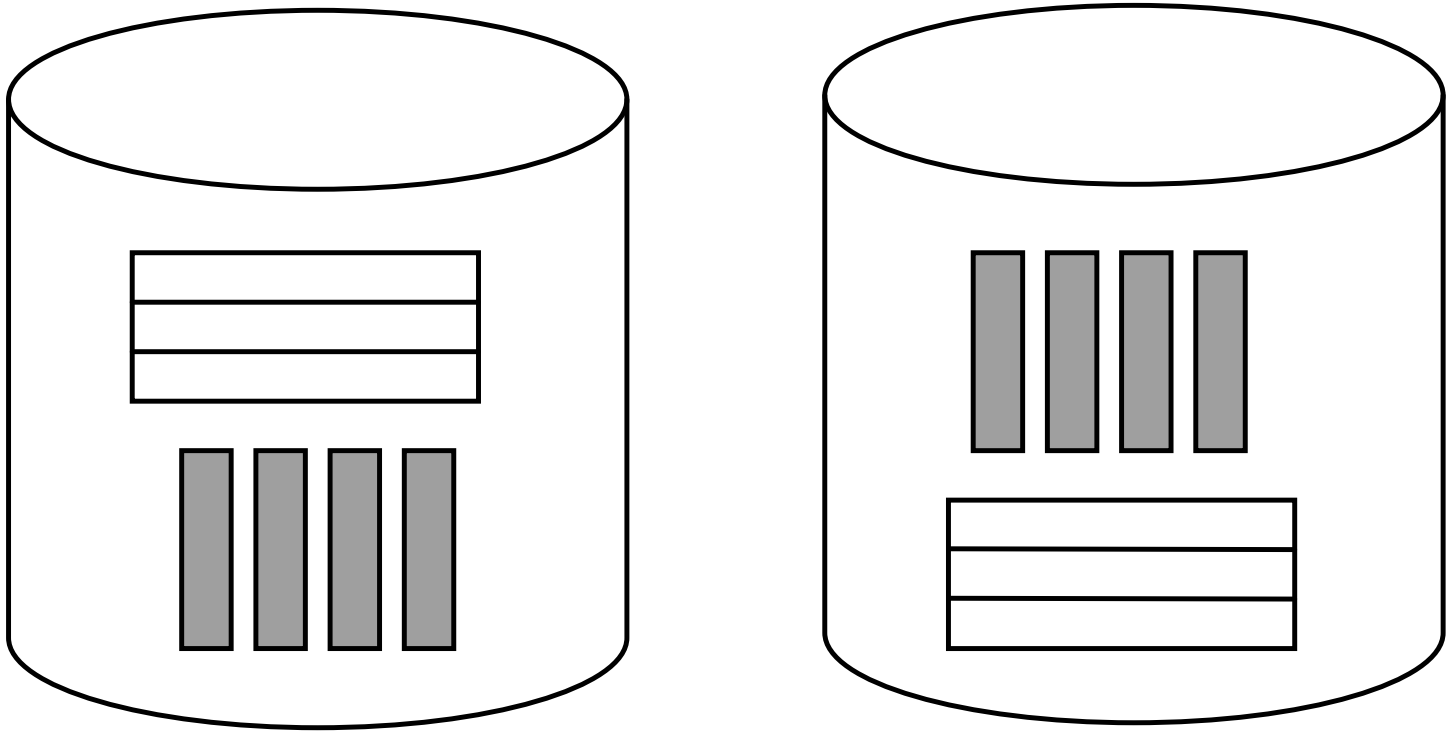


Figure 2: Data placement for fractured mirrors.

Query optimization: optimize-twice

Two relations: $R\,(R1, R2, R3)$ and $S\,(S1, S2)$

$Q = \pi R2(R \bowtie S)$ (join on $R1 = S1$)

DMS query: $\pi R2(R - 1 \bowtie R - 2 \bowtie S - 1)$

Query optimization: combined-search

$Q = \pi R2(R \bowtie S)$ (join on $R1 = S1$)

Query optimization: combined-search

$Q = \pi R2(R \bowtie S)$ (join on $R1 = S1$)

Start with:

$$\{R\}, \{R1\}, \{R2\}, \{S\}, \{S1\}$$

Query optimization: combined-search

$Q = \pi R2(R \bowtie S)$ (join on $R1 = S1$)

Start with:

$$\{R\}, \{R1\}, \{R2\}, \{S\}, \{S1\}$$

Combine and group by equivalence class:

- Class 1: $\{R, S\}$, $\{R, S1\}$. Best plan: $\{R, S1\}$
- Class 2: $\{R1, S1\}$, $\{R1, S\}$. Best plan: $\{R1, S1\}$
- Class 3: $\{R1, R2\}$

Query optimization: combined-search

$Q = \pi R2(R \bowtie S)$ (join on $R1 = S1$)

Start with:

$$\{R\}, \{R1\}, \{R2\}, \{S\}, \{S1\}$$

Combine and group by equivalence class:

- Class 1: $\{R, S\}$, $\{R, S1\}$. Best plan: $\{R, S1\}$
- Class 2: $\{R1, S1\}$, $\{R1, S\}$. Best plan: $\{R1, S1\}$
  - $\{R1, S1, R2\}$.
- Class 3: $\{R1, R2\}$
  - $\{R1, R2, S\}$, $\{R1, R2, S1\}$. Best Plan: $\{R1, R2, S1\}$

Best plan overall: $\{R, S1\}$ (hybrid plan!)