

CSE 444: Database Internals

Lecture 7 Query Execution and Operator Algorithms (part 1)

CSE 444 - Spring 2019

1

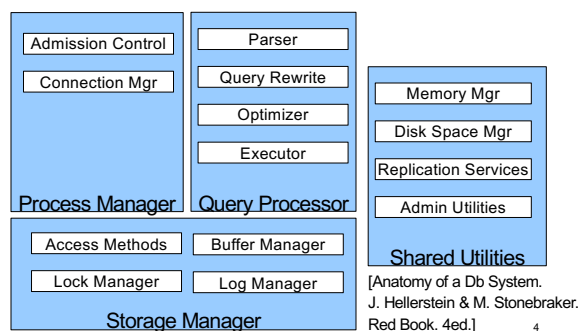
What We Have Learned So Far

- Overview of the architecture of a DBMS
- Access methods
 - Heap files, sequential files, Indexes (hash or B+ trees)
- Role of buffer manager
- Practiced the concepts in hw1 and lab1

CSE 444 - Spring 2019

3

DBMS Architecture



4

Next Lectures

- How to answer queries **efficiently!**
 - **Physical query plans and operator algorithms**
- How to automatically find good query plans
 - How to compute the cost of a complete plan
 - How to pick a good query plan for a query
 - i.e., Query optimization

CSE 444 - Spring 2019

5

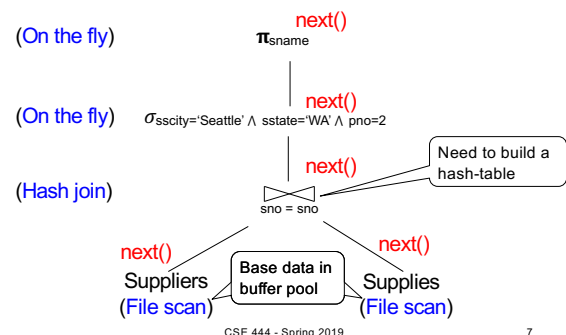
Query Execution Bottom Line

- SQL query transformed into **physical plan**
 - **Access path selection** for each relation
 - **Implementation choice** for each operator
 - **Scheduling decisions** for operators
 - Single-threaded or parallel, pipelined or with materialization, etc.
- Execution of the physical plan is pull-based
- Operators **given a limited amount of memory**

CSE 444 - Spring 2019

6

Pipelined Query Execution



CSE 444 - Spring 2019

7

Memory Management

Each operator:

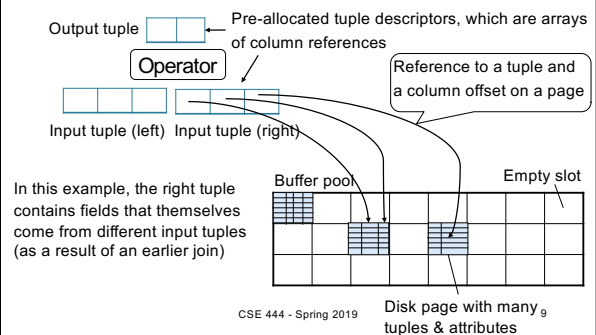
- **Pre-allocates heap space for input/output tuples**
 - Option 1: Array of pointers to base data in buffer pool
 - Option 2: New tuples on the heap
- **Allocates memory for its internal state**
 - Either on heap or in buffer pool (depends on system)

DBMS **limits** how much memory each operator, or each query can use

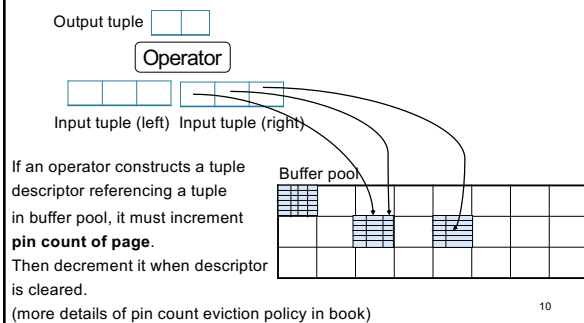
CSE 444 - Spring 2019

8

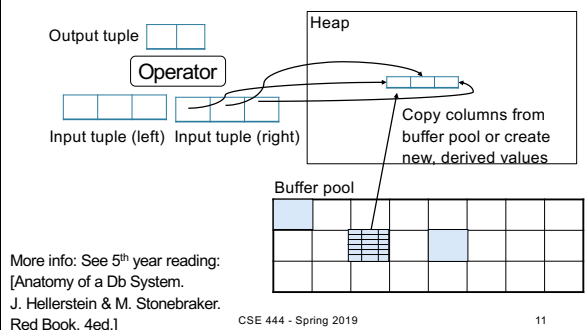
In Flight Tuples (option 1)



In Flight Tuples (option 1)



In Flight Tuples (option 2)



Operator Algorithms (Quick review from 344 today & new algorithms next time)

CSE 444 - Spring 2019

12

Operator Algorithms

Design criteria

- Cost: IO, CPU, Network
- Memory utilization
- Load balance (for parallel operators)

CSE 444 - Spring 2019

13

Cost Parameters

- **Cost = total number of I/Os**
 - This is a simplification that ignores CPU, network
- **Parameters:**
 - $B(R)$ = # of blocks (i.e., pages) for relation R
 - $T(R)$ = # of tuples in relation R
 - $V(R, a)$ = # of distinct values of attribute a
 - When a is a key, $V(R, a) = T(R)$
 - When a is not a key, $V(R, a)$ can be anything $< T(R)$

CSE 444 - Spring 2019

14

Convention

- Cost = the cost of **reading** operands from disk
- Cost of **writing** the **final** result to disk is *not included*; need to count it separately when applicable

CSE 444 - Spring 2019

15

Outline

- **Join operator algorithms**
 - Review {
 - One-pass algorithms (Sec. 15.2 and 15.3)
 - Index-based algorithms (Sec 15.6)
 - New {
 - Two-pass algorithms (Sec 15.4 and 15.5)
- Note about readings:
 - In class, we discuss only algorithms for joins
 - Other operators are easier: read the book

CSE 444 - Spring 2019

16

Join Algorithms

- Hash join
- Nested loop join
- Sort-merge join

CSE 444 - Spring 2019

17

Hash Join

Hash join: $R \bowtie S$

- Scan R, build buckets in main memory
- Then scan S and join
- Cost: $B(R) + B(S)$
- One-pass algorithm when $B(R) \leq M$

CSE 444 - Spring 2019

18

Hash Join Example

Patient(pid, name, address)

Insurance(pid, provider, policy_nb)

Patient \bowtie Insurance

Patient

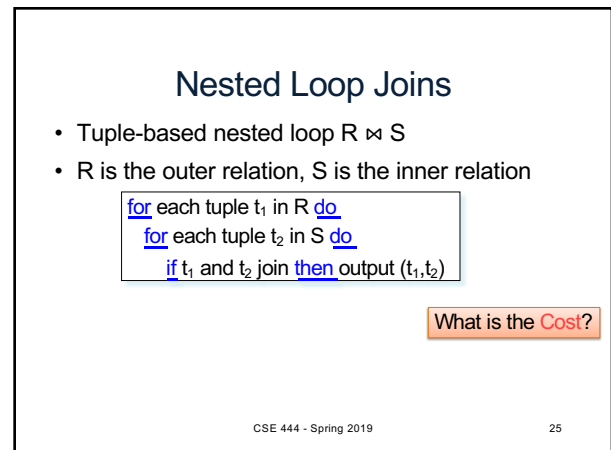
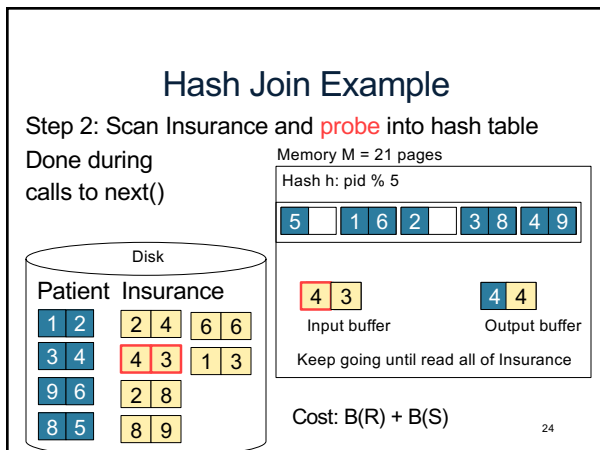
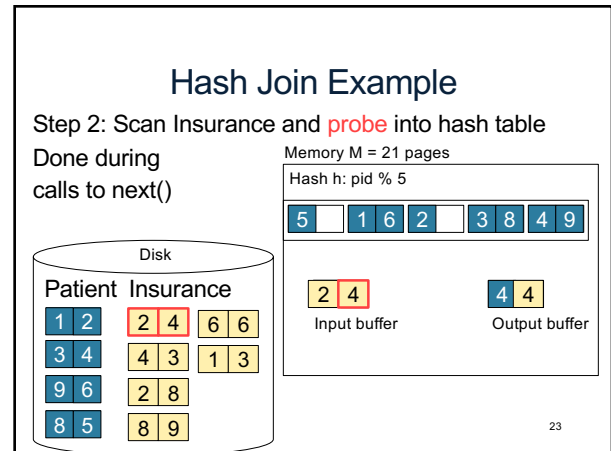
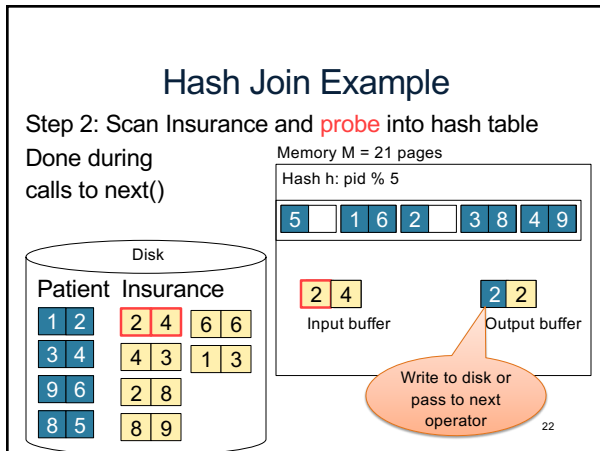
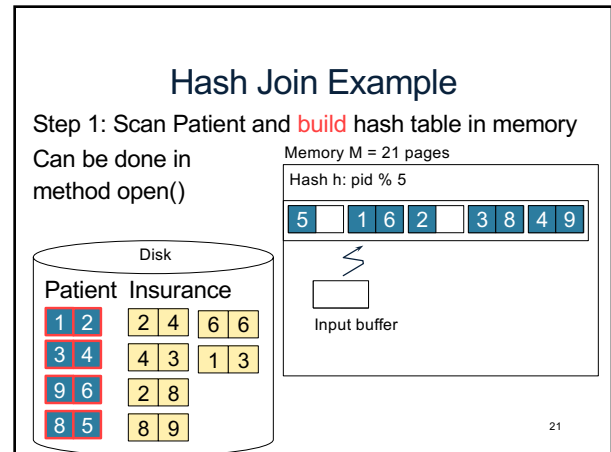
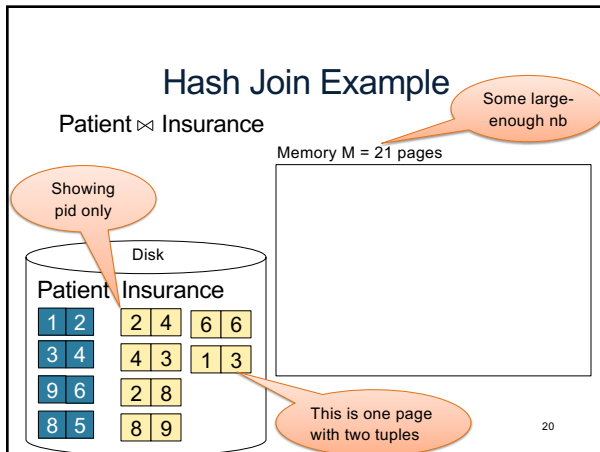
1	'Bob'	'Seattle'
2	'Ela'	'Everett'
3	'Jill'	'Kent'
4	'Joe'	'Seattle'

Insurance

2	'Blue'	123
4	'Prem'	432
4	'Prem'	343
	'GrpH'	554

Two tuples per page

19



Nested Loop Joins

- Tuple-based nested loop $R \bowtie S$
- R is the outer relation, S is the inner relation

```
for each tuple  $t_1$  in R do
  for each tuple  $t_2$  in S do
    if  $t_1$  and  $t_2$  join then output ( $t_1, t_2$ )
```

- Cost: $B(R) + T(R) B(S)$
- Multiple-pass since S is read many times

What is the Cost?

CSE 444 - Spring 2019

26

Page-at-a-time Refinement

```
for each page of tuples r in R do
  for each page of tuples s in S do
    for all pairs of tuples  $t_1$  in r,  $t_2$  in s
      if  $t_1$  and  $t_2$  join then output ( $t_1, t_2$ )
```

What is the Cost?

CSE 444 - Spring 2019

27

Page-at-a-time Refinement

```
for each page of tuples r in R do
  for each page of tuples s in S do
    for all pairs of tuples  $t_1$  in r,  $t_2$  in s
      if  $t_1$  and  $t_2$  join then output ( $t_1, t_2$ )
```

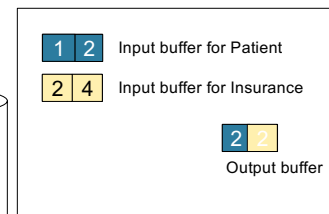
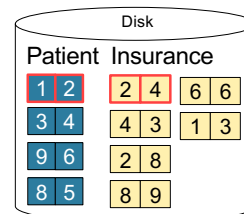
- Cost: $B(R) + B(R)B(S)$

What is the Cost?

CSE 444 - Spring 2019

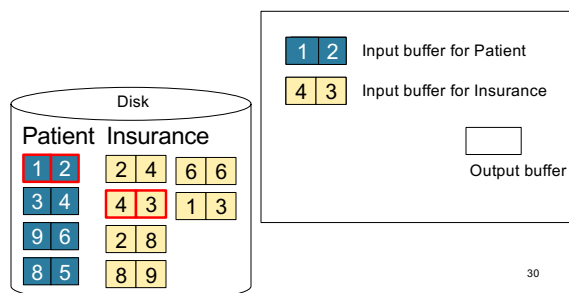
28

Page-at-a-time Refinement



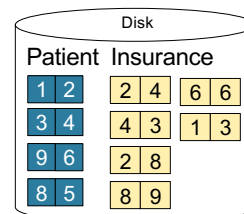
29

Page-at-a-time Refinement



30

Page-at-a-time Refinement



Keep going until read all of Insurance
Then repeat for next page of Patient... until end of Patient

Cost: $B(R) + B(R)B(S)$

31

Block-Nested-Loop Refinement

```

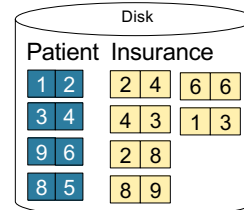
for each group of M-1 pages r in R do
  for each page of tuples s in S do
    for all pairs of tuples t1 in r, t2 in s
      if t1 and t2 join then output (t1, t2)
    
```

What is the Cost?

32

Block Memory Refinement

M=3



Input buffer for Patient

Input buffer for Insurance

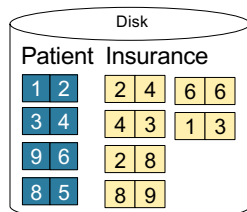
No output buffer: stream to output

Cost:

33

Block Memory Refinement

M=3



Input buffer for Patient

Input buffer for Insurance

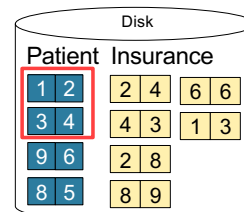
No output buffer: stream to output

Cost:

34

Block Memory Refinement

M=3



Input buffer for Patient

Input buffer for Insurance

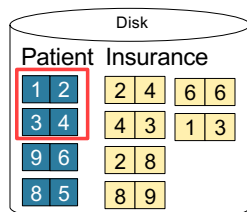
No output buffer: stream to output

Cost:

35

Block Memory Refinement

M=3



Input buffer for Patient

Input buffer for Insurance

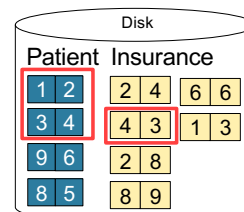
No output buffer: stream to output

Cost:

36

Block Memory Refinement

M=3



Input buffer for Patient

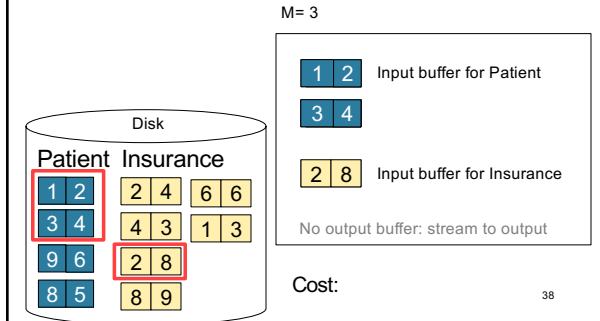
Input buffer for Insurance

No output buffer: stream to output

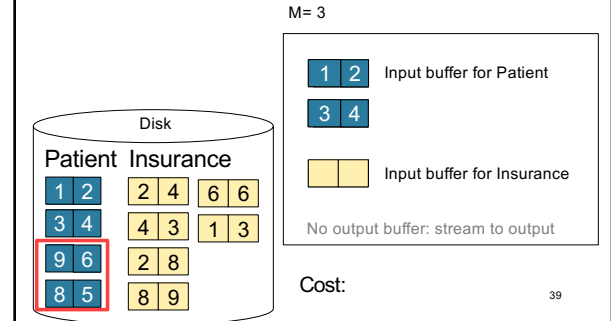
Cost:

37

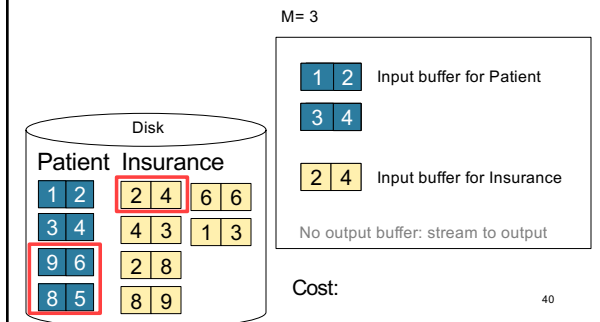
Block Memory Refinement



Block Memory Refinement



Block Memory Refinement



Block-Nested-Loop Refinement

```

for each group of M-1 pages r in R do
  for each page of tuples s in S do
    for all pairs of tuples t1 in r, t2 in s
      if t1 and t2 join then output (t1, t2)
    
```

What is the Cost?

41

Block-Nested-Loop Refinement

```

for each group of M-1 pages r in R do
  for each page of tuples s in S do
    for all pairs of tuples t1 in r, t2 in s
      if t1 and t2 join then output (t1, t2)
    
```

- Cost: $B(R) + B(R)B(S)/(M-1)$

What is the Cost?

42

Sort-Merge Join

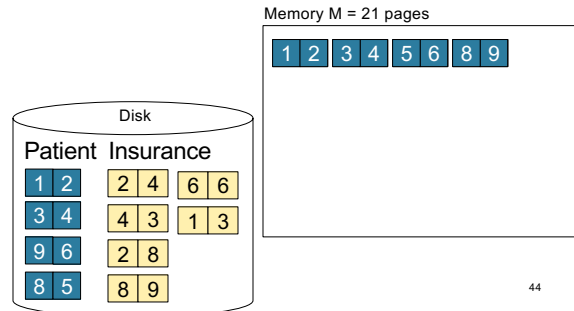
Sort-merge join: $R \bowtie S$

- Scan R and sort in main memory
- Scan S and sort in main memory
- Merge R and S
- Cost: $B(R) + B(S)$
- One pass algorithm when $B(S) + B(R) \leq M$
- Typically, this is NOT a one pass algorithm

43

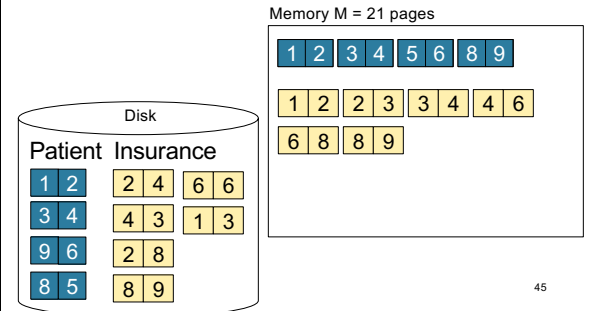
Sort-Merge Join Example

Step 1: Scan Patient and **sort** in memory



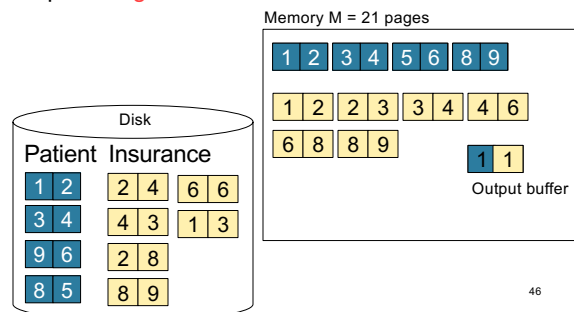
Sort-Merge Join Example

Step 2: Scan Insurance and **sort** in memory



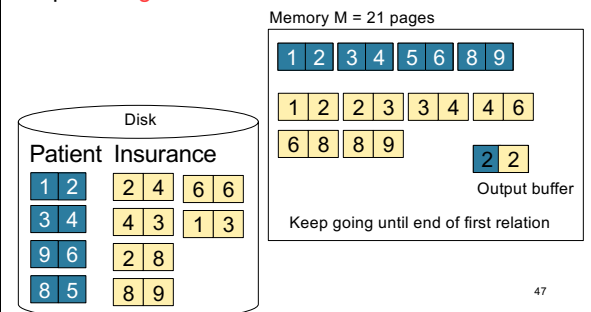
Sort-Merge Join Example

Step 3: **Merge** Patient and Insurance



Sort-Merge Join Example

Step 3: **Merge** Patient and Insurance



Outline

- **Join operator algorithms**
 - One-pass algorithms (Sec. 15.2 and 15.3)
 - Index-based algorithms (Sec 15.6)
 - Two-pass algorithms (Sec 15.4 and 15.5)

Index Based Selection

Selection on equality: $\sigma_{a=v}(R)$

- $B(R)$ = size of R in blocks
- $T(R)$ = number of tuples in R
- $V(R, a)$ = # of distinct values of attribute a

Index Based Selection

Selection on equality: $\sigma_{a=v}(R)$

- $B(R)$ = size of R in blocks
- $T(R)$ = number of tuples in R
- $V(R, a)$ = # of distinct values of attribute a

What is the cost in each case?

- Clustered index on a :
- Unclustered index on a :

CSE 444 - Spring 2019

50

Index Based Selection

Selection on equality: $\sigma_{a=v}(R)$

- $B(R)$ = size of R in blocks
- $T(R)$ = number of tuples in R
- $V(R, a)$ = # of distinct values of attribute a

What is the cost in each case?

- Unclustered index on a : $T(R)/V(R, a)$
- Clustered index on a : $B(R)/V(R, a)$

CSE 444 - Spring 2019

51

Index Based Selection

Selection on equality: $\sigma_{a=v}(R)$

- $B(R)$ = size of R in blocks
- $T(R)$ = number of tuples in R
- $V(R, a)$ = # of distinct values of attribute a

What is the cost in each case?

- Clustered index on a : $B(R)/V(R, a)$
- Unclustered index on a : $T(R)/V(R, a)$

Note: we ignore I/O cost for index pages

52

Index Based Selection

• Example:

$B(R) = 2000$
 $T(R) = 100,000$
 $V(R, a) = 20$

cost of $\sigma_{a=v}(R) = ?$

- Table scan:
- Index based selection:

CSE 444 - Spring 2019

53

Index Based Selection

• Example:

$B(R) = 2000$
 $T(R) = 100,000$
 $V(R, a) = 20$

cost of $\sigma_{a=v}(R) = ?$

- Table scan: $B(R) = 2,000$ I/Os
- Index based selection:

CSE 444 - Spring 2019

54

Index Based Selection

• Example:

$B(R) = 2000$
 $T(R) = 100,000$
 $V(R, a) = 20$

cost of $\sigma_{a=v}(R) = ?$

- Table scan: $B(R) = 2,000$ I/Os
- Index based selection:
 - If index is clustered:
 - If index is unclustered:

CSE 444 - Spring 2019

55

Index Based Selection

- Example: $B(R) = 2000$
 $T(R) = 100,000$
 $V(R, a) = 20$ cost of $\sigma_{a=v}(R) = ?$
- Table scan: $B(R) = 2,000$ I/Os
- Index based selection:
 - If index is clustered: $B(R)/V(R,a) = 100$ I/Os
 - If index is unclustered:

CSE 444 - Spring 2019

56

Index Based Selection

- Example: $B(R) = 2000$
 $T(R) = 100,000$
 $V(R, a) = 20$ cost of $\sigma_{a=v}(R) = ?$
- Table scan: $B(R) = 2,000$ I/Os
- Index based selection:
 - If index is clustered: $B(R)/V(R,a) = 100$ I/Os
 - If index is unclustered: $T(R)/V(R,a) = 5,000$ I/Os

CSE 444 - Spring 2019

57

Index Based Selection

- Example: $B(R) = 2000$
 $T(R) = 100,000$
 $V(R, a) = 20$ cost of $\sigma_{a=v}(R) = ?$
- Table scan: $B(R) = 2,000$ I/Os
- Index based selection:
 - If index is clustered: $B(R)/V(R,a) = 100$ I/Os
 - If index is unclustered: $T(R)/V(R,a) = 5,000$ I/Os

Lesson: Don't build unclustered indexes when $V(R,a)$ is small !

CSE 444 - Spring 2019

58

Index Nested Loop Join

$R \bowtie S$

- Assume S has an index on the join attribute
- Iterate over R , for each tuple fetch corresponding tuple(s) from S
- Cost:
 - If index on S is clustered: $B(R) + T(R)B(S)/V(S,a)$
 - If index on S is unclustered: $B(R) + T(R)T(S)/V(S,a)$

CSE 444 - Spring 2019

59