CSE544
Data Management

Lectures 9

Query Optimization (Part 1)

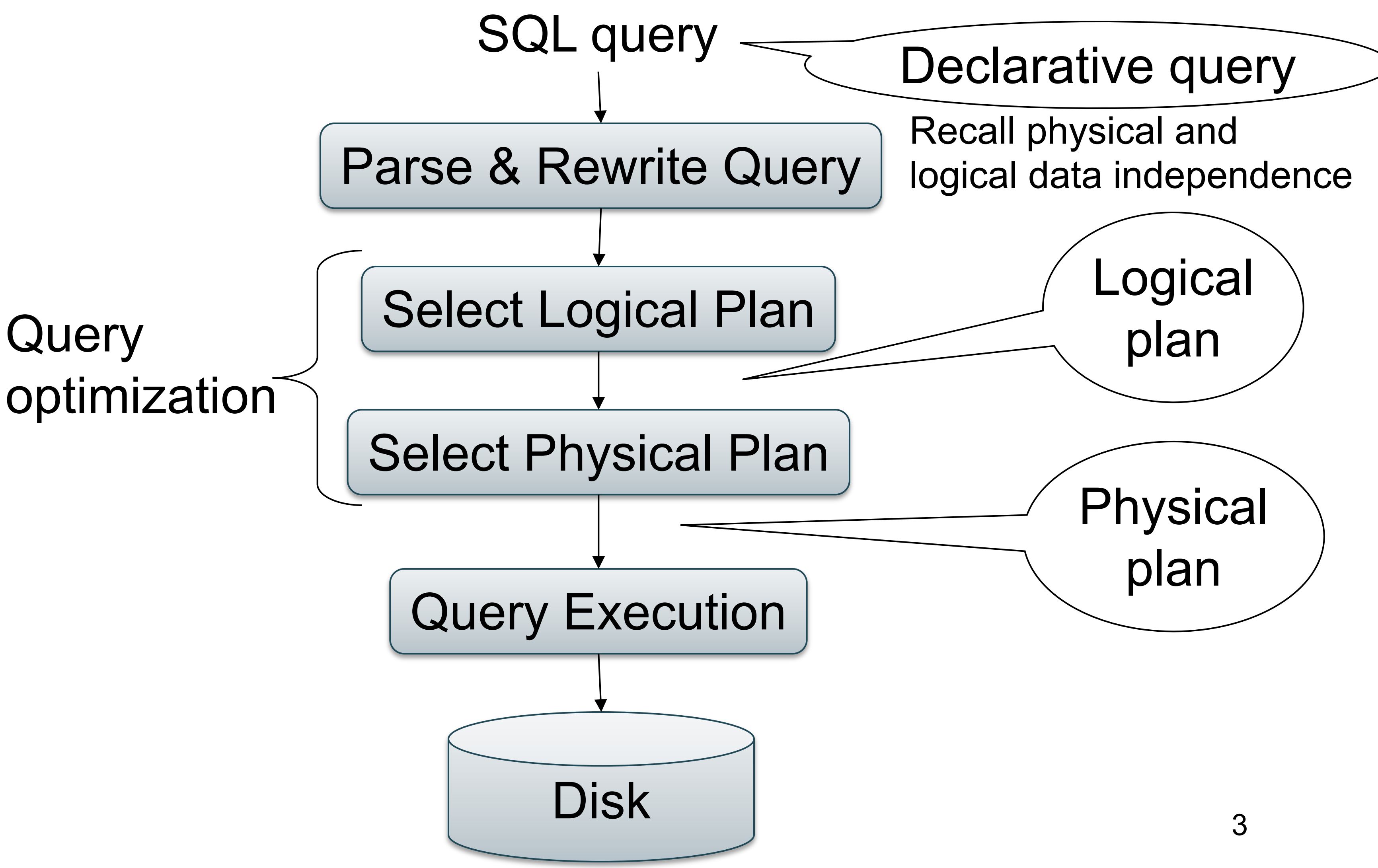
Announcements

- HW2 is due tomorrow!
- HW3 will be posted on Wednesday

- Review 5 (How good?) due Wednesday

- Mini-project guidelines posted

Query Optimization Motivation



Query Optimization

Goal:

- Given a query plan, find a cheaper (cheapest?) equivalent plan

- Why difficult:
  - Need to explore a large number of plans
  - Need to estimate the cost of each plan

Query Optimization

<u>Three major components</u>:

1. Cardinality and cost estimation

2. Search space

3. Plan enumeration algorithms

Query Optimization

Three major components:

1. Cardinality and cost estimation
2. Search space
3. Plan enumeration algorithms

Cardinality Estimation

Problem: given statistics on base tables and a query, estimate size of the answer

Very difficult, because:

- Need to do it very fast
- Need to use very little memory

Statistics on Base Data

- Number of tuples (cardinality) T(R)
- Number of physical pages B(R)
- Indexes, number of keys in the index V(R,a)
- Histogram on single attribute (1d)
- Histogram on two attributes (2d)

Computed periodically, often using sampling

Assumptions

- Uniformity
- Independence
- Containment of values
- Preservation of values

Size Estimation

Selection: size decreases by ***selectivity factor*** θ

$$T(\sigma_{pred}(R)) = \theta_{pred} * T(R)$$

Size Estimation

Selection: size decreases by ***selectivity factor*** θ

$$T(\sigma_{pred}(R)) = \theta_{pred} * T(R)$$

$$T(R \bowtie_{A=B} S) = \theta_{A=B} * T(R) * T(S)$$

$T(\sigma_{pred}(R)) = \theta_{pred} * T(R)$

Selectivity Factors

Uniformity assumption

Equality:

$\boxed{\sigma_{A=c}(R)}$

$T(\sigma_{pred}(R)) = \theta_{pred} * T(R)$

Selectivity Factors

Uniformity assumption

$\boxed{\sigma_{A=c}(R)}$

Equality:

- $\theta_{A=c} = 1/V(R,A)$

$T(\sigma_{pred}(R)) = \theta_{pred} * T(R)$

Selectivity Factors

Uniformity assumption

$\sigma_{A=c}(R)$

Equality:

- $\theta_{A=c} = 1/V(R,A)$

$\sigma_{c1<A<c2}(R)$

Range:

- $\theta_{c1<A<c2} = (c2 - c1)/(max(R,A) - min(R,A))$

$T(\sigma_{pred}(R)) = \theta_{pred} * T(R)$

Selectivity Factors

Uniformity assumption

$\boxed{\sigma_{A=c}(R)}$

Equality:

- $\theta_{A=c} = 1/V(R,A)$

$\boxed{\sigma_{c1<A<c2}(R)}$

Range:

- $\theta_{c1<A<c2} = (c2 - c1)/(max(R,A) - min(R,A))$

Conjunction

$\boxed{\sigma_{A=c \text{ and } B=d}(R)}$

$T(\sigma_{pred}(R)) = \theta_{pred} * T(R)$

Selectivity Factors

Uniformity assumption

Equality:

- $\theta_{A=c} = 1/V(R,A)$

$\boxed{\sigma_{A=c}(R)}$

$\boxed{\sigma_{c1<A<c2}(R)}$

Range:

- $\theta_{c1<A<c2} = (c2 - c1)/(max(R,A) - min(R,A))$

Conjunction

$\boxed{\sigma_{A=c \text{ and } B=d}(R)}$

Independence assumption

- $\theta_{pred1 \text{ and } pred2} = \theta_{pred1} * \theta_{pred2} = 1/V(R,A) * 1/V(R,B)$

$T(R \bowtie_{A=B} S) = \theta_{A=B} * T(R) * T(S)$

Selectivity Factors

$\boxed{R \bowtie_{R.A=S.B} S}$

Join

$T(R \bowtie_{A=B} S) = \theta_{A=B} * T(R) * T(S)$

Selectivity Factors

$\boxed{R \bowtie_{R.A=S.B} S}$

Join

- $\theta_{R.A=S.B} = 1/\ (\ MAX(\ V(R,A),\ V(S,B)))$

Why? Will explain next...

Selectivity Factors

$R \bowtie_{R.A=S.B} S$

Containment of values: if $V(R,A) \leq V(S,B)$, then the set of A values of R is included in the set of B values of S

- Note: this indeed holds when A is a foreign key in R, and B is a key in S

Selectivity Factors

$\boxed{R \bowtie_{R.A=S.B} S}$

Assume V(R,A) ≤ V(S,B)

- Tuple t in R joins with T(S)/V(S,B) tuples in S

Selectivity Factors

$R \bowtie_{R.A=S.B} S$

Assume $V(R,A) \leq V(S,B)$

- Tuple t in R joins with $T(S)/V(S,B)$ tuples in S
- Hence $T(R \bowtie_{A=B} S) = T(R)\, T(S) / V(S,B)$

$T(R \bowtie_{A=B} S) = \theta_{A=B} * T(R) * T(S)$

Selectivity Factors

$R \bowtie_{R.A=S.B} S$

Assume $V(R,A) \leq V(S,B)$

- Tuple t in R joins with $T(S)/V(S,B)$ tuples in S
- Hence $T(R \bowtie_{A=B} S) = T(R)\ T(S) \ / \ V(S,B)$

In general:

- $T(R \bowtie_{A=B} S) = T(R)\ T(S) \ / \ \max(V(R,A),V(S,B))$
- $\theta_{R.A=S.B} = 1/ (\ \max(\ V(R,A),\ V(S,B))$

Final Assumption

Preservation of values:

For any other attribute C:

- $V(R \bowtie_{A=B} S, C) = V(R, C)$ or
- $V(R \bowtie_{A=B} S, C) = V(S, C)$

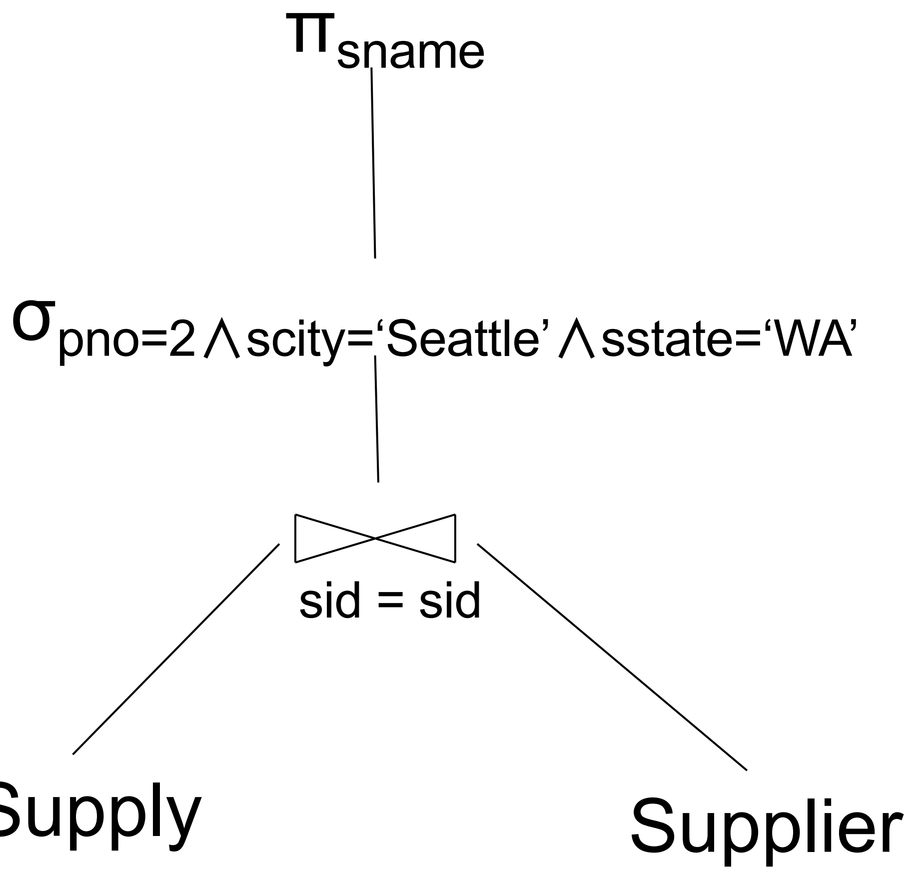
- This is needed higher up in the plan

Computing the Cost of a Plan

- Estimate <u>cardinalities</u> bottom-up
- Estimate <u>cost</u> by using estimated cardinalities
- Examples next...

Supplier(sid, sname, scity, sstate)
Supply(sid, pno, quantity)

Logical Query Plan 1




```sql
SELECT sname
FROM Supplier x, Supply y
WHERE x.sid = y.sid
    and y.pno = 2
    and x.scity = 'Seattle'
    and x.sstate = 'WA'
```

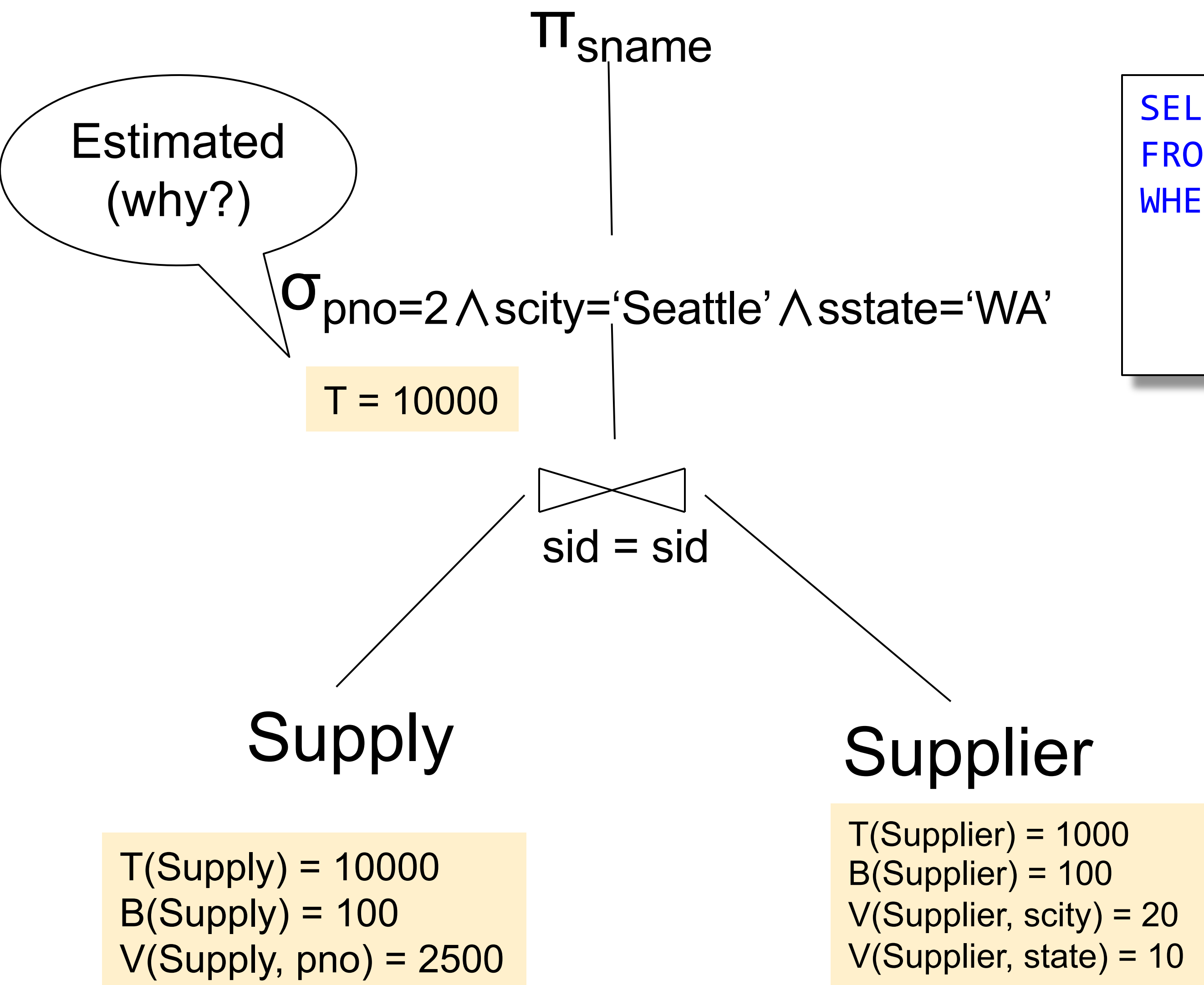
T(Supply) = 10000
B(Supply) = 100
V(Supply, pno) = 2500

T(Supplier) = 1000
B(Supplier) = 100
V(Supplier, scity) = 20
V(Supplier, state) = 10

M=11

Supplier(sid, sname, scity, sstate)
Supply(sid, pno, quantity)

Logical Query Plan 1




```
SELECT sname
FROM Supplier x, Supply y
WHERE x.sid = y.sid
    and y.pno = 2
    and x.scity = 'Seattle'
    and x.sstate = 'WA'
```

M=11

Supplier(sid, sname, scity, sstate)
Supply(sid, pno, quantity)

Logical Query Plan 1

π_{sname}

Estimated (why?)

$\sigma_{pno=2 \wedge scity='Seattle' \wedge sstate='WA'}$

T = 10000

Because key / foreign-key

$\bowtie_{sid = sid}$

Supply

T(Supply) = 10000
B(Supply) = 100
V(Supply, pno) = 2500

Supplier

T(Supplier) = 1000
B(Supplier) = 100
V(Supplier, scity) = 20
V(Supplier, state) = 10

```
SELECT sname
FROM Supplier x, Supply y
WHERE x.sid = y.sid
    and y.pno = 2
    and x.scity = ‘Seattle’
    and x.sstate = ‘WA’
```

M=11

Supplier(sid, sname, scity, sstate)
Supply(sid, pno, quantity)

Logical Query Plan 1

π_{sname}

Estimated (why?)

$\sigma_{pno=2 \wedge scity='Seattle' \wedge sstate='WA'}$

T = 10000

$\bowtie_{sid = sid}$

Because key / foreign-key

Also: θ = 1/max(V(Supply,sid)*V(Supplier,sid) = 1/V(Supplier,sid)

Supply

T(Supply) = 10000
B(Supply) = 100
V(Supply, pno) = 2500

Supplier

T(Supplier) = 1000
B(Supplier) = 100
V(Supplier, scity) = 20
V(Supplier, state) = 10

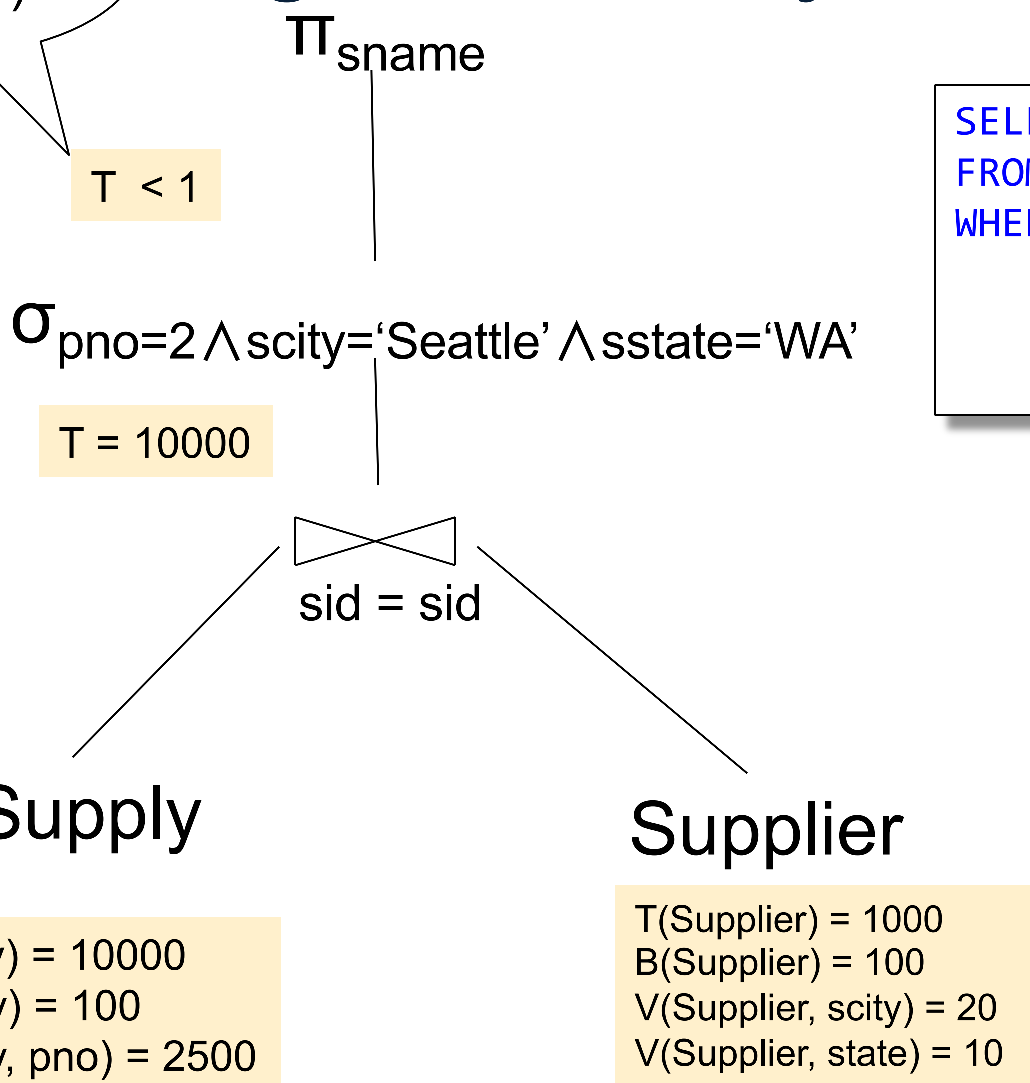
```
SELECT sname
FROM Supplier x, Supply y
WHERE x.sid = y.sid
    and y.pno = 2
    and x.scity = 'Seattle'
    and x.sstate = 'WA'
```

M=11

Supplier(sid, sname, scity, sstate)
Supply(sid, pno, quantity)

Logical Query Plan 1

Estimated
(why?)

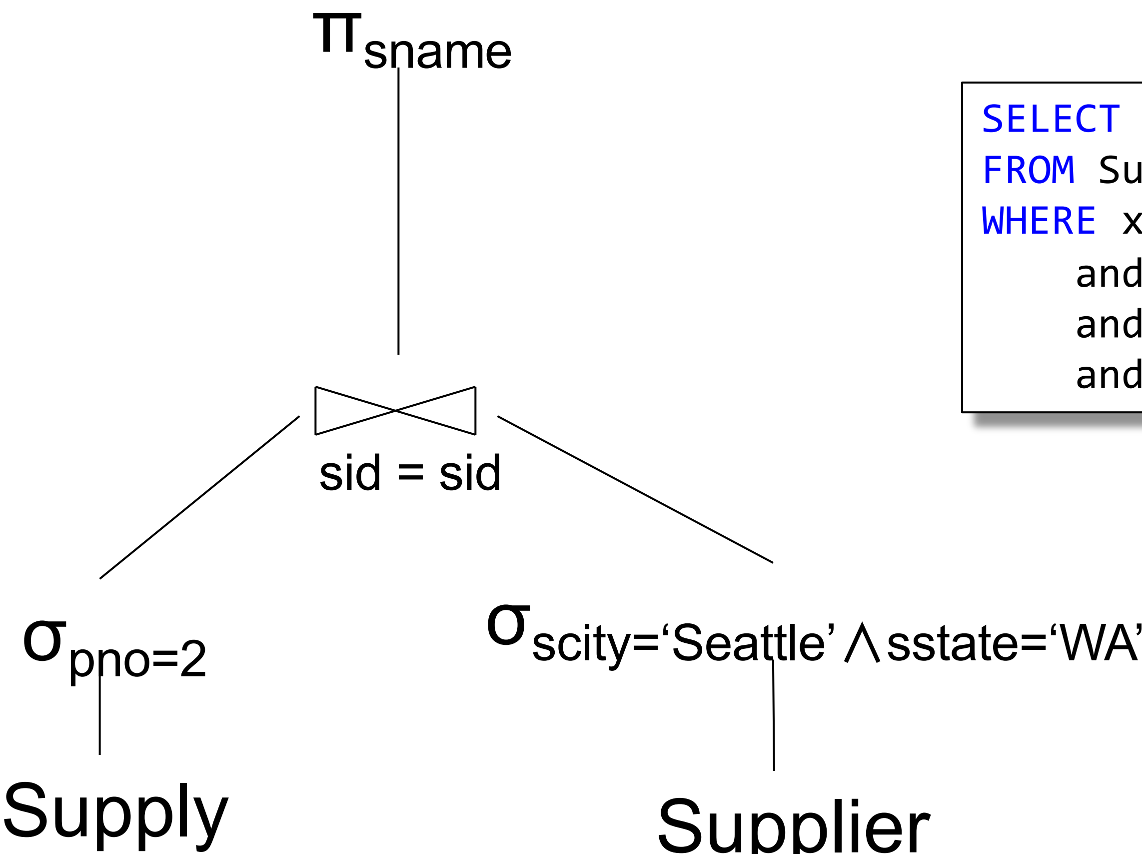



```
SELECT sname
FROM Supplier x, Supply y
WHERE x.sid = y.sid
    and y.pno = 2
    and x.scity = 'Seattle'
    and x.sstate = 'WA'
```

M=11

Supplier(sid, sname, scity, sstate)
Supply(sid, pno, quantity)

Logical Query Plan 2




```
SELECT sname
FROM Supplier x, Supply y
WHERE x.sid = y.sid
    and y.pno = 2
    and x.scity = 'Seattle'
    and x.sstate = 'WA'
```

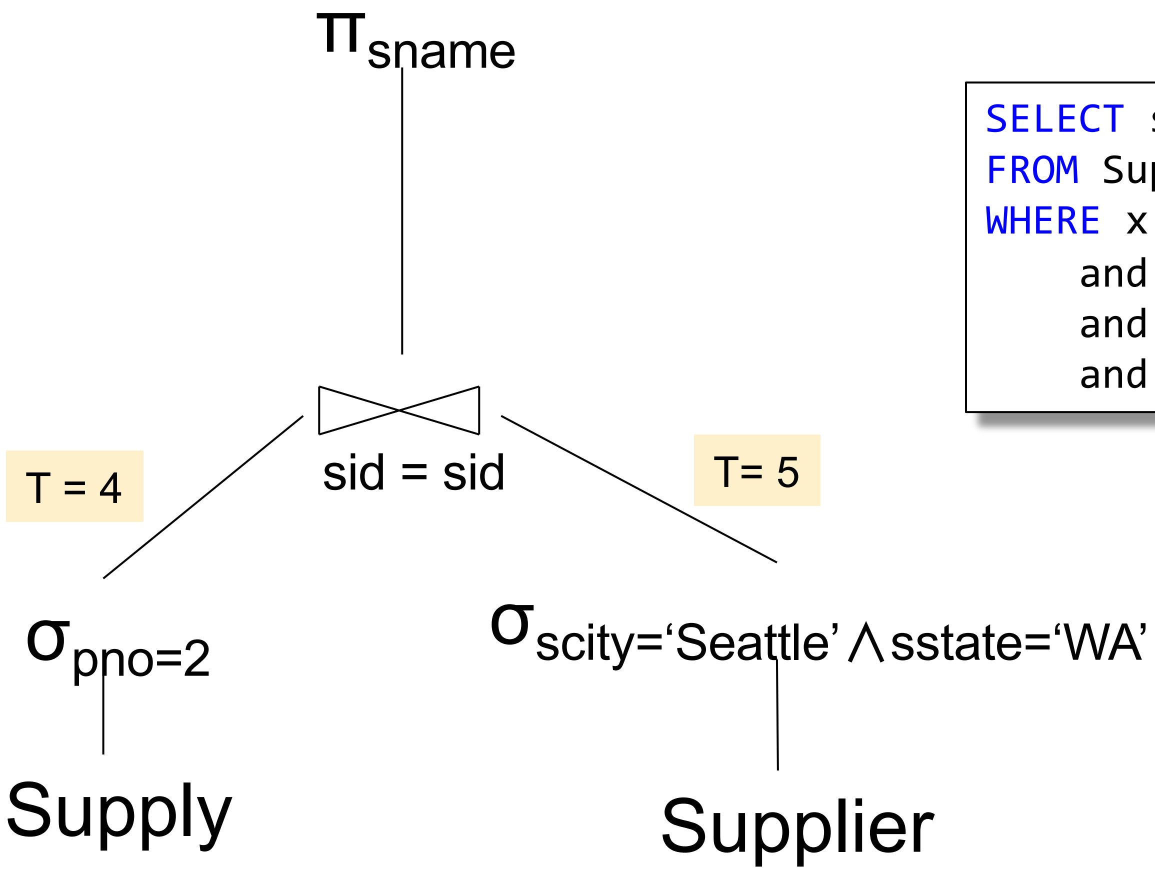
T(Supply) = 10000
B(Supply) = 100
V(Supply, pno) = 2500

T(Supplier) = 1000
B(Supplier) = 100
V(Supplier, scity) = 20
V(Supplier, state) = 10

M=11

Supplier(sid, sname, scity, sstate)
Supply(sid, pno, quantity)

Logical Query Plan 2




```
SELECT sname
FROM Supplier x, Supply y
WHERE x.sid = y.sid
    and y.pno = 2
    and x.scity = ‘Seattle’
    and x.sstate = ‘WA’
```

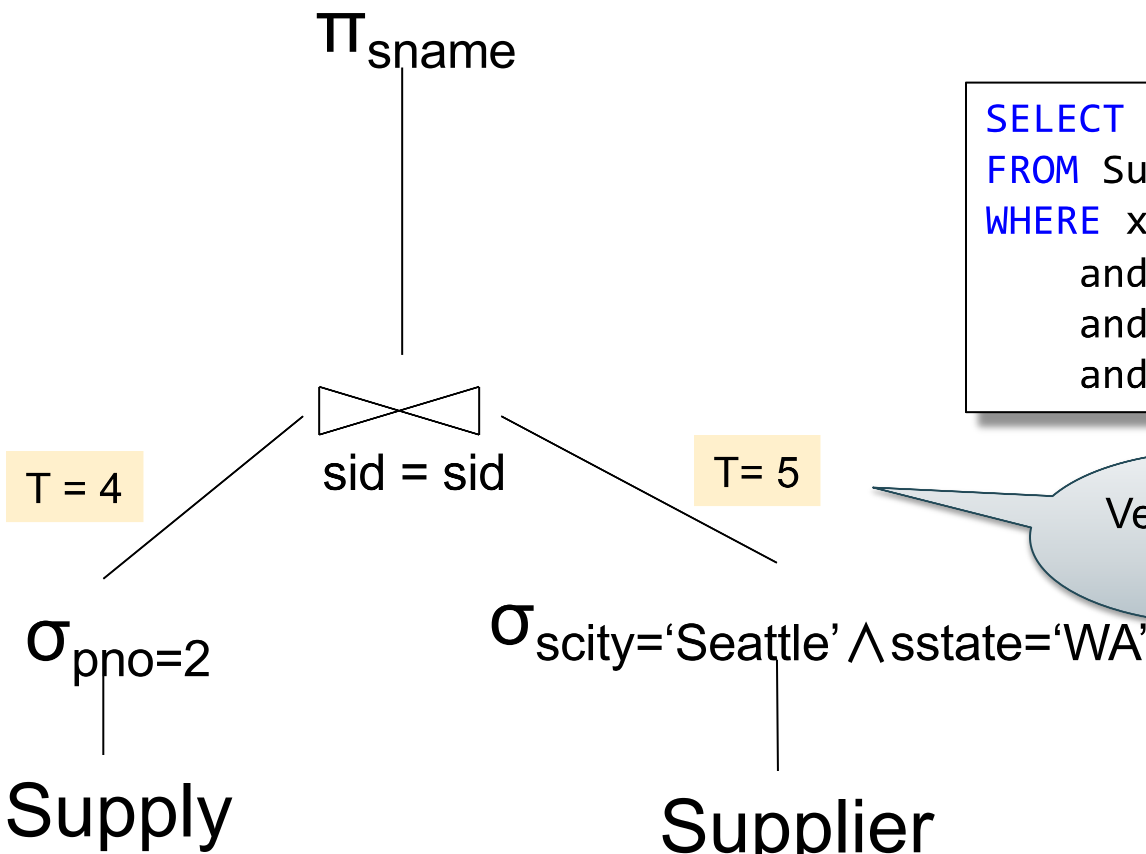
T(Supply) = 10000
B(Supply) = 100
V(Supply, pno) = 2500

T(Supplier) = 1000
B(Supplier) = 100
V(Supplier, scity) = 20
V(Supplier, state) = 10

M=11

Supplier(sid, sname, scity, sstate)
Supply(sid, pno, quantity)

Logical Query Plan 2




```
SELECT sname
FROM Supplier x, Supply y
WHERE x.sid = y.sid
    and y.pno = 2
    and x.scity = ‘Seattle’
    and x.sstate = ‘WA’
```

Very wrong!
Why?

T(Supply) = 10000
B(Supply) = 100
V(Supply, pno) = 2500

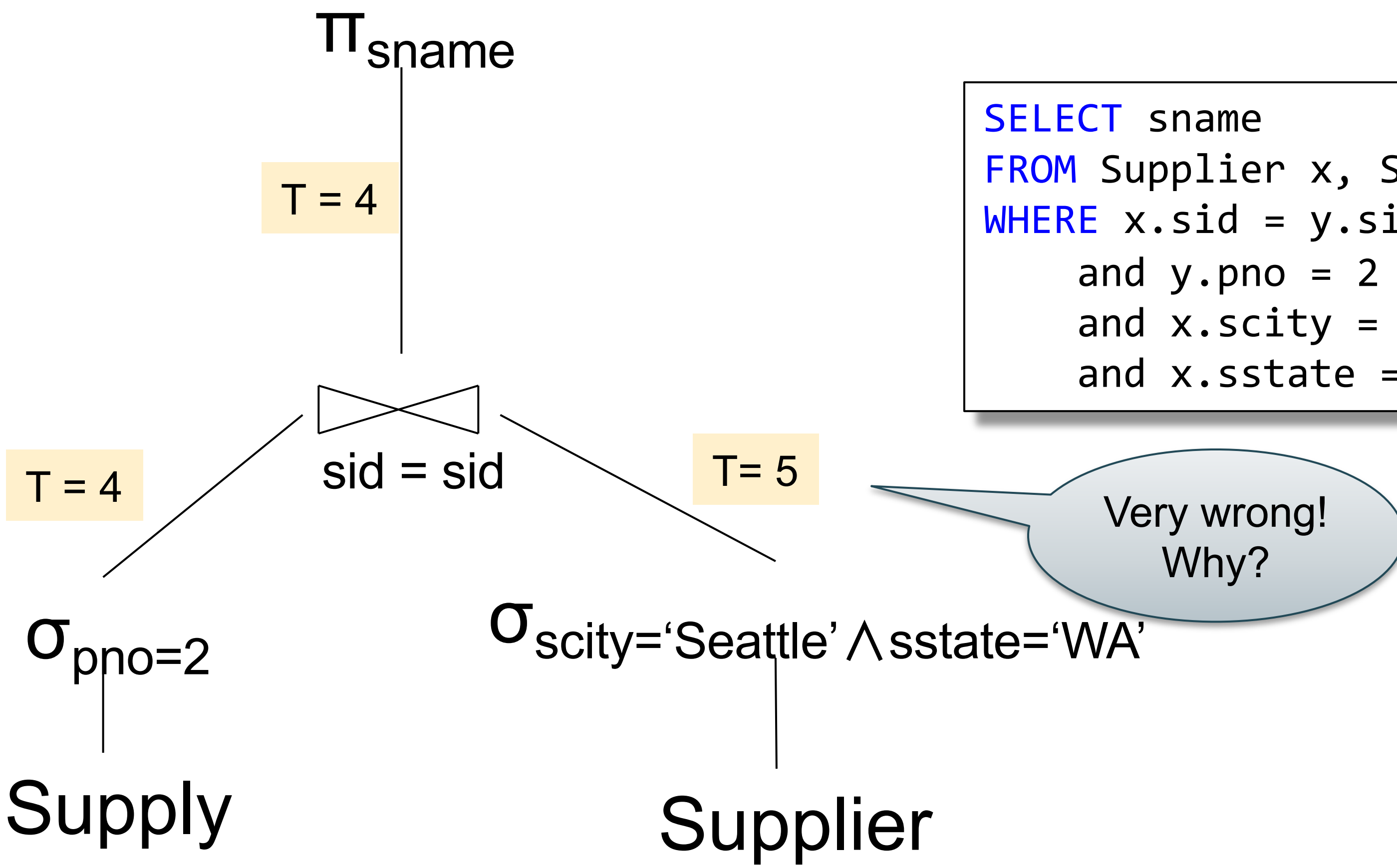
T(Supplier) = 1000
B(Supplier) = 100
V(Supplier, scity) = 20
V(Supplier, state) = 10

M=11

Supplier(sid, sname, scity, sstate)
Supply(sid, pno, quantity)

Logical Query Plan 2

```
SELECT sname
FROM Supplier x, Supply y
WHERE x.sid = y.sid
    and y.pno = 2
    and x.scity = ‘Seattle’
    and x.sstate = ‘WA’
```



T(Supply) = 10000
B(Supply) = 100
V(Supply, pno) = 2500

T(Supplier) = 1000
B(Supplier) = 100
V(Supplier, scity) = 20
V(Supplier, state) = 10

M=11

Supplier(sid, sname, scity, sstate)
Supply(sid, pno, quantity)

Logical Query Plan 2

```
SELECT sname
FROM Supplier x, Supply y
WHERE x.sid = y.sid
    and y.pno = 2
    and x.scity = ‘Seattle’
    and x.sstate = ‘WA’
```

π_{sname}

T = 4

Different estimate ☹

$\bowtie_{sid = sid}$

T = 4

T= 5

Very wrong! Why?

$\sigma_{pno=2}$

$\sigma_{scity=\text{‘Seattle’} \wedge sstate=\text{‘WA’}}$

Supply

Supplier

T(Supply) = 10000
B(Supply) = 100
V(Supply, pno) = 2500

T(Supplier) = 1000
B(Supplier) = 100
V(Supplier, scity) = 20
V(Supplier, state) = 10

M=11

Supplier(sid, sname, scity, sstate)
Supply(sid, pno, quantity)

Physical Plan 1

π_{sname}

T < 1

$\sigma_{pno=2 \wedge scity='Seattle' \wedge sstate='WA'}$

T = 10000

$\bowtie_{sid = sid}$

Block nested loop join

Scan Supply

Scan Supplier

T(Supply) = 10000
B(Supply) = 100
V(Supply, pno) = 2500

T(Supplier) = 1000
B(Supplier) = 100
V(Supplier, scity) = 20
V(Supplier, state) = 10

Total cost:

M=11

Supplier(sid, sname, scity, sstate)
Supply(sid, pno, quantity)

Physical Plan 1

π_{sname}

T < 1

$\sigma_{pno=2 \wedge scity='Seattle' \wedge sstate='WA'}$

T = 10000

Total cost: 100+100*100/10 = 1100

$\bowtie_{sid = sid}$

Block nested loop join

Scan Supply

Scan Supplier

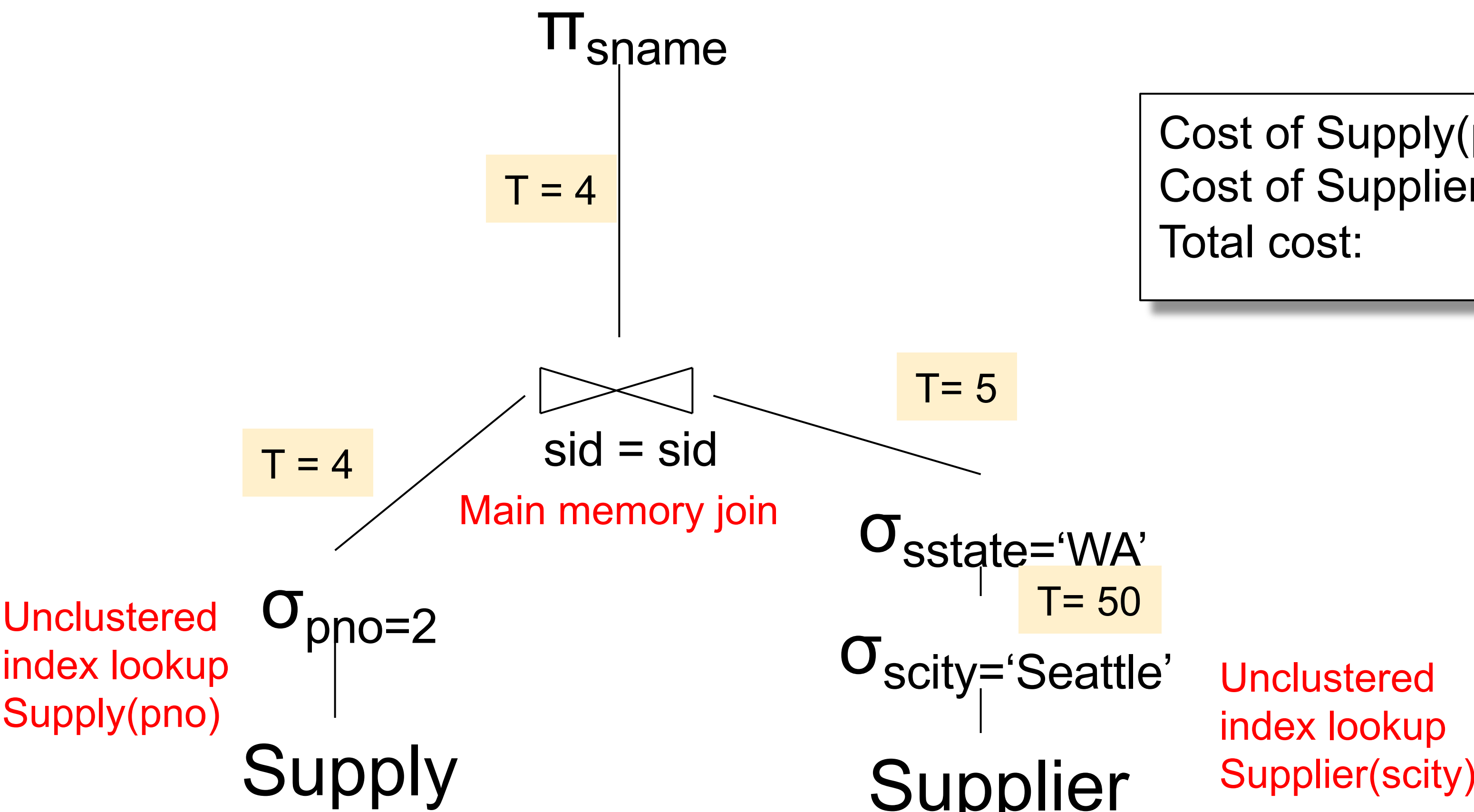
T(Supply) = 10000
B(Supply) = 100
V(Supply, pno) = 2500

T(Supplier) = 1000
B(Supplier) = 100
V(Supplier, scity) = 20
V(Supplier, state) = 10

M=11

Supplier(<u>sid</u>, sname, scity, sstate)
Supply(<u>sid, pno</u>, quantity)

Physical Plan 2



Cost of Supply(pno) =
Cost of Supplier(scity) =
Total cost:

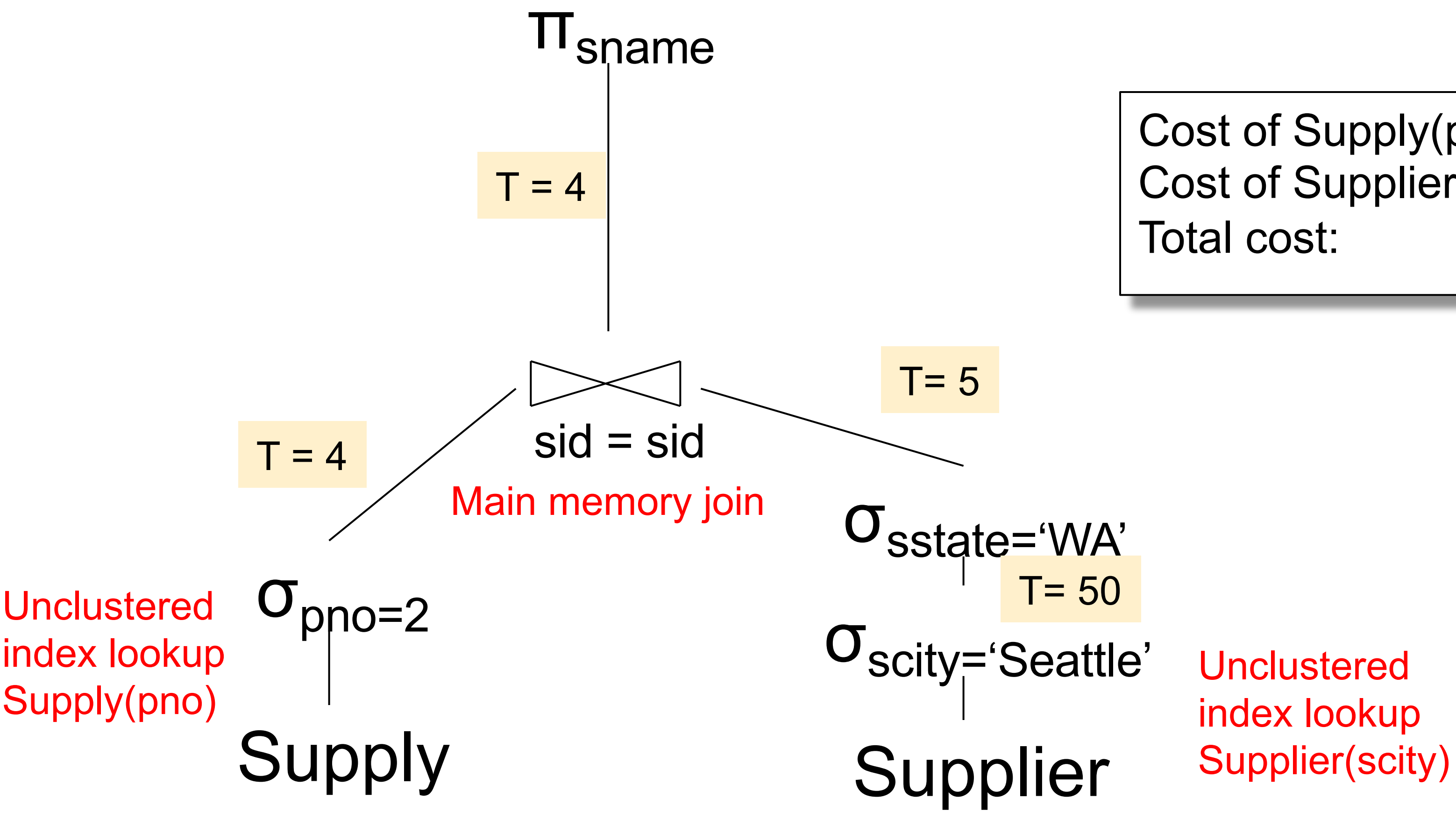
T(Supply) = 10000
B(Supply) = 100
V(Supply, pno) = 2500

T(Supplier) = 1000
B(Supplier) = 100
V(Supplier, scity) = 20
V(Supplier, state) = 10

M=11

Supplier(sid, sname, scity, sstate)
Supply(sid, pno, quantity)

Physical Plan 2



Cost of Supply(pno) = 4
Cost of Supplier(scity) =
Total cost:

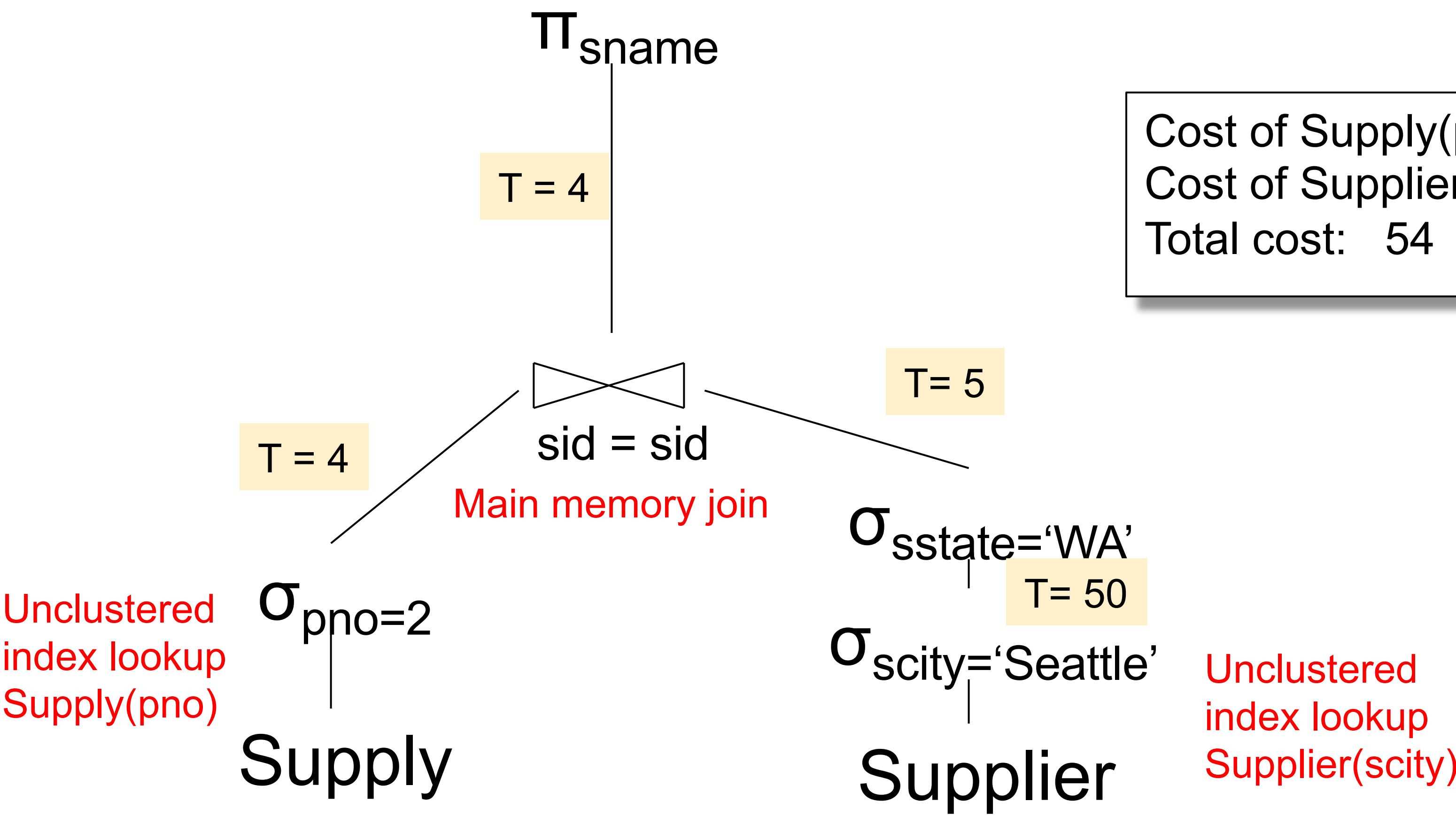
T(Supply) = 10000
B(Supply) = 100
V(Supply, pno) = 2500

T(Supplier) = 1000
B(Supplier) = 100
V(Supplier, scity) = 20
V(Supplier, state) = 10

M=11

Supplier(sid, sname, scity, sstate)
Supply(sid, pno, quantity)

Physical Plan 2



Cost of Supply(pno) = 4
Cost of Supplier(scity) = 50
Total cost: 54

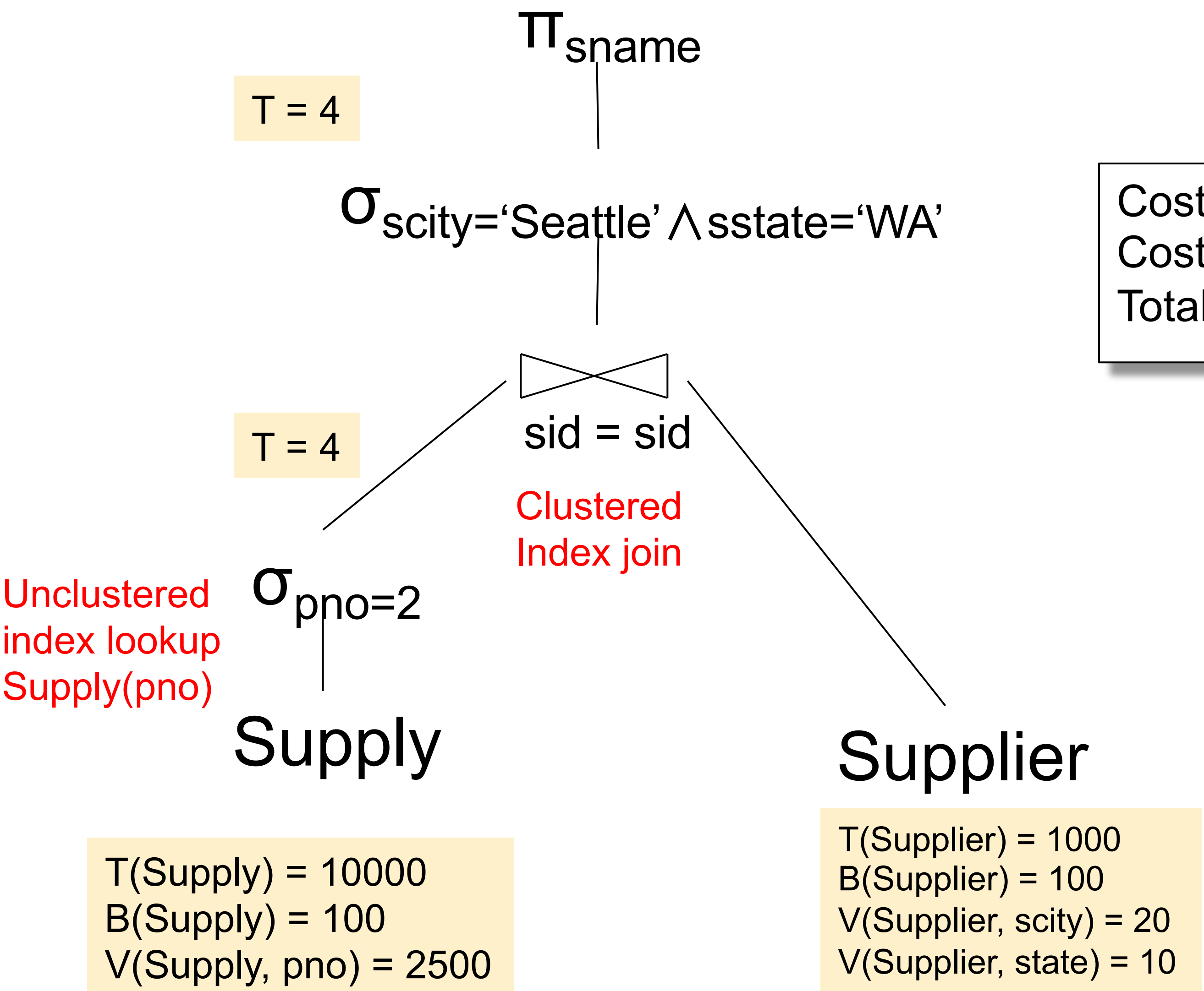
T(Supply) = 10000
B(Supply) = 100
V(Supply, pno) = 2500

T(Supplier) = 1000
B(Supplier) = 100
V(Supplier, scity) = 20
V(Supplier, state) = 10

M=11

Supplier(sid, sname, scity, sstate)
Supply(sid, pno, quantity)

Physical Plan 3

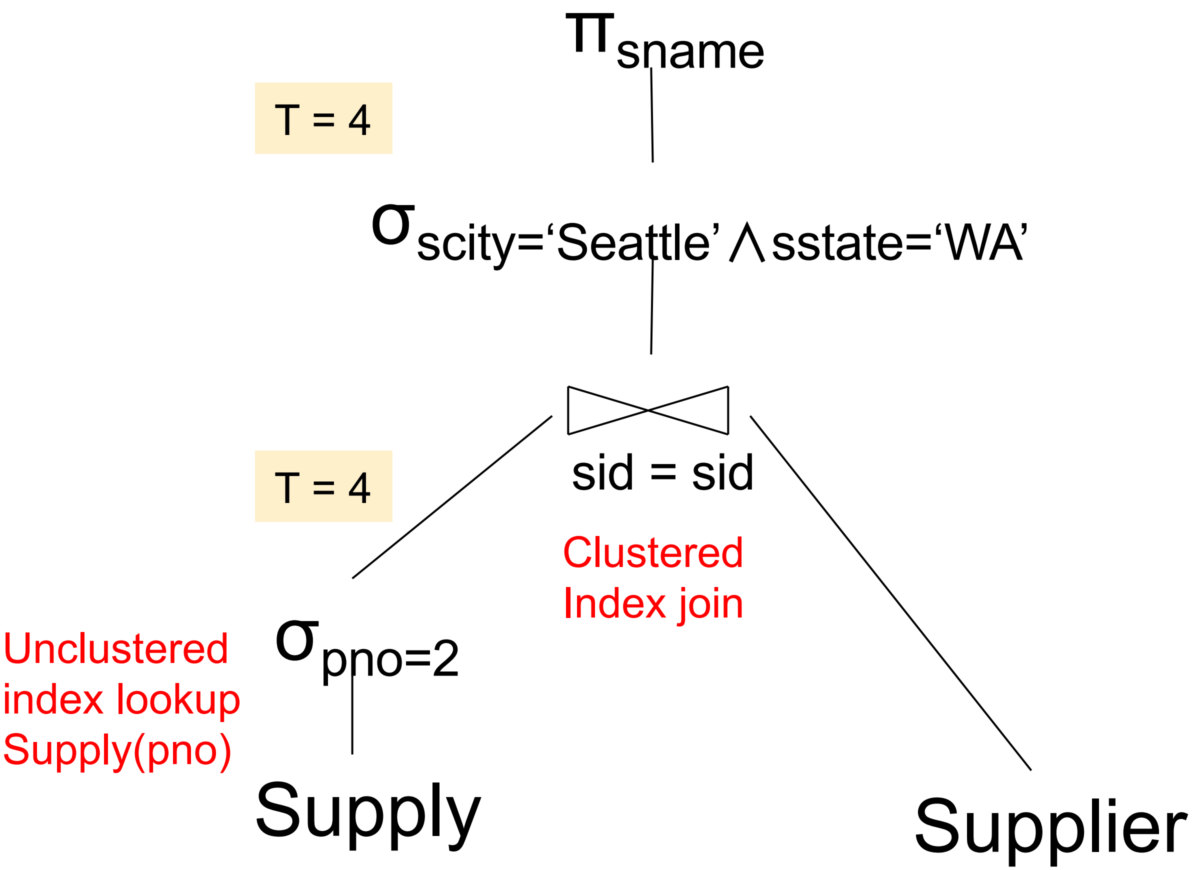


Cost of Supply(pno) =
Cost of Index join =
Total cost:

M=11

Supplier(sid, sname, scity, sstate)
Supply(sid, pno, quantity)

Physical Plan 3



Cost of Supply(pno) = 4
Cost of Index join =
Total cost:

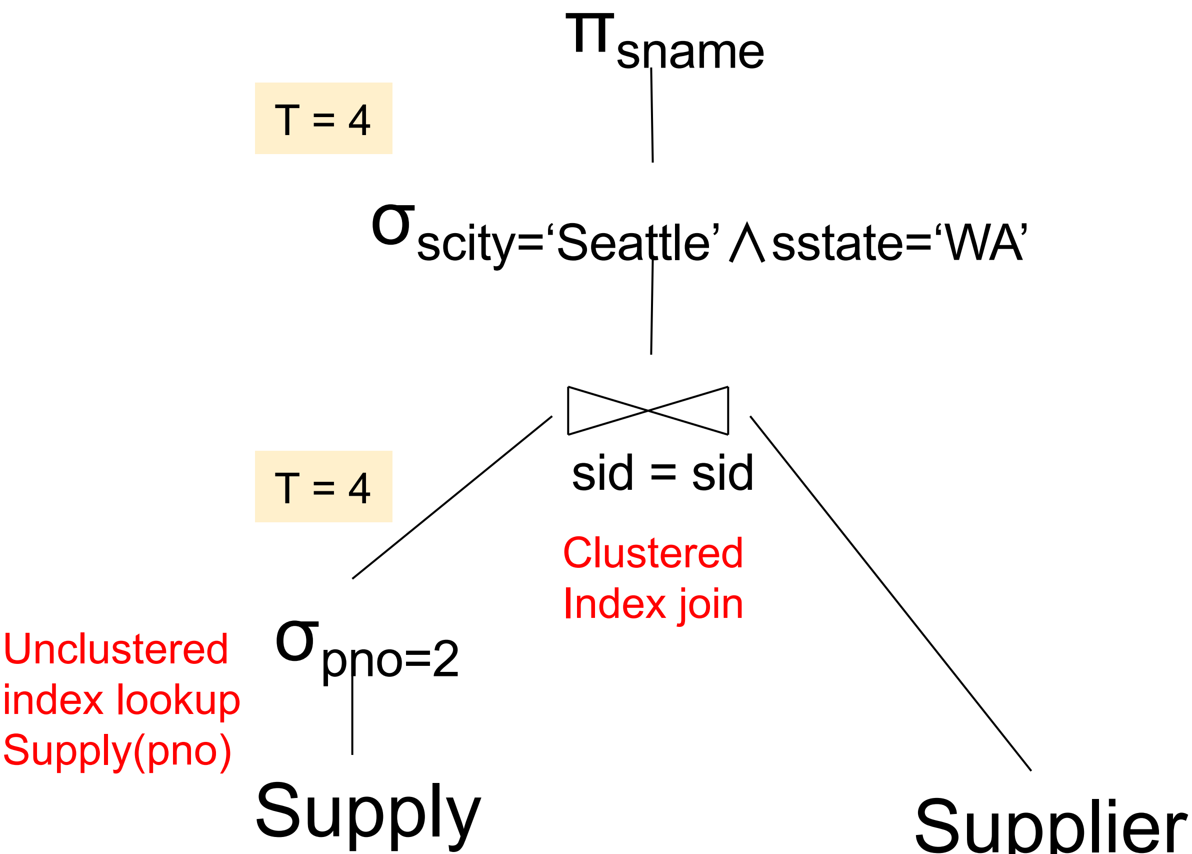
T(Supply) = 10000
B(Supply) = 100
V(Supply, pno) = 2500

T(Supplier) = 1000
B(Supplier) = 100
V(Supplier, scity) = 20
V(Supplier, state) = 10

M=11

Supplier(<u>sid</u>, sname, scity, sstate)
Supply(<u>sid, pno</u>, quantity)

Physical Plan 3



Cost of Supply(pno) = 4
Cost of Index join = 4
Total cost: 8

T(Supply) = 10000
B(Supply) = 100
V(Supply, pno) = 2500

T(Supplier) = 1000
B(Supplier) = 100
V(Supplier, scity) = 20
V(Supplier, state) = 10

M=11

Discussion

- We considered only IO cost;
  real systems need to consider IO+CPU
- Each system has its own hacks
- We assumed that all index pages were in memory: sometimes we need to add the cost of fetching index pages from disk

Histograms

- T(R), V(R,A) too coarse
- Histogram: separate stats per bucket

- In each bucket store:
  - T(bucket)
  - V(bucket,A) – optional

Histograms

Employee(ssn, name, age)

T(Employee) = 25000, V(Empolyee, age) = 50

$\sigma_{age=48}$(Empolyee) = ?

Histograms

Employee(<u>ssn</u>, name, age)

T(Employee) = 25000, V(Empolyee, age) = 50

$\sigma_{age=48}$(Empolyee) = ?

Estimate: T(Employee) / V(Employee,age) = 500

Histograms

Employee(ssn, name, age)

T(Employee) = 25000, V(Empolyee, age) = 50

$\sigma_{age=48}$(Empolyee) = ?

Estimate: T(Employee) / V(Employee,age) = 500

Age:	0..20	20..29	30-39	40-49	50-59	> 60
T =	200	800	5000	12000	6500	500

Histograms

Employee(<u>ssn</u>, name, age)

T(Employee) = 25000, V(Empolyee, age) = 50

$\sigma_{age=48}$(Empolyee) = ?

Estimate: T(Employee) / V(Employee,age) = 500

Age:	0..20	20..29	30-39	40-49	50-59	> 60
T =	200	800	5000	12000	6500	500

Assume V = 10

Histograms

Employee(ssn, name, age)

T(Employee) = 25000, V(Empolyee, age) = 50

$\sigma_{age=48}$(Empolyee) = ?

~~Estimate: T(Employee) / V(Employee,age) = 500~~

Age:	0..20	20..29	30-39	40-49	50-59	> 60
T =	200	800	5000	12000	6500	500

Estimate: 12000/10 = 1200

Assume V = 10

Histograms

Employee(ssn, name, age)

T(Employee) = 25000, V(Empolyee, age) = 50

$\sigma_{age=48}$(Empolyee) = ?

~~Estimate: T(Employee) / V(Employee,age) = 500~~

Age:	0..20	20..29	30-39	40-49	50-59	> 60
T =	200	800	5000	12000	6500	500
V =	3	10	7	6	5	4

Estimate: 12000/10 = 1200

Histograms

Employee(ssn, name, age)

T(Employee) = 25000, V(Empolyee, age) = 50

$\sigma_{age=48}$(Empolyee) = ?

~~Estimate: T(Employee) / V(Employee,age) = 500~~

Age:	0..20	20..29	30-39	40-49	50-59	> 60
T =	200	800	5000	12000	6500	500
V =	3	10	7	6	5	4

Estimate: ~~12000/10 = 1200~~ 12000/6 = 2000

Types of Histograms

- Eq-Width
- Eq-Depth
- Compressed: store outliers separately
- V-Optimal histograms

Employee(ssn, name, age)

Histograms

Eq-width:

Age:	0..20	20..29	30-39	40-49	50-59	> 60
Tuples	200	800	5000	12000	6500	500

Eq-depth:

Age:	0..32	33..41	42-46	47-52	53-58	> 60
Tuples	1800	2000	2100	2200	1900	1800

Compressed: store separately highly frequent values: (48,1900)

V-Optimal Histograms

- Error:

$$\sum_{v \in Domain(A)} \left(|\sigma_{A=v}(R)| - est_{Hist}\left(\sigma_{A=v}(R)\right) \right)^2$$

- Bucket boundaries = $\text{argmin}_{\text{Hist}}(\text{Error})$
- Dynamic programming
- Modern databases systems use V-optimal histograms or some variations

Discussion

- Cardinality estimation = still unsolved
- Histograms:
  - Small number of buckets (why?)
  - Updated only periodically (why?)
  - No 2d histograms (except db2) why?
- Samples:
  - Fail for low selectivity estimates, joins
- Cross-join correlation – still unsolved