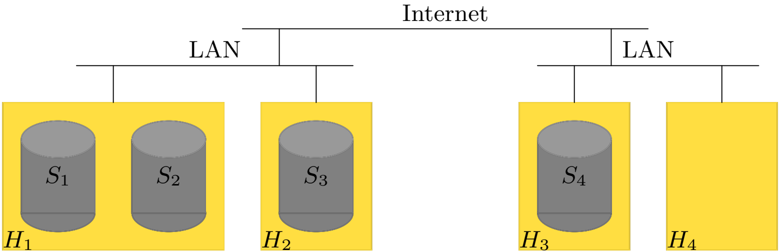
Distributed Query Processing

P.J. McBrien

Imperial College London

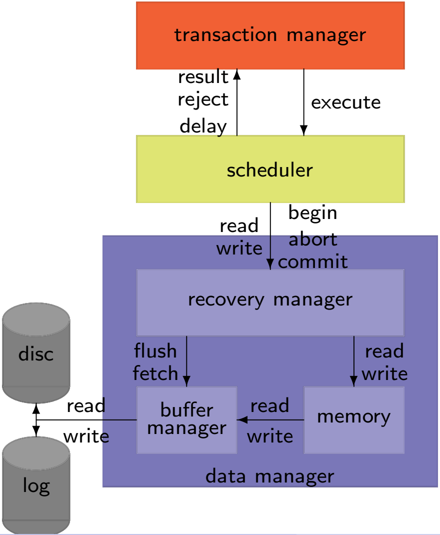
Distributed Databases



Need to distribute transactions of databases that might have

- Remote Tables
- Fragmentation
- Replication
- Migration

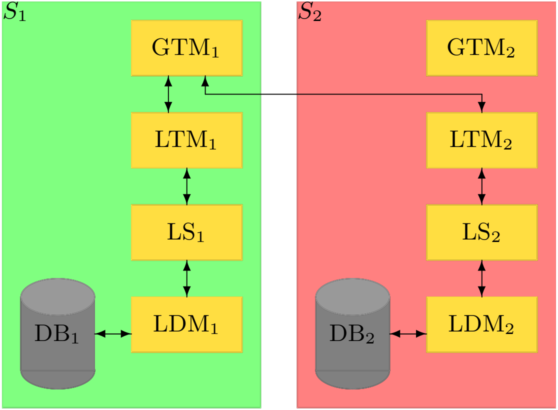
Single-Site Query Processing



Three Main Components of a DBMS

- Transaction Manager: query processor plans and translates queries to set of primitive operations
- Schedular: schedules the primitive operators to obey ACID properties
- Data Manager: efficient use of memory, maintain durability of transactions

Distributed Query Processing



DDB Architecture

- Each site runs **local transaction manager** (**LTM**), scheduler, and data manager
- One or more sites run a **global transaction manager** (**GTM**)
- Slow part of distributed query processing is network communication

Distribution of TP: GTM Plans Execution

```
BEGIN TRANSACTION T1
    UPDATE branch
    SET cash=cash-10000
    WHERE sortcode=56

    UPDATE branch
    SET cash=cash+10000
    WHERE sortcode=34
COMMIT TRANSACTION T1
```

GTM

$\text{branch}_1 = \sigma_{\text{sortcode}<50}\ \text{branch}$

$\text{branch}_2 = \sigma_{\text{sortcode}\geq 50}\ \text{branch}$

branch

<u>sortcode</u>	bname	cash
34	'Goodge St'	8900.67

S_1

branch

<u>sortcode</u>	bname	cash
56	'Wimbledon'	94340.45
67	'Strand'	34005.00

S_2

- GTM must transform transaction into sub transactions for each site
  - remote tables/fragmentation → breakup transaction into fragments
  - replication → read only transactions goto one site, read/write transactions to all sites
  - migration → change query plans with migration

Distribution of TP: Example 1

```
BEGIN TRANSACTION T1
    UPDATE branch
    SET cash=cash-10000
    WHERE sortcode=56

    UPDATE branch
    SET cash=cash+10000
    WHERE sortcode=34
COMMIT TRANSACTION T1
```

GTM

```
BEGIN TRANSACTION T1_1
    UPDATE branch
    SET cash=cash+10000
    WHERE sortcode=34
COMMIT TRANSACTION T1_1
```

LTM_1

```
BEGIN TRANSACTION T1_2
    UPDATE branch
    SET cash=cash-10000
    WHERE sortcode=56
COMMIT TRANSACTION T1_2
```

LTM_2

Distribution of TP: Example 2

```
BEGIN TRANSACTION T4
    SELECT SUM(cash)
    FROM branch
COMMIT TRANSACTION T4
```

GTM

```
BEGIN TRANSACTION T4_1
    SELECT @X=SUM(cash)
    FROM branch
    WHERE sortcode<50
COMMIT TRANSACTION T4_1
```

LTM_1

```
BEGIN TRANSACTION T4_2
    SELECT @Y=SUM(cash)
    FROM branch
    WHERE sortcode>=50
COMMIT TRANSACTION T4_2
```

LTM_2

Distribution of TP: Example 2

```
BEGIN TRANSACTION T4_1
   SELECT @X=SUM(cash)
   FROM branch
   WHERE sortcode<50
COMMIT TRANSACTION T4_1
```

$\Rightarrow$ LTM_1

```
BEGIN TRANSACTION T4_2
   SELECT @Y=SUM(cash)
   FROM branch
   WHERE sortcode>=50
COMMIT TRANSACTION T4_2
```

$\Rightarrow$ LTM_2

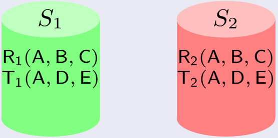
$\Rightarrow$

```
BEGIN TRANSACTION T4
   SELECT @X+@Y
COMMIT TRANSACTION T4
```

$\Rightarrow$ GTM

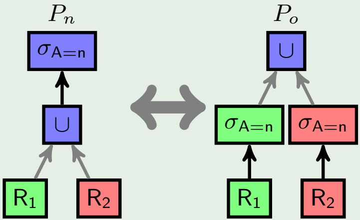
RA Equivalences: Selects over Horizontal Fragmentation (1)

Query Pattern



S_3: $\sigma_{A=n}$ R

Query Plan

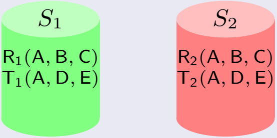


Query Cost

$\text{NDB}(P_n) = |\text{R}|.\text{RowSize}(\text{R})$
$\text{NDB}(P_o) = |\sigma_{A=n}\,\text{R}|.\text{RowSize}(\text{R})$

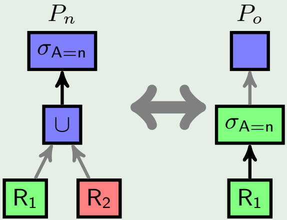
RA Equivalences: Selects over Horizontal Fragmentation (2)

Query Pattern



S_3: $\sigma_{A=n}\,R$
$n \in K_1$
$R_1 = \sigma_{A \in K_1}\,R$
$R_2 = \sigma_{A \in K_2}\,R$

Query Plan



Query Cost

$NDB(P_n) = |R|.RowSize(R)$
$NDB(P_o) = |\,\sigma_{A=n}\,R|.RowSize(R)$

SQL DML: An Implementation of the RA

SQL SELECT statements: Rough Equivalence to RA

```
SELECT A1, ..., An
FROM   R1, ..., Rm
WHERE  P1
AND    ...
AND    Pk
```

$\Leftrightarrow$ $\pi_{A_1,\ldots,A_n}\sigma_{P_1\wedge\ldots\wedge P_k}R_1 \times \ldots \times R_m$

SQL SELECT implements RA π, σ and $\times$

$\pi_{bname,no}\,\sigma_{branch.sortcode=account.sortcode\wedge account.type=\text{'current'}}(branch \times account)$

```sql
SELECT branch.bname,
       account.no
FROM   account, branch
WHERE  account.sortcode=branch.sortcode
AND    account.type='current'
```

Horizontal Fragmentation In SQL

Distributing a query over Horizontal Fragmentation

- Selection of horizontal fragment decided using σ
- WHERE clause (σ) may determine only certain horizontal fragments are used for a query

account$_1$				
no	type	cname	rate	sortcode
100	'current'	'McBrien, P.'	NULL	67
103	'current'	'Boyd, M.'	NULL	34
107	'current'	'Poulovassilis, A.'	NULL	56
125	'current'	'Bailey, J.'	NULL	56

account$_2$				
no	type	cname	rate	sortcode
101	'deposit'	'McBrien, P.'	5.25	67
119	'deposit'	'Poulovassilis, A.'	5.50	56

- $\text{account}_1 = \sigma_{\text{type}=\text{'}current\text{'}}\text{account}$
- $\text{account}_2 = \sigma_{\text{type}\neq\text{'}current\text{'}}\text{account}$

Query Distribution

```
SELECT account.rate
       branch.bname
FROM   account
       JOIN branch
       USING(sortcode)
WHERE  account.type='deposit'
AND    account.no=101
```

- Only send to S_2

Vertical Fragmentation in SQL

Distributing a query over Vertical Fragmentation

- Selection of vertical fragmentation decided using π
- SELECT clause (π) may determine only certain vertical fragments are used, but must ensure JOIN and WHERE clauses (σ) can be processed.

account_1			
no	type	rate	sortcode
100	'current'	NULL	67
101	'deposit'	5.25	67
103	'current'	NULL	34
107	'current'	NULL	56
119	'deposit'	5.50	56
125	'current'	NULL	56

account_2	
no	cname
100	'McBrien, P.'
101	'McBrien, P.'
103	'Boyd, M.'
107	'Poulovassilis, A.'
119	'Poulovassilis, A.'
125	'Bailey, J.'

Query Distribution

```
SELECT account.rate
       branch.bname
FROM   account
       JOIN branch
       USING(sortcode)
WHERE  account.type='deposit'
AND    account.no=101
```

- Only send to S_1

- $\text{account}_1 = \pi_{\text{no,type,rate,sortcode}}\text{account}$
- $\text{account}_2 = \pi_{\text{no,cname}}\text{account}$

Worksheet: SQL Queries over Fragmented Data

S_1

account			
no	type	rate?	sortcode
100	'current'	NULL	67
103	'current'	NULL	34
107	'current'	NULL	56
125	'current'	NULL	56

S_3

account	
no	cname
100	'McBrien, P.'
103	'Boyd, M.'
107	'Poulovassilis, A.'
125	'Bailey, J.'

S_2

account			
no	type	rate?	sortcode
101	'deposit'	5.25	67
119	'deposit'	5.50	56

S_4

account	
no	cname
101	'McBrien, P.'
119	'Poulovassilis, A.'

$$\text{account}_1 = \pi_{\text{no},\text{type},\text{rate},\text{sortcode}}(\sigma_{\text{type}=\text{current}}\text{account})$$
$$\text{account}_2 = \pi_{\text{no},\text{type},\text{rate},\text{sortcode}}(\sigma_{\text{type}\neq\text{current}}\text{account})$$
$$\text{account}_3 = \pi_{\text{no},\text{cname}}(\sigma_{\text{type}=\text{current}}\text{account})$$
$$\text{account}_4 = \pi_{\text{no},\text{cname}}(\sigma_{\text{type}\neq\text{current}}\text{account})$$

Worksheet: Distribution of Transactions (1)

LTM2:

```
BEGIN TRANSACTION TA_2
   UPDATE account
   SET    rate=5.5
   WHERE  type='deposit'
COMMIT TRANSACTION TA_2
```

GTM:

```
SELECT *
FROM   LTM2
```

Worksheet: Distribution of Transactions (2)

LTM3:

```
BEGIN TRANSACTION TB_3
   SELECT no
   FROM   account
   WHERE  cname='McBrien, P.'
COMMIT TRANSACTION TB_3
```

LTM4:

```
BEGIN TRANSACTION TB_4
   SELECT no
   FROM   account
   WHERE  cname='McBrien, P.'
COMMIT TRANSACTION TB_4
```

GTM:

```
SELECT *
FROM   LTM3
UNION ALL
SELECT *
FROM   LTM4
```

Worksheet: Distribution of Transactions (3)

LTM2

```
BEGIN TRANSACTION TC_2
   SELECT no,rate
   FROM   account
   WHERE  type='deposit'
COMMIT TRANSACTION TC_2
```

LTM4

```
BEGIN TRANSACTION TC_4
   SELECT no,cname
   FROM   account
COMMIT TRANSACTION TC_4
```

GTM

```
SELECT *
FROM   LTM2 NATURAL JOIN LTM4
```

Worksheet: Distribution of Transactions (4)

LTM1

```
BEGIN TRANSACTION TD_1
   SELECT COALESCE(SUM(rate),0.0) AS total_rate,
          COUNT(rate) AS no_rate
   FROM   account
COMMIT TRANSACTION TD_1
```

LTM2

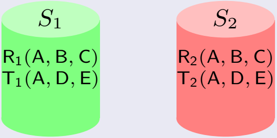
```
BEGIN TRANSACTION TD_2
   SELECT COALESCE(SUM(rate),0.0) AS total_rate,
          COUNT(rate) AS no_rate
   FROM   account
COMMIT TRANSACTION TD_2
```

GTM:

```
SELECT (LTM1.total_rate+LTM2.total_rate)/
           (LTM1.no_rate+LTM2.no_rate)
FROM   LTM1,LTM2
```

RA Equivalences: Joins over Derived Horizontal Fragmentation

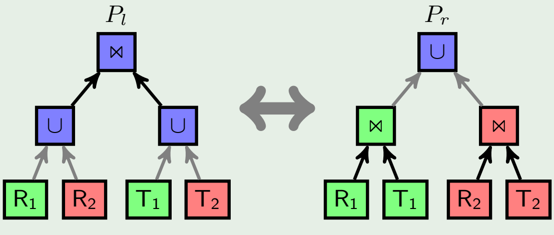
Query Pattern



S_3: $R \bowtie T$

$T_n = T \ltimes R_n$

Query Plan



Query Cost

$\text{NDB}(P_l) =$
$|R|.\text{RowSize}(R) + |T|.\text{RowSize}(T)$
$\text{NDB}(P_r) =$
$|R \bowtie T|.\text{RowSize}(R \bowtie T)$

Executing Distributed Joins over Remote Tables

S_1

account			
no	type	...	sortcode
100	'current'		67
101	'deposit'		67
103	'current'		34
107	'current'		56
119	'deposit'		56
125	'current'		56

S_2

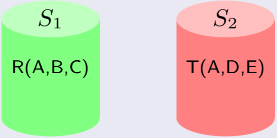
movement			
mid	no	amount	tdate
1000	100	2300.00	5/1/1999
1001	101	4000.00	5/1/1999
1002	100	-223.45	8/1/1999
1004	107	-100.00	11/1/1999
1005	103	145.50	12/1/1999
1006	100	10.23	15/1/1999
1007	107	345.56	15/1/1999
1008	101	1230.00	15/1/1999
1009	119	5600.00	18/1/1999

Suppose we want to execute

$$S_1 : \text{account} \bowtie \text{movement}$$

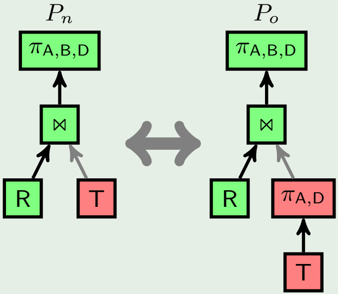
RA Equivalences: Push project inside join

Query Pattern



S_1: $\pi_{\mathsf{A,B,D}}(\mathsf{R} \bowtie \mathsf{T})$

Query Plan

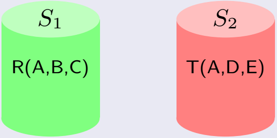


Query Cost

$\mathsf{NDB}(P_n) = |\mathsf{T}|.\mathsf{RowSize}(\mathsf{T})$
$\mathsf{NDB}(P_o) = |\,\pi_{\mathsf{A,D}}\,\mathsf{T}|.\mathsf{RowSize}(\pi_{\mathsf{A,D}}\,\mathsf{T})$

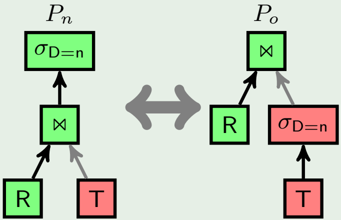
RA Equivalences: Push select inside join

Query Pattern



S_1: $\sigma_{\mathsf{D=n}}(\mathsf{R} \bowtie \mathsf{T})$

Query Plan



Query Cost

$\mathsf{NDB}(P_n) = |\mathsf{T}|.\mathsf{RowSize}(\mathsf{T})$
$\mathsf{NDB}(P_o) = |\,\sigma_{\mathsf{D=n}}\,\mathsf{T}|.\mathsf{RowSize}(\mathsf{T})$

Data Metadata

branch		
sortcode	bname	cash
56	'Wimbledon'	94340.45
34	'Goodge St'	8900.67
67	'Strand'	34005.00

movement			
mid	no	amount	tdate
1000	100	2300.00	5/1/1999
1001	101	4000.00	5/1/1999
1002	100	-223.45	8/1/1999
1004	107	-100.00	11/1/1999
1005	103	145.50	12/1/1999
1006	100	10.23	15/1/1999
1007	107	345.56	15/1/1999
1008	101	1230.00	15/1/1999
1009	119	5600.00	18/1/1999

account				
no	type	cname	rate?	sortcode
100	'current'	'McBrien, P.'	NULL	67
101	'deposit'	'McBrien, P.'	5.25	67
103	'current'	'Boyd, M.'	NULL	34
107	'current'	'Poulovassilis, A.'	NULL	56
119	'deposit'	'Poulovassilis, A.'	5.50	56
125	'current'	'Bailey, J.'	NULL	56

key branch(sortcode)
key branch(bname)
key movement(mid)
key account(no)

movement(no) $\overset{fk}{\Rightarrow}$ account(no)

account(sortcode) $\overset{fk}{\Rightarrow}$ branch(sortcode)

Metadata

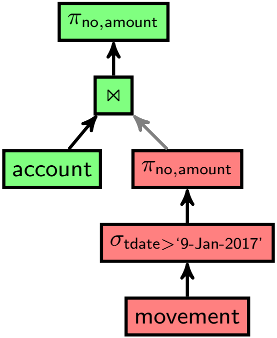
column	
name	size
account	6
amount	4
bname	32
branch	3

column	
name	size
cname	32
mid	4
movement	9
no	4

column	
name	size
rate	4
sortcode	1
tdate	4
type	8

$|\text{account}| = 6$
$|\text{branch}| = 3$
$|\text{movement}| = 9$

Quiz 1: Number of Network Data Bytes Required for a Query

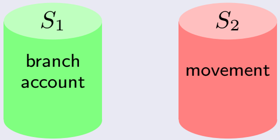


What is the number of **network data bytes (NDB)** required for the above distributed query plan, if the selectivity of the query $\sigma_{\text{tdate}>\text{'9-Jan-2017'}}$ is $\frac{2}{3}$?

A	B	C	D
8	16	48	144

Example Comparison (1)

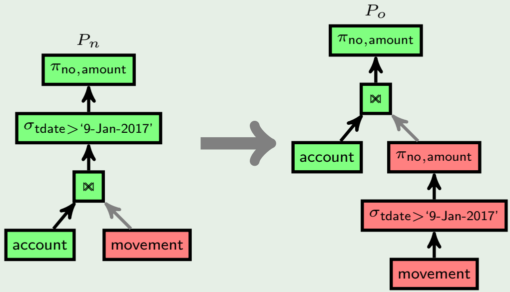
Query Pattern



S_1:

```sql
SELECT account.no,
       movement.amount
FROM   account
       JOIN movement
       USING (no)
WHERE  tdate>'9-Jan-2017'
```

Query Plan



Example Comparison (1): Distributed Query Plans

Direct Execution of Query

$$\begin{aligned} S_2\, to\, S_1 &= \mathsf{RowSize(movement)} \times |\mathsf{movement}| \\ &= (4+4+4+4) \times 9 \\ &= 144 \end{aligned}$$

Project and Selects Pushed inside Join

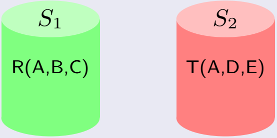
$$\begin{aligned} S_2\, to\, S_1 &= \mathsf{RowSize}(\pi_{\mathsf{no,amount}}\,\mathsf{movement}) \times |\pi_{\mathsf{no,amount}}\ \sigma_{\mathsf{tdate}>\text{'9-Jan-2017'}}\ \mathsf{movement}| \\ &= (4+4) \times 6 \\ &= 48 \end{aligned}$$

Optimisation Rule

Pushing selects and projections inside a join will always make the query more efficient (unless the selections or projections where 'trivial', in which case it will make no difference).

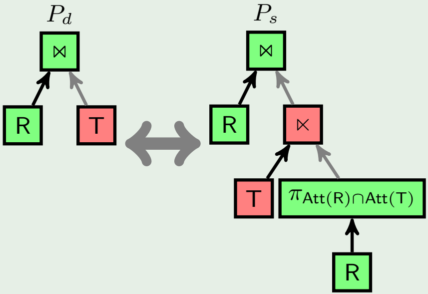
RA Equivalences: Direct and Semi Joins

Query Pattern



S_1: R $\bowtie$ T

Query Plan

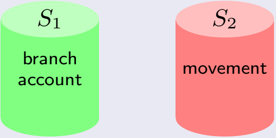


Query Cost

$\mathsf{NDB}(P_d) = |\mathsf{T}|.\mathsf{RowSize}(\mathsf{T})$
$\mathsf{NDB}(P_s) =$
$|\, \pi_{\mathsf{Att(R)} \cap \mathsf{Att(T)}}\ \mathsf{R}|.\mathsf{RowSize}(\pi_{\mathsf{Att(R)} \cap \mathsf{Att(T)}}\ \mathsf{R}) +$
$|\mathsf{T} \ltimes \mathsf{R}|.\mathsf{RowSize}(\mathsf{T})$

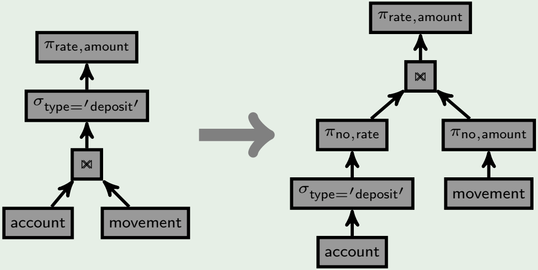
Example Comparison (2)

Query Pattern



S_1:
```
SELECT account.rate,
       movement.amount
FROM   account
       JOIN movement
       USING (no)
WHERE  account.type='deposit'
```

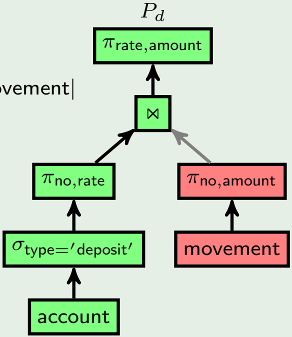
Query Plan



Example Comparison (2): Distributed Query Plans

Direct Join

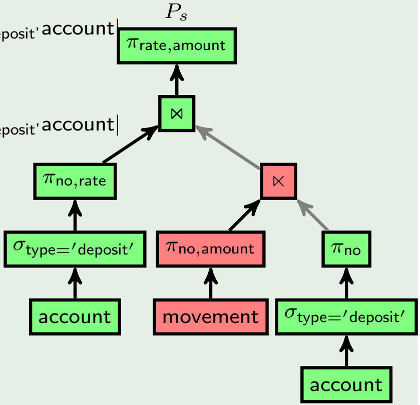
$$\begin{array}{rcl} S_2\ to\ S_1 & = & \mathsf{RowSize}(\pi_{\mathsf{no,amount}}\mathsf{movement}) \times |\pi_{\mathsf{no,amount}}\mathsf{movement}| \\ & = & (4+4) \times 9 \\ & = & 72 \end{array}$$



Example Comparison (2): Distributed Query Plans

Semi-Join

$$
\begin{aligned}
S_1 \; to \; S_2 &= \mathsf{RowSize}(\pi_{\mathsf{no}}\mathsf{account}) \times |\pi_{\mathsf{no}}\sigma_{\mathsf{type}=\text{`}\mathsf{deposit}\text{'}}\mathsf{account}| \\
&= 4 \times 2 \\
&= 8 \\
S_2 \; to \; S_1 &= \mathsf{RowSize}(\pi_{\mathsf{no,amount}}\mathsf{movement}) \times \\
&\qquad |\pi_{\mathsf{no,amount}}\mathsf{movement} \ltimes \sigma_{\mathsf{type}=\text{`}\mathsf{deposit}\text{'}}\mathsf{account}| \\
&= (4+4) \times 3 \\
&= 24 \\
\mathbf{total} &= 32
\end{aligned}
$$



Quiz 2: General Efficiency of Direct Join compared to Semi-Join

Comparing the Direct Join plan P_d with the Semi-Join plan P_s, for any query Q is it the case that

A

Always

NDB(P_d) $\leq$ NDB(P_s)

B

Always

NDB(P_d) $\geq$ NDB(P_s)

C

Sometimes

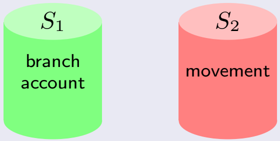
NDB(P_d) $\leq$ NDB(P_s)

Sometimes

NDB(P_d) $\geq$ NDB(P_s)

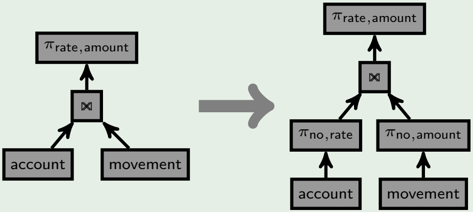
Example Comparison (3)

Query Pattern



S_1:
```
SELECT account.rate,
       movement.amount
FROM   account
       JOIN movement
       USING (no)
```

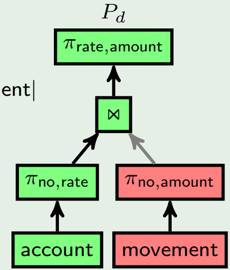
Query Plan



Example Comparison (3): Distributed Query Plans

Direct Join

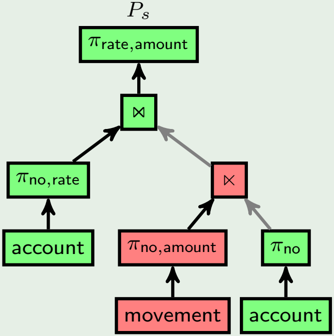
$$
\begin{aligned}
S_2\ to\ S_1 &= \mathsf{RowSize}(\pi_{\mathsf{no,amount}}\mathsf{movement}) \times |\pi_{\mathsf{no,amount}}\mathsf{movement}| \\
&= (4+4)\times 9 \\
&= 72
\end{aligned}
$$



Example Comparison (3): Distributed Query Plans

Semi-Join

$$
\begin{aligned}
S_1\ to\ S_2 &= \text{RowSize}(\pi_{\text{no}}\text{account}) \times |\pi_{\text{no}}\text{account}| \\
&= 4 \times 6 \\
&= 24 \\
S_2\ to\ S_1 &= \text{RowSize}(\pi_{\text{no,amount}}\text{movement}) \times \\
&\qquad |\pi_{\text{no,amount}}\text{movement} \ltimes \text{account}| \\
&= (4+4) \times 9 \\
&= 72 \\
\textbf{total} &= 96
\end{aligned}
$$



Worksheet: Joins over Remote Tables

directory		
telephone	name	charge
1000	Adams	10.00
1001	Jones	120.25
1002	Black	344.00
1003	Khan	243.50
1004	Smith	44.00
1005	Brown	-100.00
1006	Patel	200.00
1007	White	222.00

S_1

business	
telephone	rate
1002	A
1004	B
1006	B

S_2

column	bytes
telephone	5
name	6
charge	4
rate	1

Worksheet: $\pi_{\text{name,rate}}\sigma_{\text{charge}>300.00}\text{business} \bowtie \text{directory}$

(1) **Direct Join**

introduce $f_1 = \frac{|\sigma_{\text{charge}>300.00}\text{ directory}|}{|\text{directory}|} = \frac{1}{8}$

$$\begin{aligned} S_1 \rightarrow S_2 &= \text{RowSize}(\pi_{\text{telephone,name}}\text{ directory}) \times |\sigma_{\text{charge}>300.00}\text{ directory}| \\ &= (5+6) \times f_1 d \\ &= 11 f_1 d \text{ Bytes} \end{aligned}$$

(2) **Semi Join**

introduce $f_2 = \frac{|\sigma_{\text{charge}>300.00}\text{ directory} \ltimes \text{business}|}{|\text{business}|} = \frac{1}{3}$

$$\begin{aligned} S_2 \rightarrow S_1 &= \text{RowSize}(\pi_{\text{telephone}}\text{ directory}) \times |\text{business}| = 5b \\ S_1 \rightarrow S_2 &= \text{RowSize}(\pi_{\text{telephone,name}}\text{ directory}) \times |\sigma_{\text{charge}>300.00}\text{ directory} \ltimes \text{business}| \\ &= (5+6) \times f_2 b = 11 f_2 b \end{aligned}$$

Total data transfer$=5b + 11f_2b$ Bytes

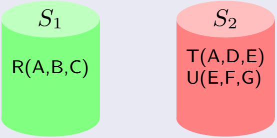
For current data, total data transfer $= 26$ Bytes

(3) $11f_1d = 5b + 11f_2b$

$\frac{b}{d} = \frac{11f_1}{(5+11f_2)} = 15.9\%$

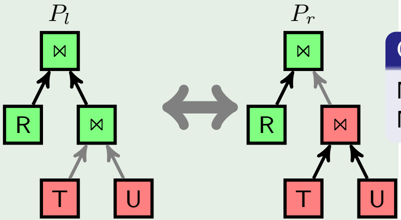
RA Equivalences: Local and Remote joins

Query Pattern



S_1: R $\bowtie$ T $\bowtie$ U

Query Plan

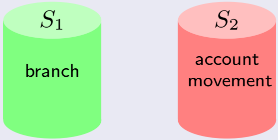


Query Cost

NDB(P_l) = |T|.RowSize(T) + |U|.RowSize(U)
NDB(P_r) = |T $\bowtie$ U|.RowSize(T $\bowtie$ U)

Example Comparison (4)

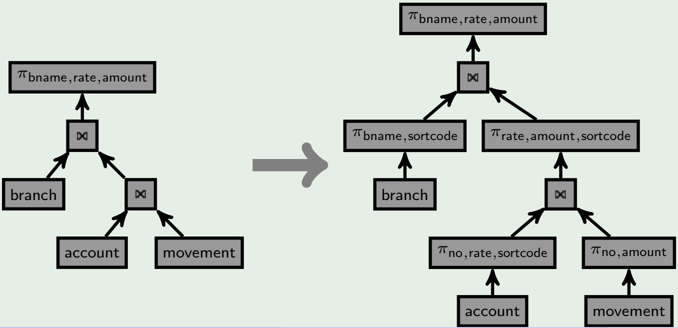
Query Pattern



S_1:

```
SELECT branch.bname,
       account.rate,
       movement.amount
FROM   account
       JOIN movement USING (no)
       JOIN branch USING (sortcode)
```

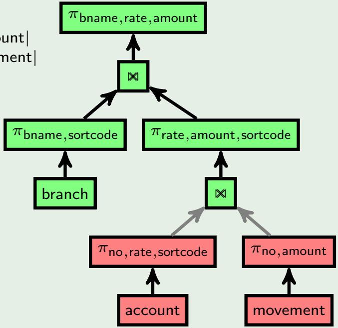
Query Plan



Example Comparison (4): Distributed Query Plans

Local Join

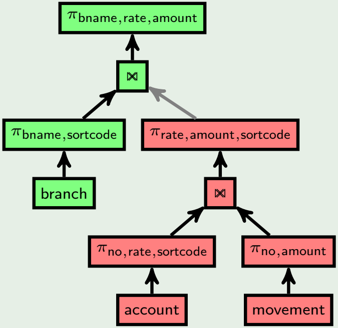
$$
\begin{aligned}
S_2\ to\ S_1 &= \mathsf{RowSize}(\pi_{\mathsf{no,rate,sortcode}}\mathsf{account}) \times |\pi_{\mathsf{no,rate,sortcode}}\mathsf{account}| \\
&= \quad \mathsf{RowSize}(\pi_{\mathsf{no,amount}}\mathsf{movement}) \times |\pi_{\mathsf{no,amount}}\mathsf{movement}| \\
&= (4+4+1)\times 6 + (4+4)\times 9 \\
&= 54 + 72 \\
&= 126
\end{aligned}
$$



Example Comparison (4): Distributed Query Plans

Remote Join

$$
\begin{array}{lcl}
S_2\ to\ S_1 & = & \mathsf{RowSize}(\pi_{\mathsf{rate,amount,sortcode}}\mathsf{account} \bowtie \mathsf{movement}) \times \\
 & = & \quad |\pi_{\mathsf{rate,amount,sortcode}}\mathsf{account} \bowtie \mathsf{movement}| \\
 & = & (4+4+1) \times 9 \\
 & = & 81
\end{array}
$$



Quiz 3: General Efficiency of Local Join compared to Remote Join

Comparing the Local Join plan P_l with the Remote Join plan P_r, for any query Q is it the case that

A

Always

$\mathsf{NDB}(P_l) \leq \mathsf{NDB}(P_r)$

B

Always

$\mathsf{NDB}(P_l) \geq \mathsf{NDB}(P_r)$

C

Sometimes

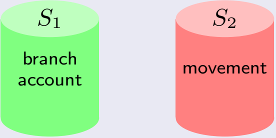
$\mathsf{NDB}(P_l) \leq \mathsf{NDB}(P_r)$

Sometimes

$\mathsf{NDB}(P_l) \geq \mathsf{NDB}(P_r)$

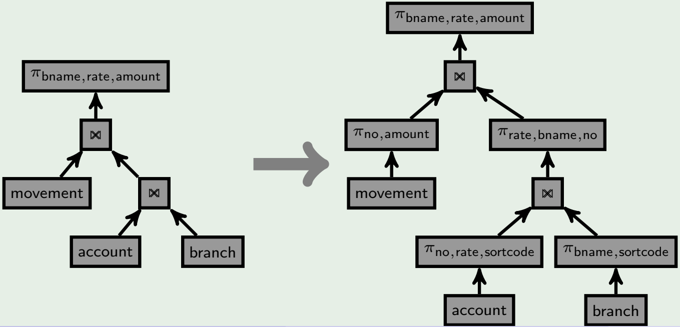
Example Comparison (5)

Query Pattern



S_2:
```
SELECT branch.bname,
       account.rate,
       movement.amount
FROM   account
       JOIN movement USING (no)
       JOIN branch USING (sortcode)
```

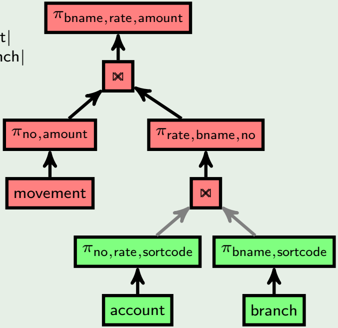
Query Plan



Example Comparison (5): Distributed Query Plans

Local Join

$$
\begin{aligned}
S_1\ to\ S_2 &= \mathsf{RowSize}(\pi_{\mathsf{no,rate,sortcode}}\mathsf{account}) \times |\pi_{\mathsf{no,rate,sortcode}}\mathsf{account}| \\
&= \mathsf{RowSize}(\pi_{\mathsf{bname,sortcode}}\mathsf{branch}) \times |\pi_{\mathsf{bname,sortcode}}\mathsf{branch}| \\
&= (4+4+1) \times 6 + (32+1) \times 3 \\
&= 54 + 99 \\
&= 153
\end{aligned}
$$



Example Comparison (5): Distributed Query Plans

Remote Join

$$
\begin{array}{rcl}
S_1\ to\ S_2 & = & \text{RowSize}(\pi_{\text{rate},\text{bname},\text{no}}\text{account} \bowtie \text{branch}) \times \\
 & = & \quad |\pi_{\text{rate},\text{bname},\text{no}}\text{account} \bowtie \text{branch}| \\
 & = & (4+32+4) \times \times 6 \\
 & = & 240
\end{array}
$$