C-Store 8 Years Later

Stephen Walkauskas
Software Engineer, HP Vertica

CMU PDL February 13, 2014

You told them what?

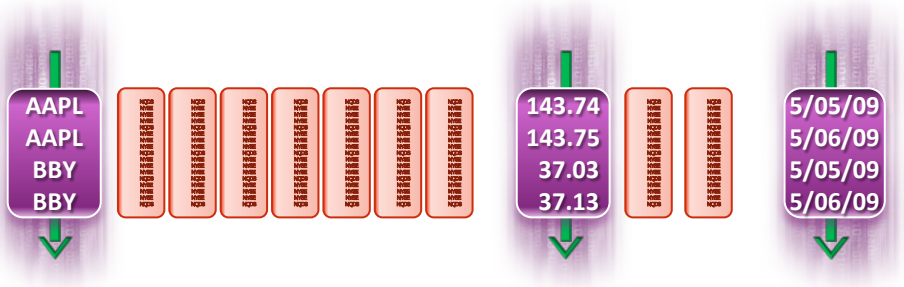
Building from a prototype

Improved results

Column Store – Column-Based Disk I/O

Typical FinServ price per stock for 1 day

Column Store - Reads 3 columns




```
SELECT AVG(price)
FROM tickstore
WHERE
symbol = 'AAPL"  AND date = '5/06/09'
```

Row Store - Reads all columns

Materialize columns

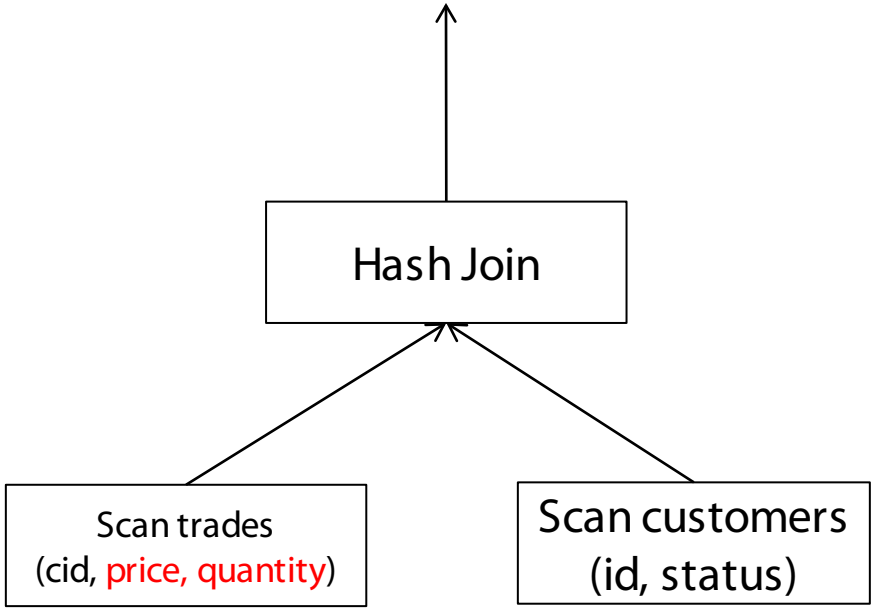
```
  SELECT 1, AVG(price * quanity)
    FROM trades
   WHERE sym = ‘HPQ’
GROUP BY extract(tdate,’month’);
```

Materialize columns

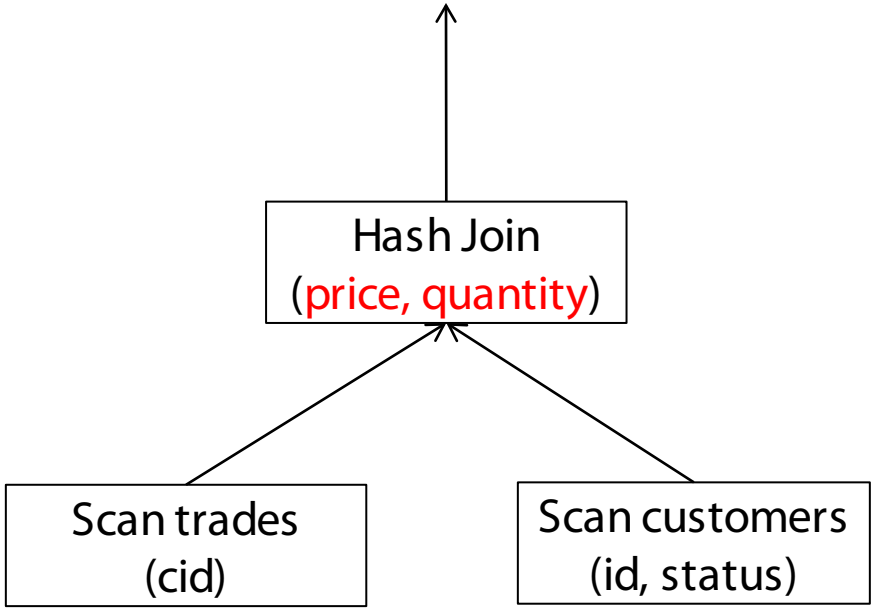
```
  SELECT AVG(t.price*t.quantity), t.id
    FROM trades t JOIN customer c ON t.cid = c.id
   WHERE c.status = 'SPECIAL'
GROUP BY t.id;
```


7

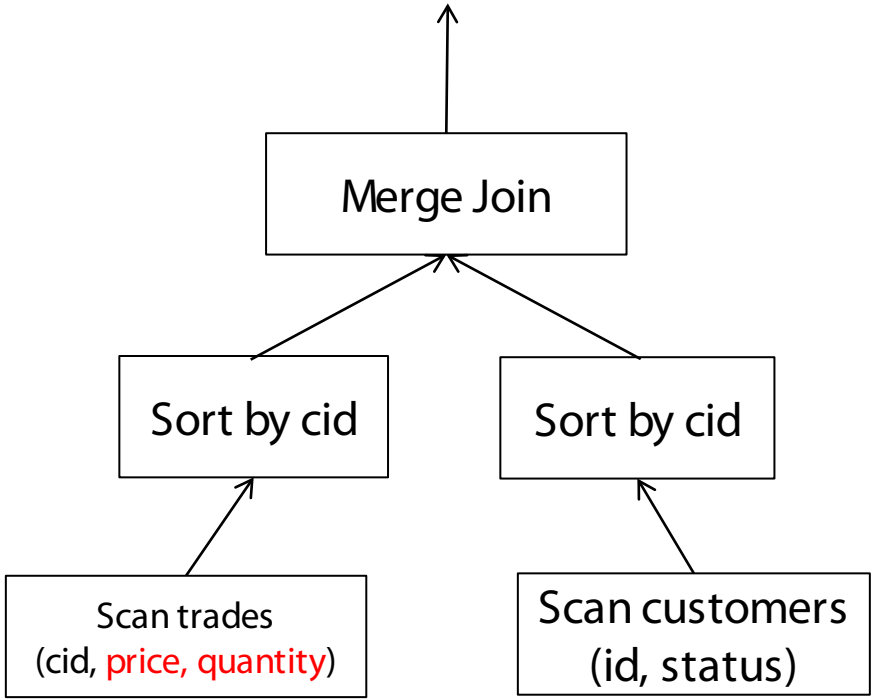
Early Materialization



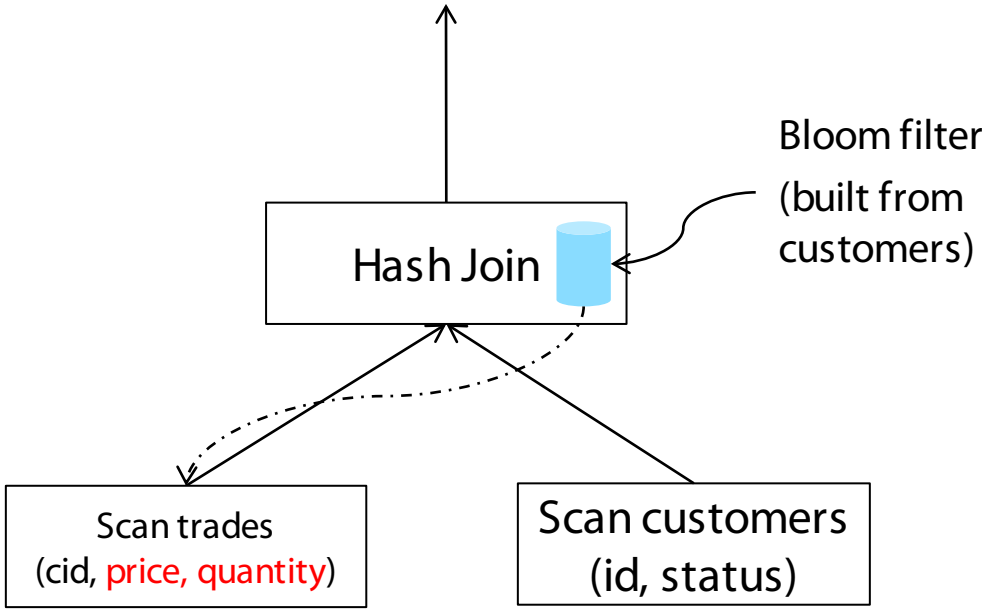
Late Materialization



~~Late Materialization~~ (join spills)

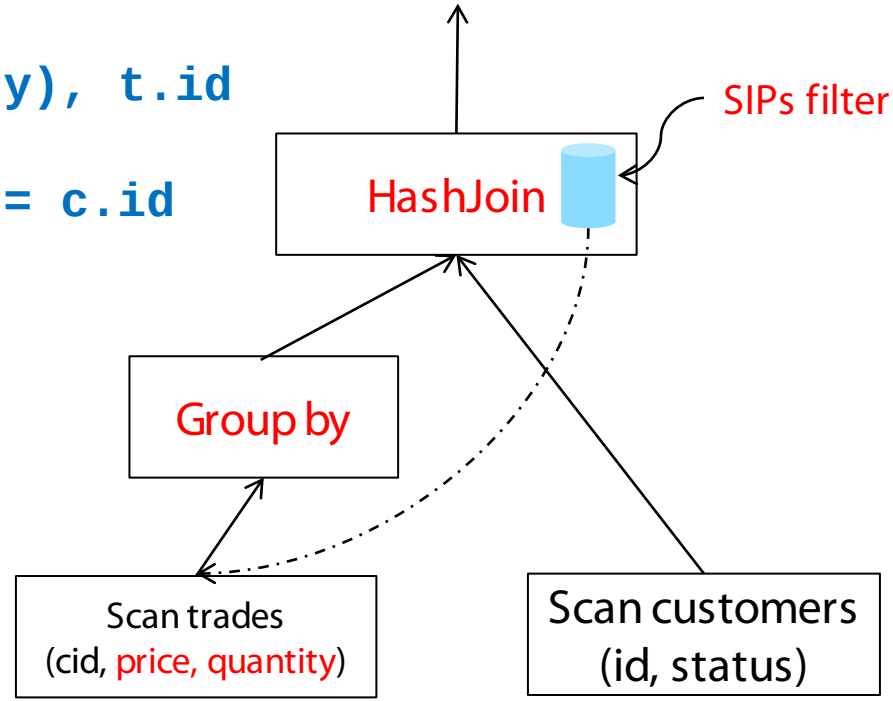


SIPs

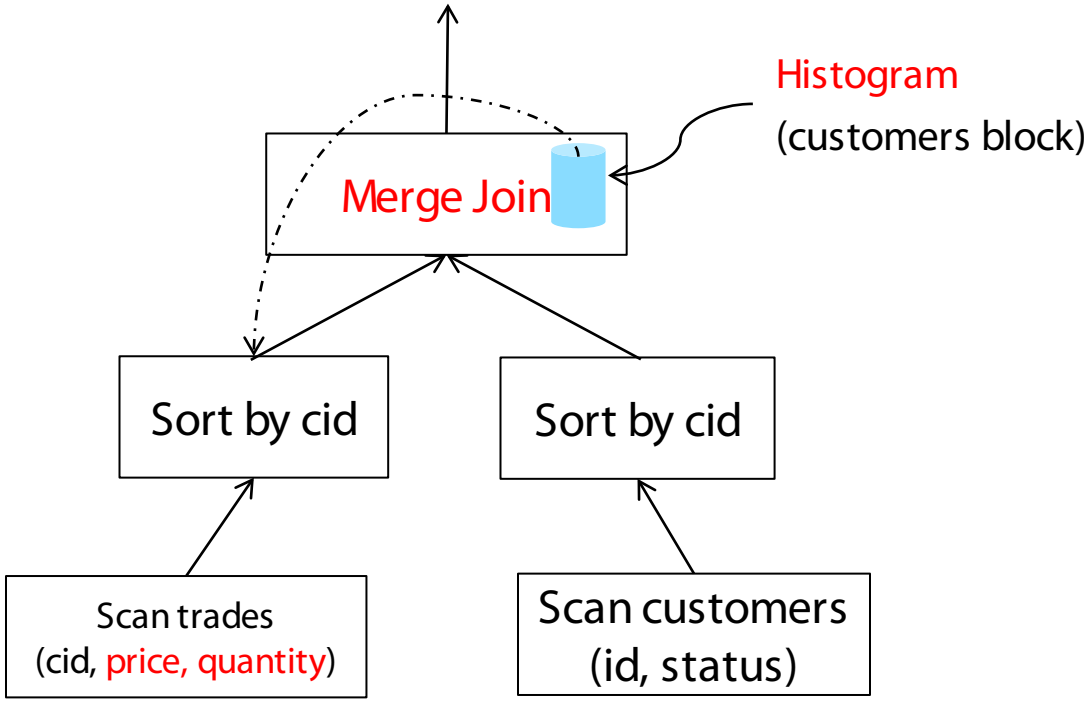


SIPs (pre-pass aggregation)

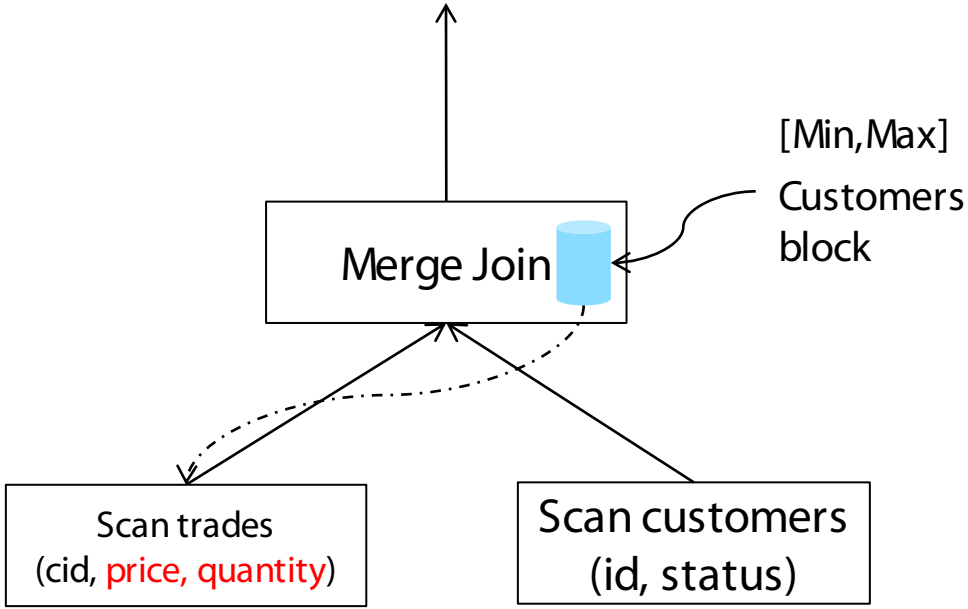
```
SELECT AVG(t.price*t.quantity), t.id
    FROM trades t
    JOIN customer c ON t.cid = c.id
   WHERE c.status = 'SPECIAL'
GROUP BY t.id;
```



SIPs (join spills)



SIPs – Merge Join



EM / LM / SIPs

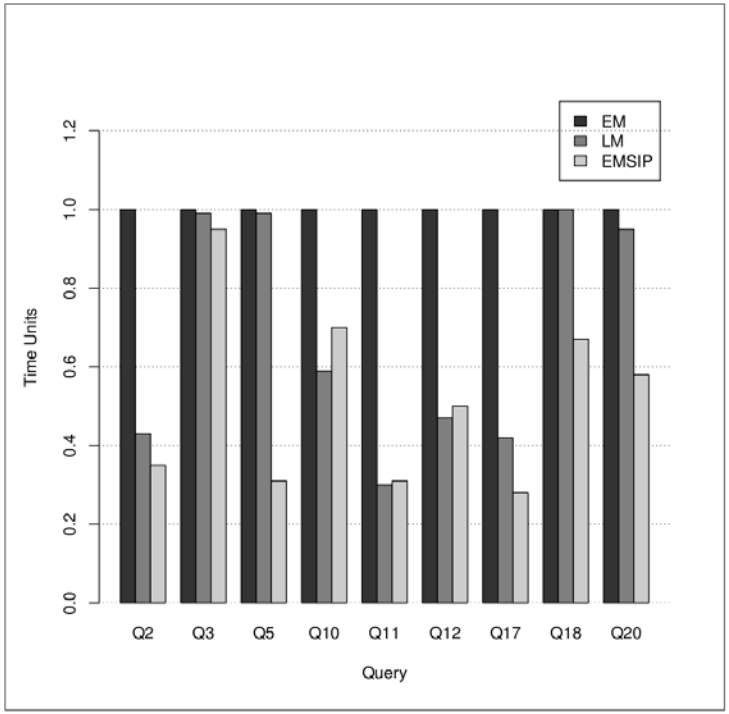
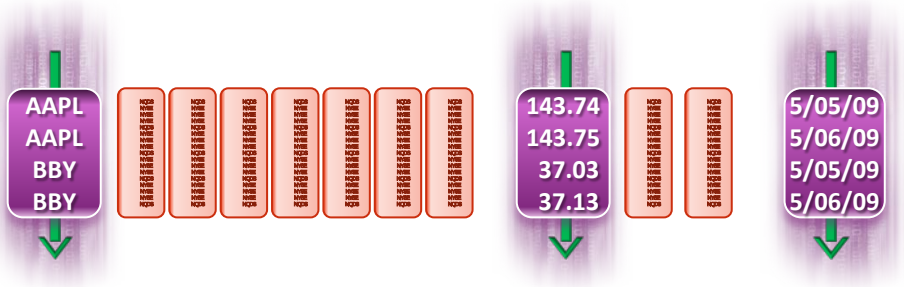


Fig. 10. Normalized Query Times

Column Store – Column-Based Disk I/O

Typical FinServ price per stock for 1 day

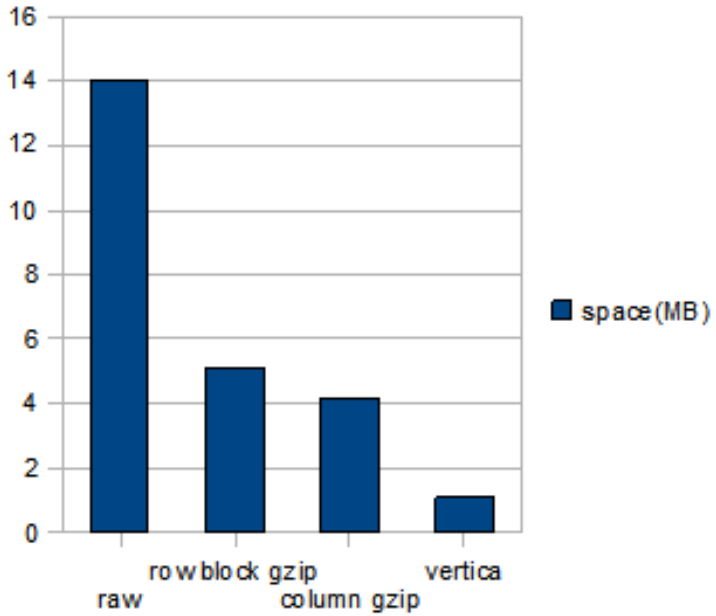
Column Store - Reads 3 columns



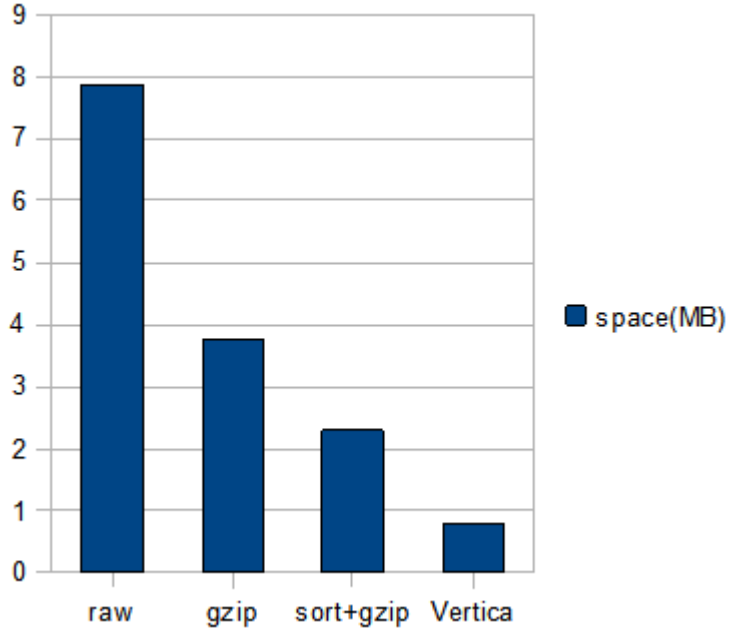

```
SELECT AVG(price)
FROM tickstore
WHERE
symbol = 'AAPL"  AND date = '5/06/09'
```

Row Store - Reads all columns

Compression

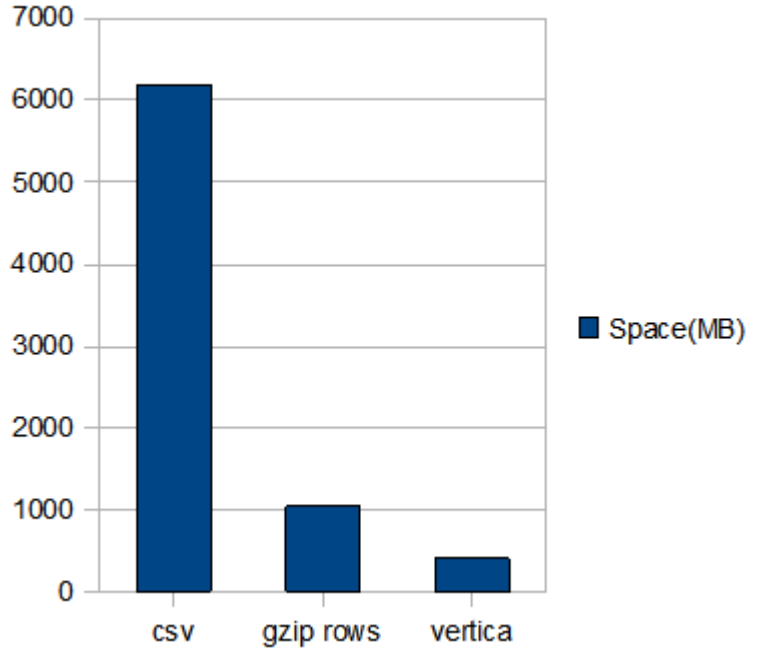


Compression + sorting



Compression on real customer data

- csv file of 200 million metric/meter/time/value rows
- raw
  - 6200 MB
  - ~31 bytes / row
- gzip
  - 1050 MB
  - ~5.25 bytes / row
- vertica
  - 418 MB
  - 2+ bytes / row
  - 15:1 compression



Projections

TABLES

trades(SYM, TDATE,PRICE,BROKER,EXCHANGE,PER_CHANGE,QUANTITY)

PROJECTIONS

trades_by_date(SYM, TDATE,PRICE,QUANTITY | TDATE)

trades_by_sym(SYM,BROKER,EXCHANGE,PER_CHANGE | SYM,TDATE)

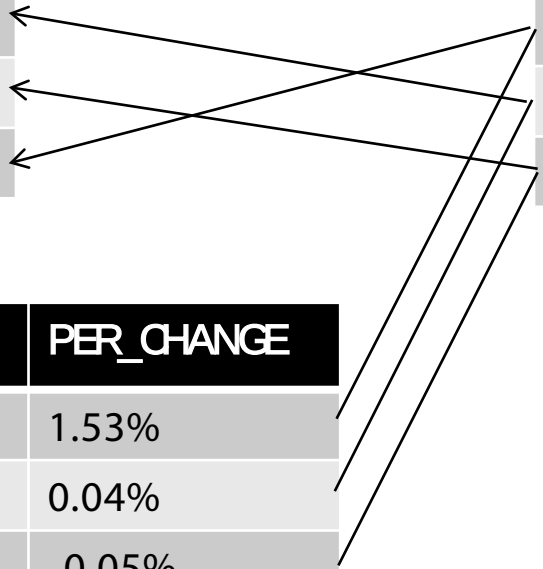
- **Segmented by range over sort order**

Projections – join index

SYM	DATE	PRICE	QUANTITY
ACME	2/10/14	$43.21	800
XYZ	2/11/14	$23.02	525
ACME	2/11/14	$43.88	600

SID	KEY
1	3
1	1
N	2

SYM	BROKER	EXCHANGE	PER_CHANGE
ACME	ML	NYSE	1.53%
ACME	VAN	NYSE	0.04%
XYZ	ML	NYSE	-0.05%



Projections - Super

TABLES

trades(SYM, TDATE,PRICE,BROKER,EXCHANGE,PER_CHANGE,QUANTITY)

PROJECTIONS

trades_super(SYM, TDATE,PRICE,BROKER,EXCHANGE,PER_CHANGE, QUANTITY, | TDATE)

trades_by_sym(SYM,BROKER,EXCHANGE,PER_CHANGE | SYM)

- **Segmented by an expression of your choosing, over a set of columns of your choosing**

Projections - Deletes

PROJECTIONS

trades_super(SYM, TDATE,PRICE,BROKER,EXCHANGE,PER_CHANGE, QUANTITY, | TDATE)

trades_by_sym(SYM,BROKER,EXCHANGE,PER_CHANGE | SYM)

DELETE FROM trades WHERE TDATE = 2/13/14;

How do we delete from trades_by_sym?

Projections – fault tolerance

trades_super1(SYM, TDATE,PRICE,BROKER,EXCHANGE,PER_CHANGE, QUANTITY, | TDATE, SYM)
trades_super2(SYM, TDATE,PRICE,BROKER,EXCHANGE,PER_CHANGE, QUANTITY, | SYM)

```
  SELECT sym, MAX(price)
    FROM trades
   WHERE tdate BETWEEN '02/11/14' AND '02/11/14'
GROUP BY sym;
```

Projections – fault tolerance

trades_super1 | TDATE,SYM

DATE	SYM	PRICE	BROKER	EXCHANGE	PER_CHANGE	QUANTITY
2/10/14	ACME	$43.21	VAN	NYSE	1.53%	600
…	…	…	…	…	…	…
2/11/14	XYZ	$23.02	ML	NYSE	-0.05%	525
2/11/14	ACME	$43.88	ML	NYSE	0.04%	800
…	…	…	…	…	…	…

trades_super2 | SYM

SYM	DATE	PRICE	BROKER	EXCHANGE	PER_CHANGE	QUANTITY
ACME	2/10/14	$43.21	VAN	NYSE	1.53%	600
ACME	2/11/14	$43.88	ML	NYSE	0.04%	800
…	…	…	…	…	…	…
XYZ	2/11/14	$23.02	ML	NYSE	-0.05%	525
…	…	…	…	…	…	…

Projections – recovery

trades_super1

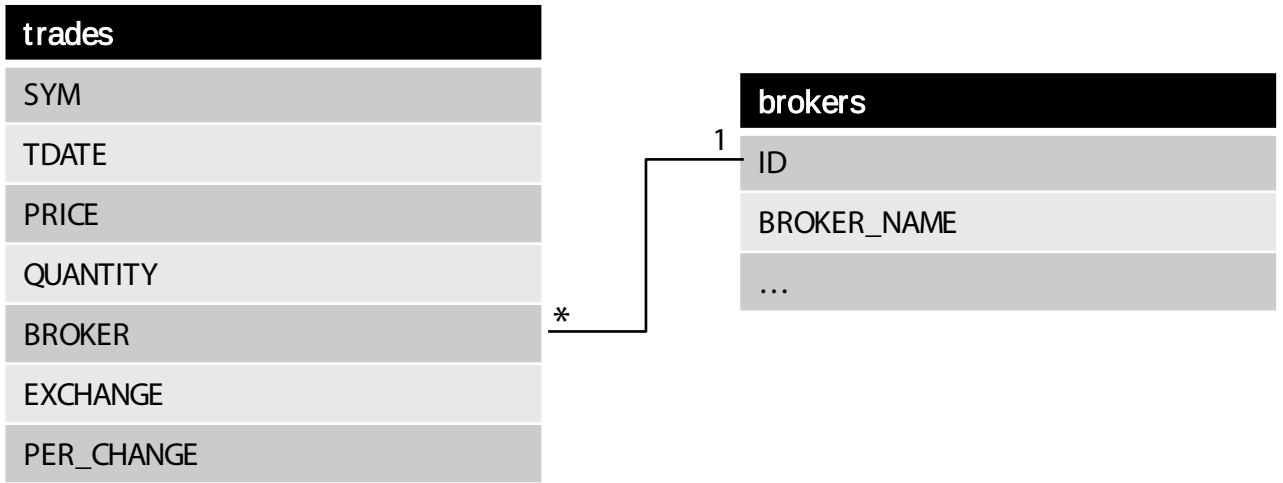
DATE	SYM	PRICE	BROKER	EXCHANGE	PER_CHANGE	QUANTITY
2/10/14	ACME	$43.21	VAN	NYSE	1.53%	600
…	…	…	…	…	…	…
2/11/14	XYZ	$23.02	ML	NYSE	-0.05%	525
2/11/14	ACME	$43.88	ML	NYSE	0.04%	800
…	…	…	…	…	…	…

trades_super2

SYM	DATE	PRICE	BROKER	EXCHANGE	PER_CHANGE	QUANTITY
ACME	2/10/14	$43.21	VAN	NYSE	1.53%	600
ACME	2/11/14	$43.88	ML	NYSE	0.04%	800
…	…	…	…	…	…	…
XYZ	2/11/14	$23.02	ML	NYSE	-0.05%	525
…	…	…	…	…	…	…

Projections – Pre-Join

```
SELECT broker_name, SUM(price)
  FROM trades t JOIN brokers b ON t.BROKER = b.ID
GROUP BY broker_name;
```



Projections – Pre-Join

trades_pjp

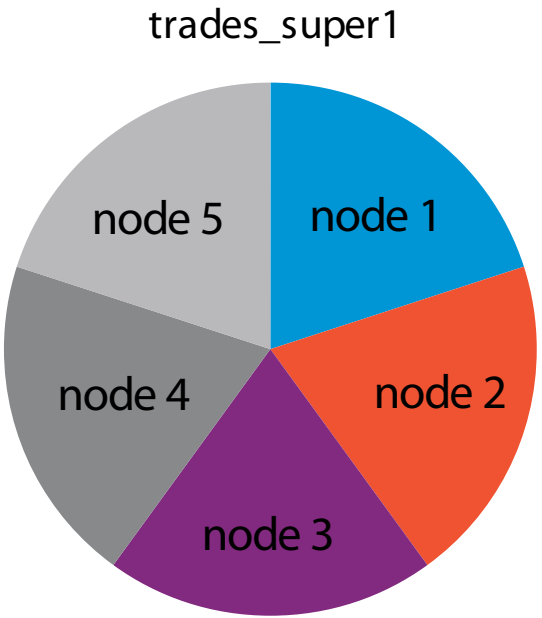
SYM	TDATE	PRICE	BROKER	EXCHANGE	PER_CHANGE	QUANTITY	BROKER_NAME
ACME	2/10/14	$43.21	VAN	NYSE	1.53%	600	Vanguard
ACME	2/11/14	$43.88	ML	NYSE	0.04%	800	Merrill
XYZ	2/11/14	$23.02	ML	NYSE	-0.05%	525	Merrill

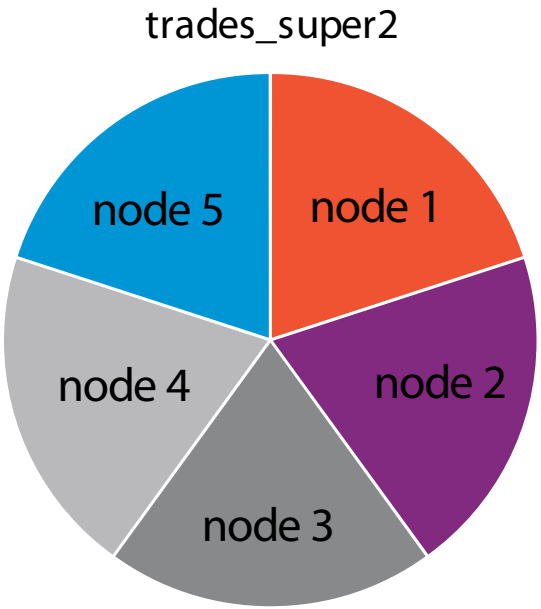
Projections – Pre-Join

trades_pjp | BROKER_NAME

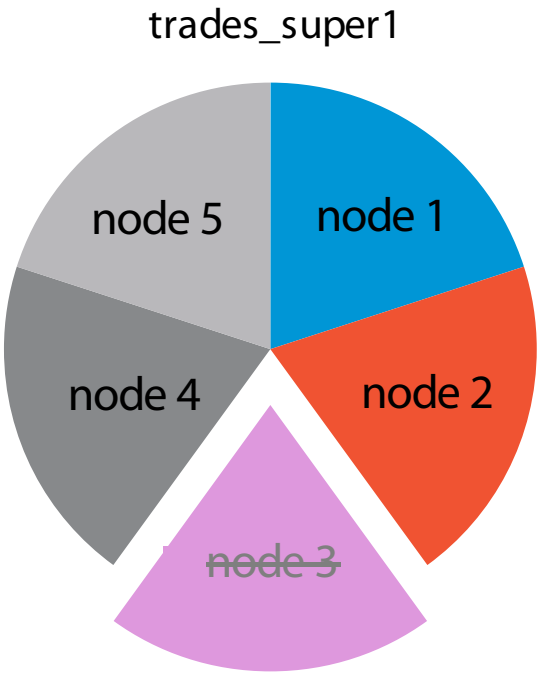
BROKER_NAME	SYM	TDATE	PRICE	BROKER	EXCHANGE	PER_CHANGE	QUANTITY
Merrill	ACME	2/11/14	$43.88	ML	NYSE	0.04%	800
Merrill	XYZ	2/11/14	$23.02	ML	NYSE	-0.05%	525
Vanguard	ACME	2/10/14	$43.21	VAN	NYSE	1.53%	600

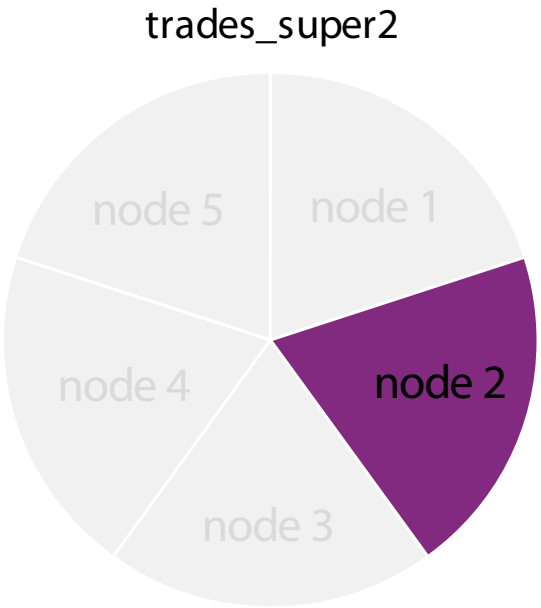
Fault tolerance



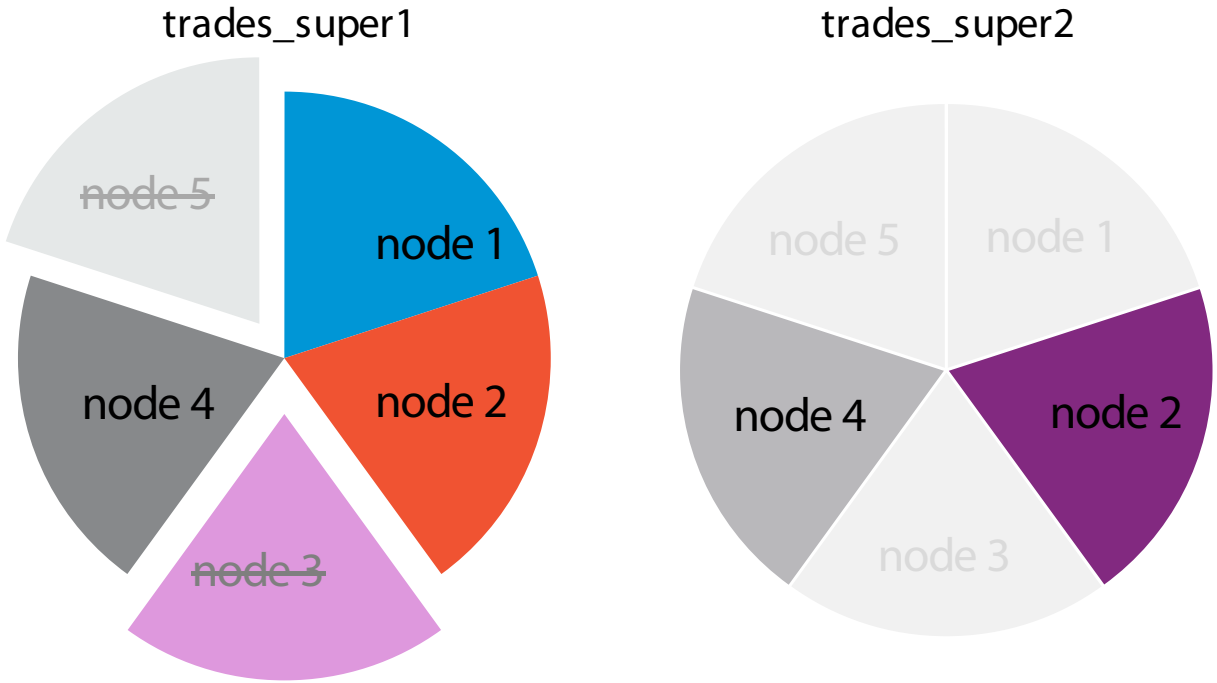


Fault tolerance

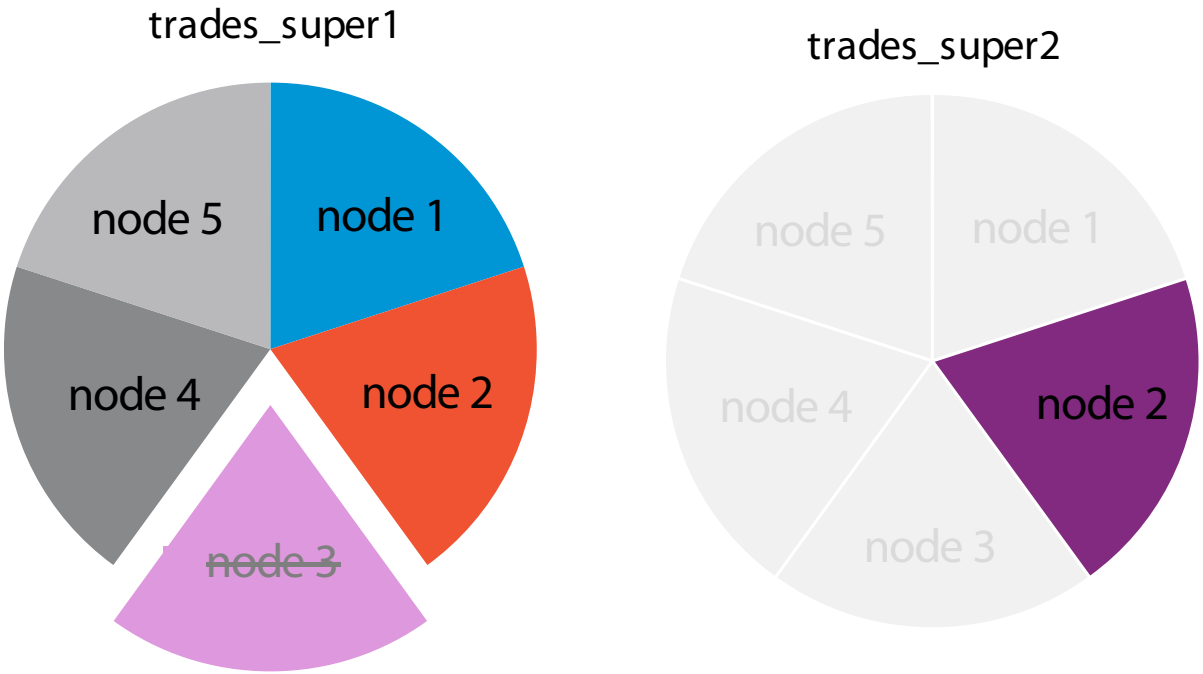




Fault tolerance



Fault tolerance



Fault tolerance

Query Scans p1

proj	node 1	node 2	~~node 3~~	node 4	node 5
p1	s1	s2	s3	s4	s5
p2	s2	s3	s4	s5	s1

Fault tolerance

Query Scans p1

proj	node 1	node 2	~~node 3~~	node 4	node 5
p1	s1	s2	s3	s4	s5
p2	s2	s3	s4	s5	s1

Fault tolerance

Query Scans p1

proj	node 1	node 2	~~node 3~~	node 4	node 5
p1	s1	s2	s3	s4	s5
p2	s2	s3	s4	s5	s1

Query Scans p1

proj	node 1	node 2	~~node 3~~	node 4	node 5
p1	s1	s2	s3	s4	s5
p2	s2	s3	s4	s5	s1

Fault tolerance

Units of Work By Node (4 queries)

node 1	node 2	node 3	node 4	node 5
4	8	0	4	4

Fault tolerance

Units of Work By Node (4 queries)

node 1	node 2	node 3	node 4	node 5
4	4	4	4	4

Fault tolerance

Units of Work By Node (4 queries)

node 1	node 2	node 3	node 4	node 5
4	4	4	4	4

Units of Work By Node (4 queries)

node 1	node 2	node 3	node 4	node 5
5	5	0	5	5

THANKS!