CS3223 Tutorial 7

Gary Lim

Admin

- Project deadline: Fri, 18 Mar 2022
- Project FAQs: https://docs.google.com/document/d/e/2PACX-1vS9sjvuTqYbYZRmqPzsl-ZoqbQ00mX8CphIn60WYYIzTRP7x8VIrLdmOnSV5fIdGtYxIk9sz3J5QPwi/pub

Chapter Review

- Summary of Search Algorithms
- Cost Estimation

Search Algorithms

Search Algorithm	Search Space	Plan Quality	Optimization Overhead
Exhaustive	Complete	✓✓	✗✗
Greedy	Small	✓/✗	✓ Polynomial
Randomized	Depends	✓/✗	Depends on how long it is ran
Systems R	Almost complete	✓	✗ $O(2^k)$

Statistics

- |R| – Number of pages in R
- ||R|| – Number of tuples in R
- S(R) – Size of **each tuple** in R
- V(R, A) – Number of distinct values of attribute A in R

Estimation Assumption

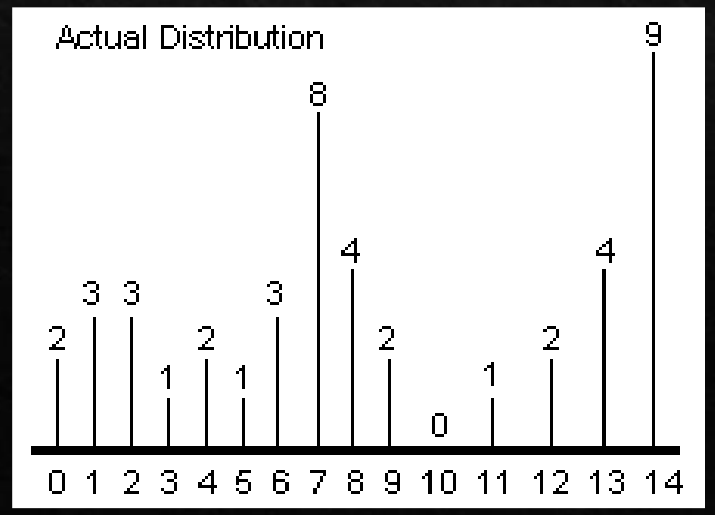
1. **Uniformity Assumption**
   Uniform distribution of attribute values
2. **Independence assumption**
   Independent distribution of values in different attributes
3. **Inclusion assumption**
   For $R \bowtie_{R.A=S.B} S$, if $V(R, A) \leq V(S, B)$, then $\pi_A(R) \subseteq \pi_A(S)$

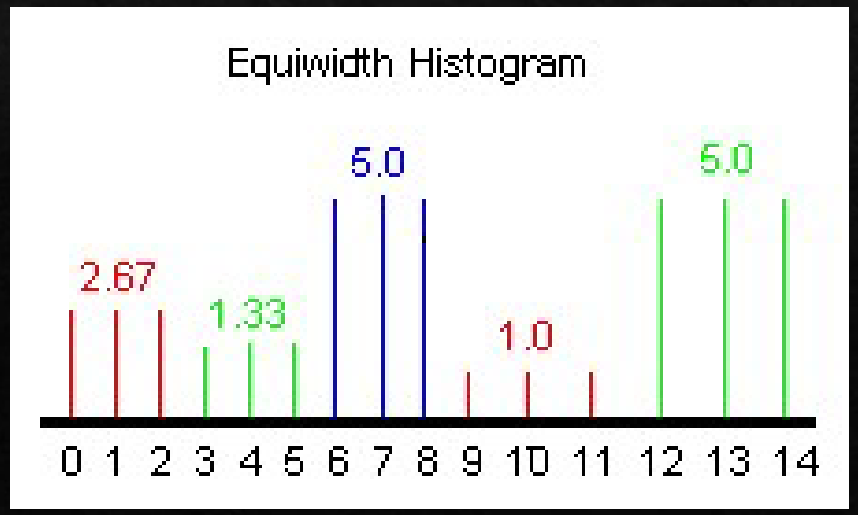
Preservation of value sets

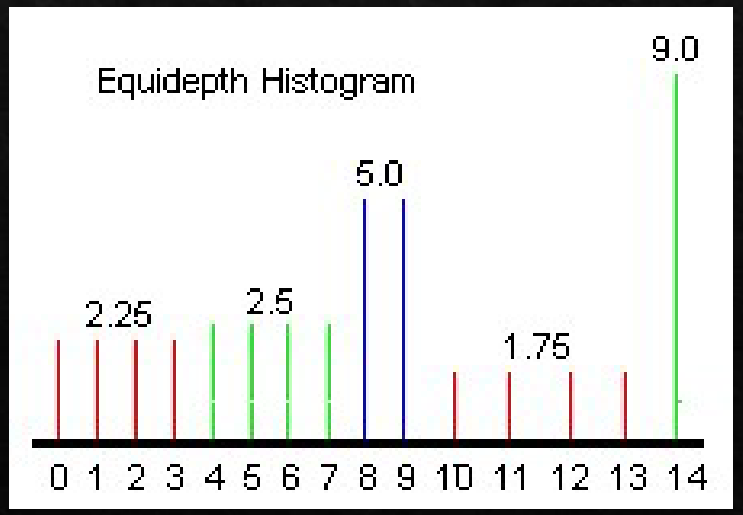
http://mlwiki.org/index.php/Query_Result_Size_Estimation

Estimation with Histograms

- Normally we assume Uniform Distribution and assume each value has equal number of tuples
- Histograms give us a more accurate estimation of how many tuples are there for each value







Tutorial Questions

Q1

T(W) – total number of tuples in W

W(a,b)	X(b, c)	Y(c, d)
T(W) = 100	T(X) = 200	T(Y) = 300
V(W, a) = 20	V(X, b) = 50	V(Y, c) = 50
V(W, b) = 60	V(X, c) = 100	V(Y, d) = 50

Estimate the sizes of the following:

$\sigma_{a=10}(W)$

$\sigma_{a=10 \wedge b>2}(W)$

$\sigma_{a>10}(W)$

$\sigma_{a=10 \wedge b=2}(W)$

Q1

W(a,b)	X(b, c)	Y(c, d)
T(W) = 100	T(X) = 200	T(Y) = 300
V(W, a) = 20	V(X, b) = 50	V(Y, c) = 50
V(W, b) = 60	V(X, c) = 100	V(Y, d) = 50

Estimate the sizes of the following:

$\sigma_{a=10}(W)$
1/20 * 100 = 5 tuples

$\sigma_{a>10}(W)$
10/20 * 100 = 50 tuples

$\sigma_{a=10 \wedge b=2}(W)$
1/20 * 1/60 * 100 = 0.0833 ≈ 1 tuple

$\sigma_{a=10 \wedge b>2}(W)$
1/20 * 58/60 * 100 = 4.833 ≈ 5 tuples

Assume:
domain(a) = [1, 20]

Assume:
domain(b) = [1, 60]

Q1

W(a,b)	X(b, c)	Y(c, d)
T(W) = 100	T(X) = 200	T(Y) = 300
V(W, a) = 20	V(X, b) = 50	**V(Y, c) = 50**
V(W, b) = 60	**V(X, c) = 100**	V(Y, d) = 50

Estimate the sizes of the following:

$\sigma_{a=10}(W)$
1/20 * 100 = 5 tuples

$\sigma_{a>10}(W)$
10/20 * 100 = 50 tuples

$\sigma_{a=10 \wedge b=2}(W)$
1/20 * 1/60 * 100 = 0.0833 ≈ 1 tuple

$\sigma_{a=10 \wedge b>2}(W)$
1/20 * 58/60 * 100 = 4.833 ~ 5 tuples

$\sigma_{c=20}(X) \bowtie Y$
1/100 * 200 * 6 = 12 tuples

Uniform distribution:
There are 6 tuples for each value of Y.c

Q1

W(a,b)	X(b, c)	Y(c, d)
T(W) = 100	T(X) = 200	T(Y) = 300
V(W, a) = 20	V(X, b) = 50	**V(Y, c) = 50**
V(W, b) = 60	**V(X, c) = 100**	V(Y, d) = 50

Estimate the sizes of the following:

$\sigma_{a=10}(W)$
1/20 * 100 = 5 tuples

$\sigma_{a>10}(W)$
10/20 * 100 = 50 tuples

$\sigma_{a=10 \wedge b=2}(W)$
1/20 * 1/60 * 100 = 0.0833 ≈ 1 tuple

$\sigma_{a=10 \wedge b>2}(W)$
1/20 * 58/60 * 100 = 4.833 ~ 5 tuples

$\sigma_{c=20}(X) \bowtie Y$
1/100 * 200 * 6 = 12 tuples
1/100 * 200 * 0 = 0 tuples

Inclusion assumption:
All values of Y.c appear in X.c BUT NOT ALL values of X.c appear in Y.c. (It is possible that no tuples exist for Y.c = 20)

Q1

W(a,b)	X(b, c)	Y(c, d)
T(W) = 100	T(X) = 200	T(Y) = 300
V(W, a) = 20	**V(X, b) = 50**	**V(Y, c) = 50**
V(W, b) = 60	**V(X, c) = 100**	V(Y, d) = 50

Estimate the sizes of the following:
W ⋈ X ⋈ Y
(200 * 100 / 60) / 100 * 300 = 1000 tuples

- 60 distinct values for W.b
- 50 distinct values for X.b
- By **inclusion assumption**, domain(X.b) ⊆ domain(W.b)
- Each tuple of X will join with k tuples of W
- By **uniform dist assumption**, k = T(W) / V(W, b)

Q1

W(a,b)	X(b, c)	Y(c, d)
T(W) = 100	T(X) = 200	T(Y) = 300
V(W, a) = 20	**V(X, b) = 50**	**V(Y, c) = 50**
V(W, b) = 60	**V(X, c) = 100**	V(Y, d) = 50

Estimate the sizes of the following:
W ⋈ X ⋈ Y
(200 * 100 / 60) / 100 * 300 = 1000 tuples

- 60 distinct values for W.b
- 50 distinct values for X.b
- By **inclusion assumption**, domain(X.b) ⊆ domain(W.b)
- Each tuple of X will join with k tuples of W
- By **uniform dist assumption**, k = T(W) / V(W, b)

- 100 distinct values for X.c
- 50 distinct values for Y.c
- By **inclusion assumption**, domain(Y.c) ⊆ domain(X.c)
- Each tuple of X will join with k tuples of W
- By **uniform dist assumption**, k = T(W) / V(W, c)

Q1

W(a,b)	X(b, c)	Y(c, d)
T(W) = 100	T(X) = 200	T(Y) = 300
V(W, a) = 20	**V(X, b) = 50**	**V(Y, c) = 50**
V(W, b) = 60	**V(X, c) = 100**	V(Y, d) = 50

Estimate the sizes of the following:
W ⋈ X ⋈ Y
(200 * 100 / 60) / 100 * 300 = 1000 tuples

$$T(\mathrm{W} \bowtie \mathrm{X} \bowtie \mathrm{Y}) = \frac{\|W\| \times \|X\| \times \|Y\|}{\max\big(V(W,b), V(X,b)\big) \times \max\big(V(X,c), V(Y,c)\big)}$$

Q1

W(a,b)	X(b, c)	Y(c, d)
T(W) = 100	T(X) = 200	T(Y) = 300
V(W, a) = 20	V(X, b) = 50	**V(Y, c) = 50**
V(W, b) = 60	**V(X, c) = 100**	V(Y, d) = 50

Estimate the sizes of the following:

$\sigma_{a=10}(W)$
1/20 * 100 = 5 tuples

$\sigma_{a>10}(W)$
10/20 * 100 = 50 tuples

$\sigma_{a=10 \wedge b=2}(W)$
1/20 * 1/60 * 100 = 0.0833 ≈ 1 tuple

$\sigma_{a=10 \wedge b>2}(W)$
1/20 * 58/60 * 100 = 4.833 ~ 5 tuples

$\sigma_{c=20}(X) \bowtie Y$
1/100 * 200 * 6 = 12 tuples
1/100 * 200 * 0 = 0 tuples

$W \bowtie X \bowtie Y$
(100 * 200 / 60) * 300 / 100 = 1000 tuples

Q2

Given **V(R,b) = V(S, b) = V(T, b) = 100**

	0	1	2	3	4	Others
R.b	72	30	10			388
S.b		40	16	250		194
T.b	100		7		20	873

SELECT a, b FROM R, S, T WHERE R.b = S.b AND S.b = T.b

Estimate the size of this query in terms of pages and tuples

Q2

Given **V(R,b) = V(S, b) = V(T, b) = 100**

	0	1	2	3	4	5 - 99
R.b	72	30	10	4	4	4 each
S.b	2	40	16	250	2	2 each
T.b	100	9	7	9	20	9 each

Ignore this

SELECT a, b FROM R, S, T WHERE R.b = S.b AND S.b = T.b

Estimate the size of this query in terms of pages and tuples

- 388 / 97 = 4
- 194/ 97 = 2
- 873 / 97 = 9

Q2

Given **V(R,b) = V(S, b) = V(T, b) = 100**

	0	1	2	3	4	5 - 99
R.b	72	30	10	4	4	4 each
S.b	2	40	16	250	2	2 each
T.b	100	9	7	9	20	9 each
Join Size	14400	10800	1120	900	160	72 each

SELECT **a, b** FROM R, S, T WHERE R.b = S.b AND S.b = T.b

Estimate the size of this query in terms of pages and tuples

- 388 / 97 = 4
- 194/ 97 = 2
- 873 / 97 = 9

Total size = 14400 + 10800 + 1120 + 900 + 160 + 72 * **95** = **42320 tuples**

 # pages = [42320 / 15] = **2822 pages**

1 page can fit 10 tuples of R
1 page can fit 30 attributes
1 result tuple has **2** attributes
1 page can fit **15 tuples**

Q3a

R1(a,b)	R2(b, c)	R3(c, d)	R4(a, d)
T(R1) = 1000	T(R2) = 100	T(R3) = 100	T(R4) = 1000
V(R1, a) = 100	V(R2, b) = 100	V(R3, c) = 10	V(R4, a) = 100
V(R1, b) = 100	V(R2, c) = 10	V(R3, d) = 100	V(R4, d) = 100

cost metric = number of intermediate result tuples

What is the join order selected by the greedy algorithm?

Q3a

R1(a,b)	R2(b, c)	R3(c, d)	R4(a, d)
T(R1) = 1000	T(R2) = 100	T(R3) = 100	T(R4) = 1000
V(R1, a) = 100	V(R2, b) = 100	V(R3, c) = 10	V(R4, a) = 100
V(R1, b) = 100	V(R2, c) = 10	V(R3, d) = 100	V(R4, d) = 100

cost metric = number of intermediate result tuples

$$T(\mathrm{X} \bowtie \mathrm{Y}) = \frac{\|X\| \times \|Y\|}{\max\big(V(X,c), V(Y,c)\big)}$$

What is the join order selected by the greedy algorithm?

- R1 ⋈ R2 = (1000)(100) / 100 = 1000 intermediate tuples (same as R3 ⋈ R4)
- R2 ⋈ R3 = (100)(100) / 10 = 1000 intermediate tuples
- R1 ⋈ R4 = (1000)(1000) / 100 = 10,000 intermediate tuples

We ignore cross products since those definitely have larger intermediate tables

Q3a

R1(a,b)	R2(b, c)	R3(c, d)	R4(a, d)
T(R1) = 1000	T(R2) = 100	T(R3) = 100	T(R4) = 1000
V(R1, a) = 100	V(R2, b) = 100	V(R3, c) = 10	V(R4, a) = 100
V(R1, b) = 100	V(R2, c) = 10	V(R3, d) = 100	V(R4, d) = 100

cost metric = number of intermediate result tuples

$$T(\mathrm{X} \bowtie \mathrm{Y}) = \frac{\|X\| \times \|Y\|}{\max\big(V(X,c), V(Y,c)\big)}$$

What is the join order selected by the greedy algorithm?

- R1 ⋈ R2 = (1000)(100) / 100 = 1000 intermediate tuples (same as R3 ⋈ R4) ✓
- R2 ⋈ R3 = (100)(100) / 10 = 1000 intermediate tuples ✓
- R1 ⋈ R4 = (1000)(1000) / 100 = 10,000 intermediate tuples

Q3a

R1(a,b)	R2(b, c)	R3(c, d)	R4(a, d)
T(R1) = 1000	T(R2) = 100	T(R3) = 100	T(R4) = 1000
V(R1, a) = 100	V(R2, b) = 100	V(R3, c) = 10	V(R4, a) = 100
V(R1, b) = 100	V(R2, c) = 10	V(R3, d) = 100	V(R4, d) = 100

cost metric = number of intermediate result tuples

$$T(\mathrm{X} \bowtie \mathrm{Y}) = \frac{\|X\| \times \|Y\|}{\max\big(V(X,c), V(Y,c)\big)}$$

What is the join order selected by the greedy algorithm?

1. R2 ⋈ R3 = (100)(100) / 10 = 1000 intermediate tuples
2. (R2 ⋈ R3) ⋈ R1 = (1000)(1000) / 100 = 10,000 intermediate tuples

Cost: 11,000

No difference if we use R4 here. Only concern with intermediate result tuples.

Q3a

R1(a,b)	R2(b, c)	R3(c, d)	R4(a, d)
T(R1) = 1000	T(R2) = 100	T(R3) = 100	T(R4) = 1000
V(R1, a) = 100	V(R2, b) = 100	V(R3, c) = 10	V(R4, a) = 100
V(R1, b) = 100	V(R2, c) = 10	V(R3, d) = 100	V(R4, d) = 100

cost metric = number of intermediate result tuples

$$T(\mathrm{X} \bowtie \mathrm{Y}) = \frac{\|X\| \times \|Y\|}{\max\big(V(X,c), V(Y,c)\big)}$$

What is the join order selected by the greedy algorithm?

1. R1 ⋈ R2 = (1000)(100) / 100 = 1000 intermediate tuples
2. (R1 ⋈ R2) ⋈ R3 = (1000)(100) / 10 = 10,000 intermediate tuples

Cost: 11,000

Still the same cost even if we perform R1 ⋈ R2 first

Q3b

R1(a,b)	R2(b, c)	R3(c, d)	R4(a, d)
T(R1) = 1000	T(R2) = 100	T(R3) = 100	T(R4) = 1000
V(R1, a) = 100	V(R2, b) = 100	V(R3, c) = 10	V(R4, a) = 100
V(R1, b) = 100	V(R2, c) = 10	V(R3, d) = 100	V(R4, d) = 100

cost metric = number of intermediate result tuples

What is the globally optimal join ordering?

$$T(\mathrm{X} \bowtie \mathrm{Y}) = \frac{\|X\| \times \|Y\|}{\max\big(V(X,c), V(Y,c)\big)}$$

Q3b

R1(a,b)	R2(b, c)	R3(c, d)	R4(a, d)
T(R1) = 1000	T(R2) = 100	T(R3) = 100	T(R4) = 1000
V(R1, a) = 100	V(R2, b) = 100	V(R3, c) = 10	V(R4, a) = 100
V(R1, b) = 100	V(R2, c) = 10	V(R3, d) = 100	V(R4, d) = 100

cost metric = number of intermediate result tuples

$$T(\mathrm{X} \bowtie \mathrm{Y}) = \frac{\|X\| \times \|Y\|}{\max\big(V(X,c), V(Y,c)\big)}$$

What is the globally optimal join ordering?

1. R1 ⋈ R2 = (1000)(100) / 100 = 1000 intermediate tuples
2. R3 ⋈ R4 = (1000)(100) / 100 = 1000 intermediate tuples

(R1 ⋈ R2) ⋈ (R3 ⋈ R4)

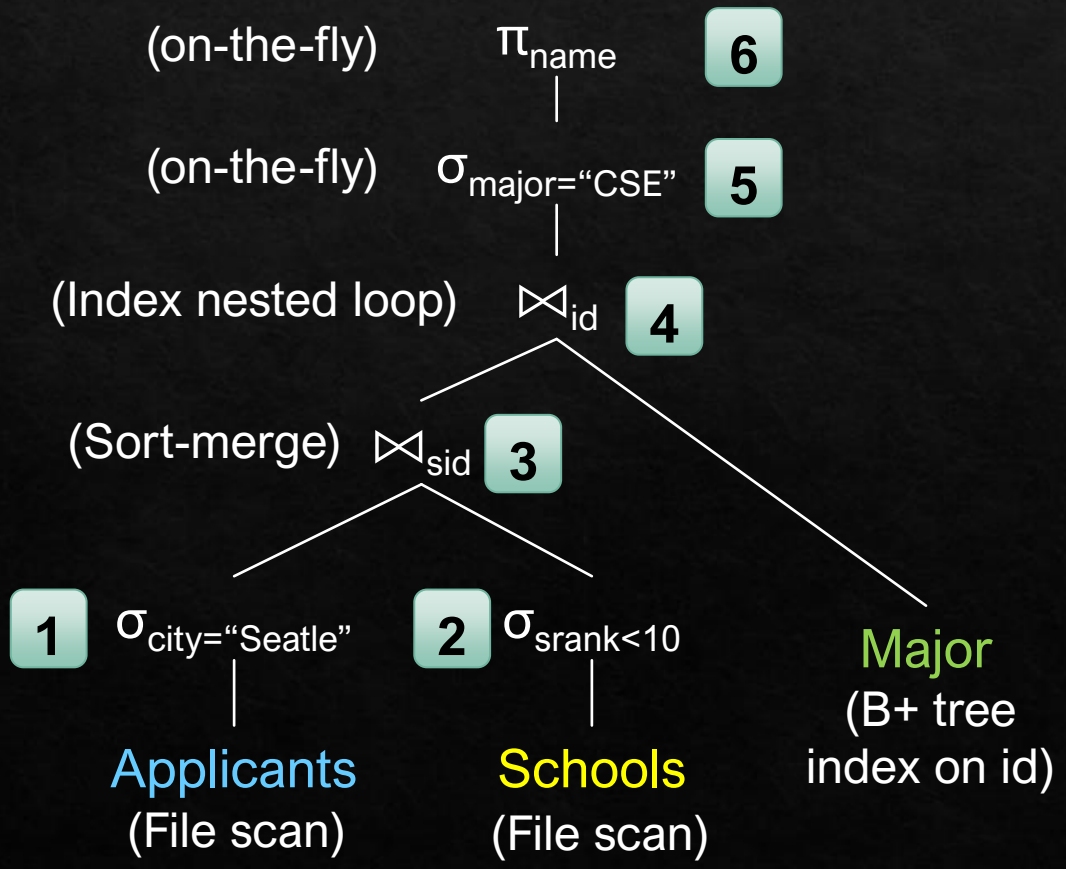
Cost: 2,000

The best strategy in the left deep tree space is far from optimal!

```
SELECT A.name
FROM Applicants A, Schools S, Major M
WHERE A.sid = S.sid
AND A.id = M.id
AND A.city = 'Seattle'
AND S.rank < 10
AND M.major = 'CSE'
```

Q4a

- What is the cost of the query plan below?

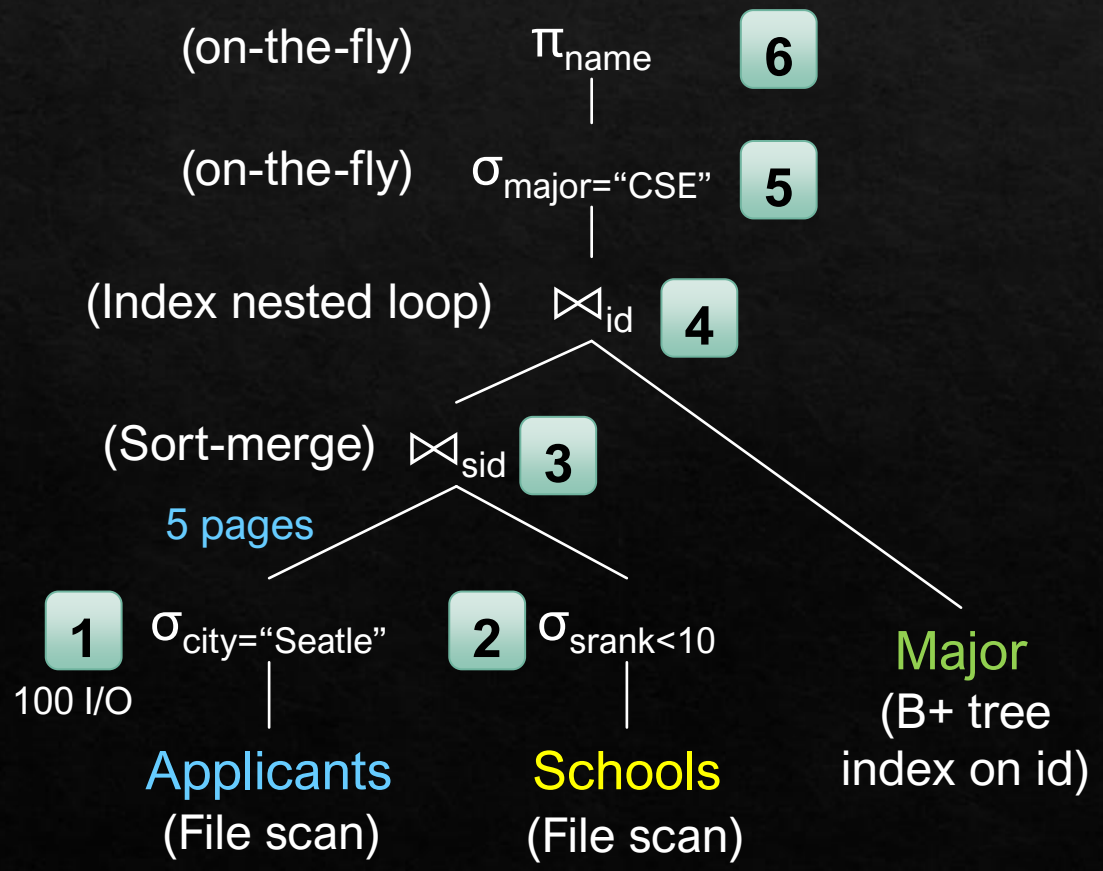


```
SELECT A.name
FROM Applicants A, Schools S, Major M
WHERE A.sid = S.sid
AND A.id = M.id
AND A.city = 'Seattle'
AND S.rank < 10
AND M.major = 'CSE'
```

Q4a

Relation	Cardinality	#pages	Pri Key
A(id, name, city, sid)	2000	100	id
S(sid, sname, srank)	100	10	sid
M(id, major)	3000	200	(id, maj)

- What is the cost of the query plan below?



(1) Select from Applicants

Read cost: **100 I/Os**

Output size: 1/20 * 2000 = **100 tuples = 5 pages**

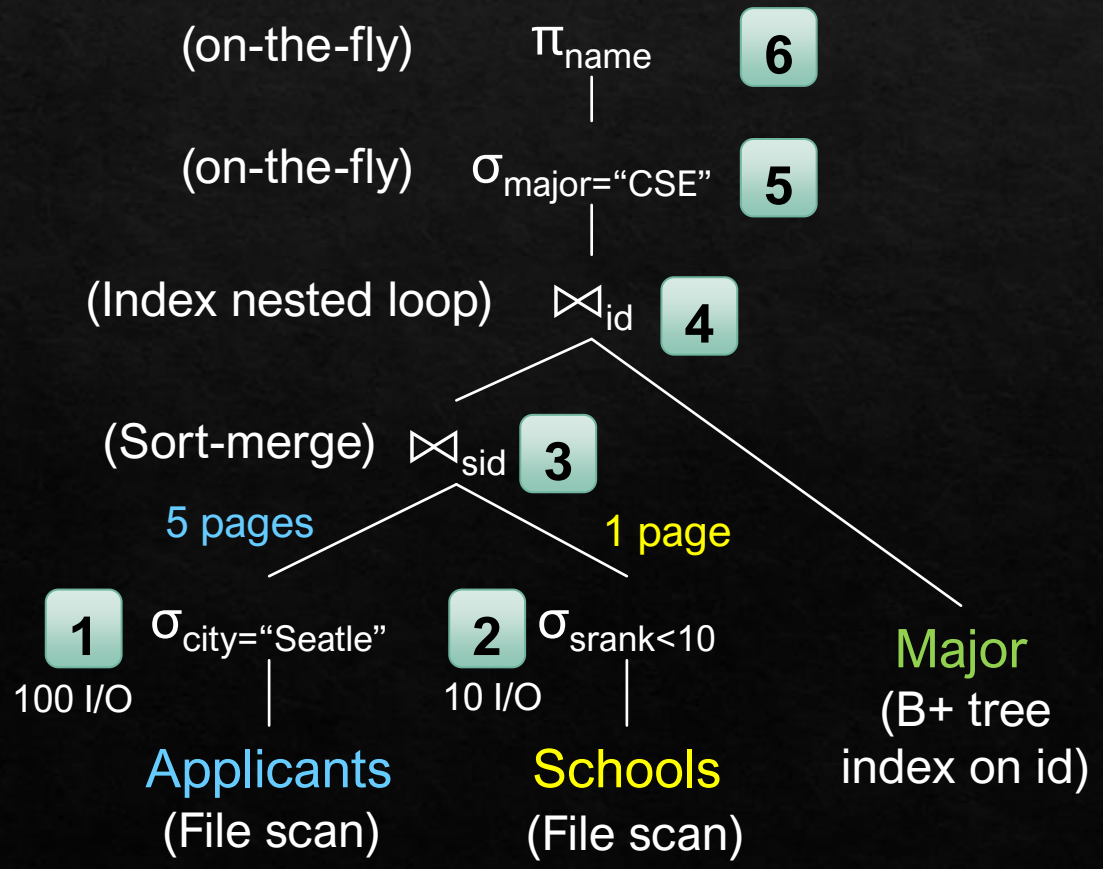
Keep output in memory – We have 10 buffer pages

```
SELECT A.name
FROM Applicants A, Schools S, Major M
WHERE A.sid = S.sid
AND A.id = M.id
AND A.city = 'Seattle'
AND S.rank < 10
AND M.major = 'CSE'
```

Q4a

Relation	Cardinality	#pages	Pri Key
A(id, name, city, sid)	2000	100	id
S(sid, sname, srank)	100	10	sid
M(id, major)	3000	200	(id, maj)

- What is the cost of the query plan below?



(2) Select from Schools

Read cost: **10 I/Os**

Output size: 9/100 * 100 = **9 tuples = 1 page**

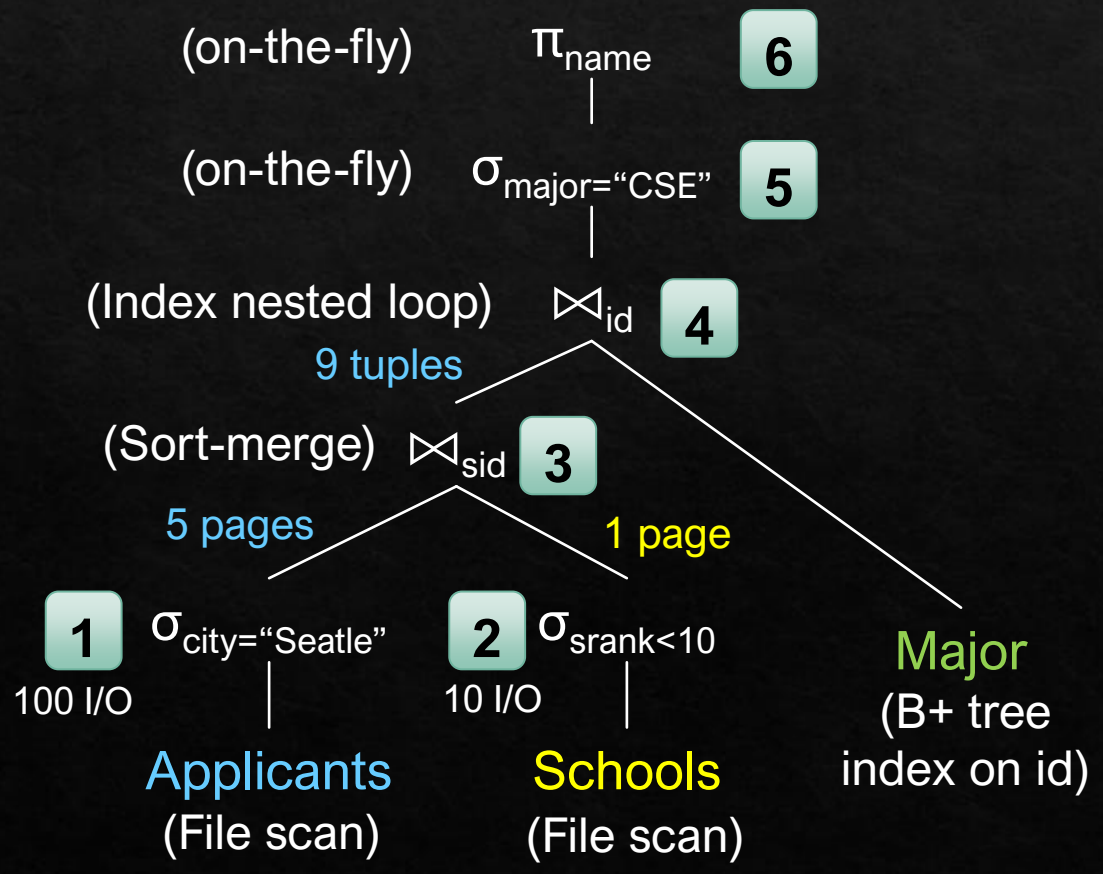
Keep output in memory – We have 10 buffer pages

```
SELECT A.name
FROM Applicants A, Schools S, Major M
WHERE A.sid = S.sid
AND A.id = M.id
AND A.city = 'Seattle'
AND S.rank < 10
AND M.major = 'CSE'
```

Q4a

Relation	Cardinality	#pages	Pri Key
A(id, name, city, sid)	2000	100	id
S(sid, sname, srank)	100	10	sid
M(id, major)	3000	200	(id, maj)

- What is the cost of the query plan below?



(3) Sort-merge join

Perform in-memory sort of 100 applicants (5 pages) and 9 schools (1 page), then merge in memory

Cost: **0 I/O**

Output size: **9 tuples**

Only 9 applicants have a join result, because even after applying σ_{city}="Seattle" on Applicants, there are still 100 unique sids among the 100 applicants

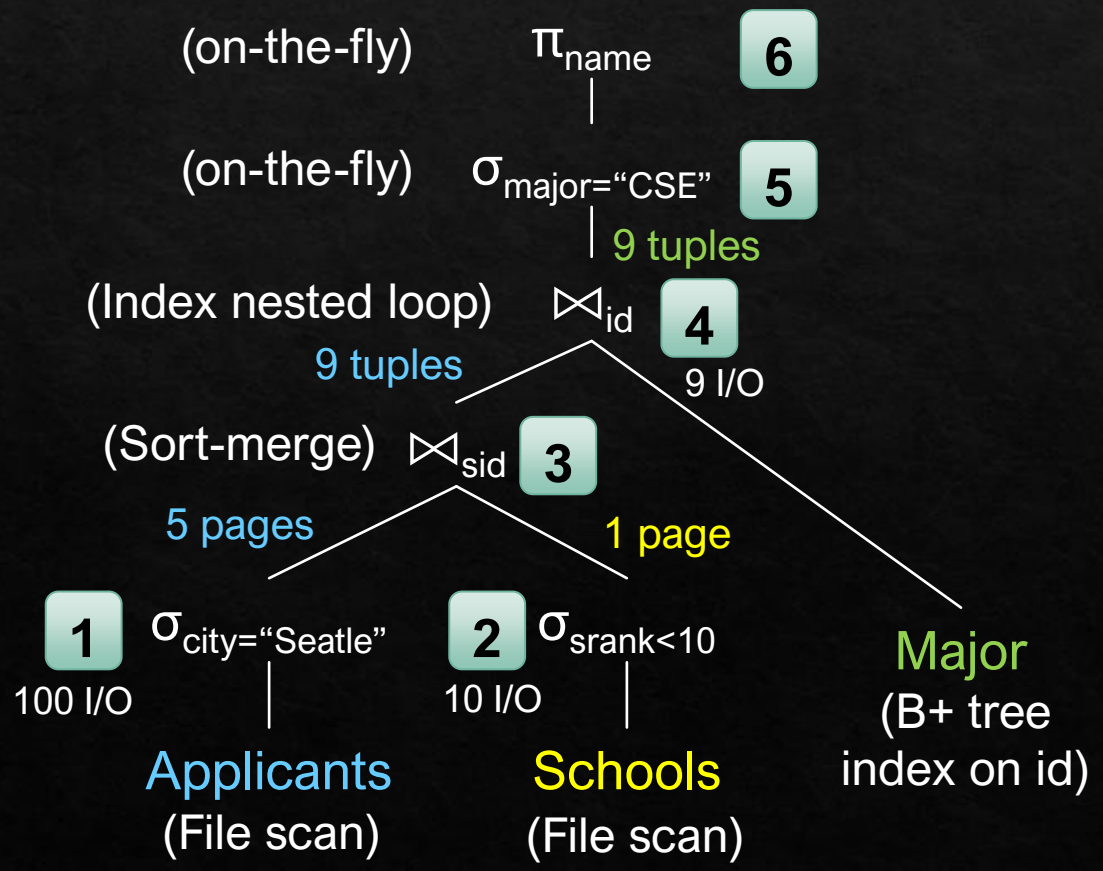
Independence assumption
The distribution of sid values and city values are independent

```
SELECT A.name
FROM Applicants A, Schools S, Major M
WHERE A.sid = S.sid
AND A.id = M.id
AND A.city = 'Seattle'
AND S.rank < 10
AND M.major = 'CSE'
```

Q4a

Relation	Cardinality	#pages	Pri Key
A(id, name, city, sid)	2000	100	id
S(sid, sname, srank)	100	10	sid
M(id, major)	3000	200	(id, maj)

- What is the cost of the query plan below?



(4) Index-nested loop join

For each of the 9 tuples, look up the B+ tree (assume format 2), and fetch the **9 corresponding data pages.**

No I/O cost for traversing tree as the index pages are in memory

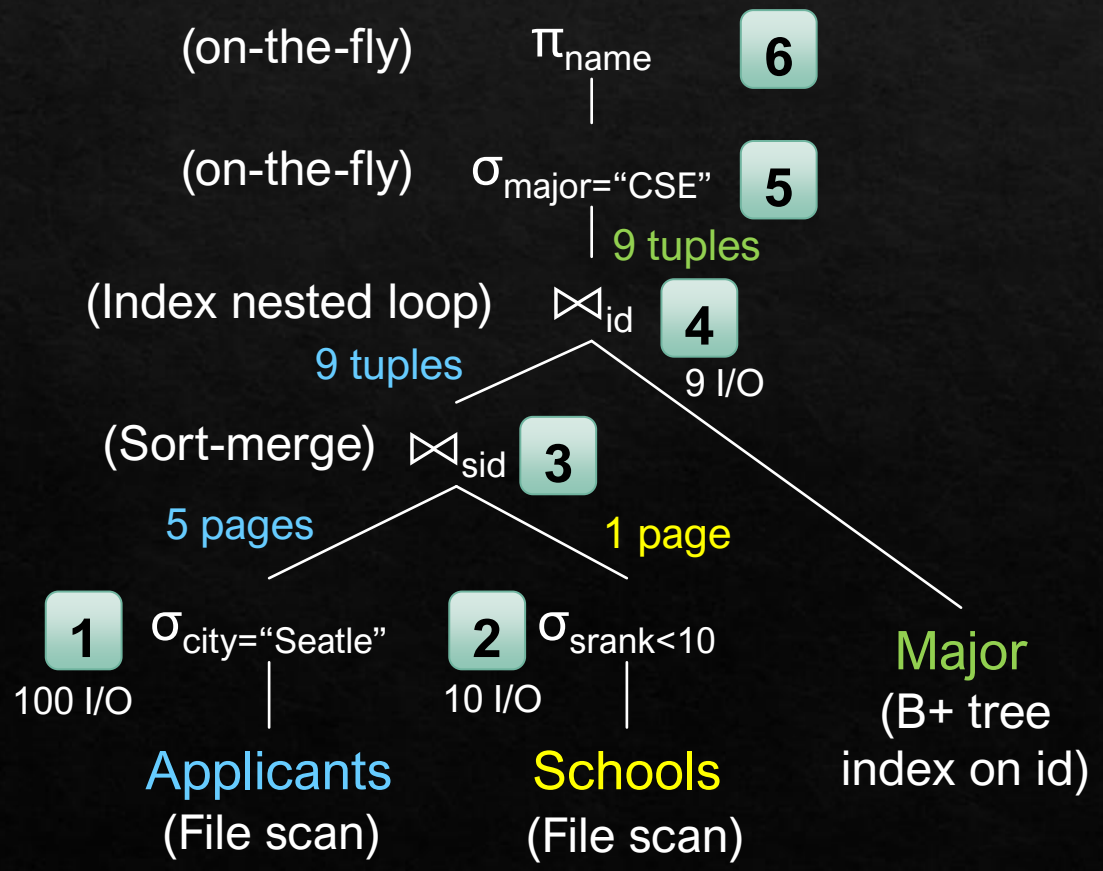
Cost: **9 I/O**

```
SELECT A.name
FROM Applicants A, Schools S, Major M
WHERE A.sid = S.sid
AND A.id = M.id
AND A.city = 'Seattle'
AND S.rank < 10
AND M.major = 'CSE'
```

Q4a

Relation	Cardinality	#pages	Pri Key
A(id, name, city, sid)	2000	100	id
S(sid, sname, srank)	100	10	sid
M(id, major)	3000	200	(id, maj)

- What is the cost of the query plan below?



(5) Selection on-the-fly

(6) Projection on-the-fly

No additional cost as unwanted tuples and attributes are just trimmed off from the input tuples

Cost: **0 I/O**

Total cost: 119 I/O

Q4b

Draw two query plans for the above query that the optimizer would NOT consider. For each query plan, indicate why it would not be considered.

Q4b

Draw two query plans for the above query that the optimizer would NOT consider. For each query plan, indicate why it would not be considered.

1. Any non left deep tree plans are not considered
   e.g. right deep tree
2. Plans involving a join between School and Major (too expensive as it is a cross product)

End