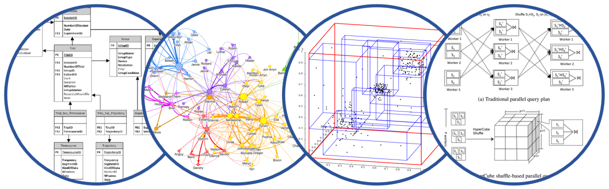


Database System Internals

Query Execution and Algorithms

Paul G. Allen School of Computer Science and Engineering
University of Washington, Seattle

1

Announcements

- Lab 2 (Operator Algorithms) released
  - Part 1 January 29
  - Part 2 February 5
- Lab 2 is published within a new “lab2” branch of the upstream repo
- You will pull the lab 2 branch into your *master* branch, lab 2 README has instructions

https://gitlab.cs.washington.edu/cse444-21wi/simple-db/-/blob/lab2/README.md

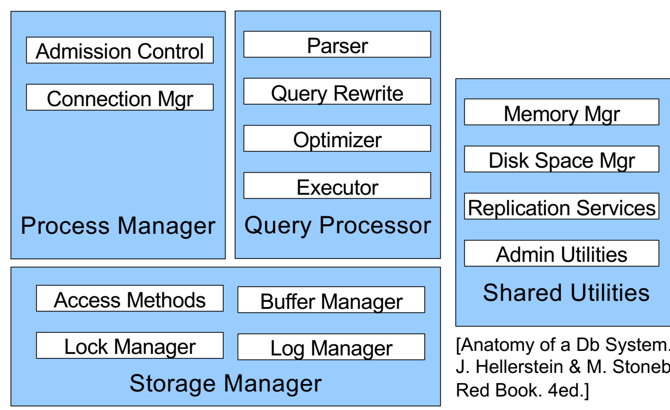
2

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

3

DBMS Architecture



[Anatomy of a Db System.
J. Hellerstein & M. Stonebraker.
Red Book. 4ed.]

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

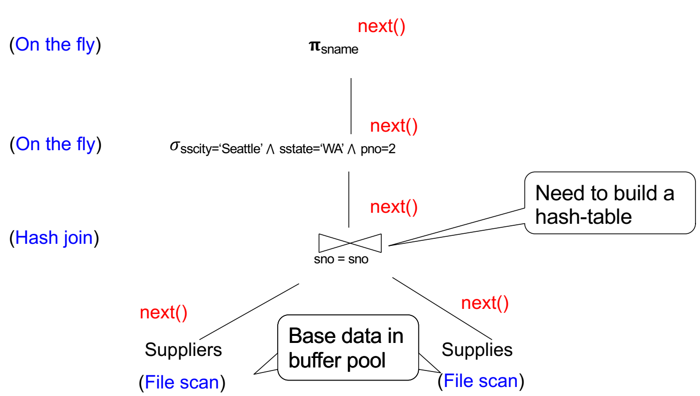
5

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*

6

Pipelined Query Execution



7

Memory Management

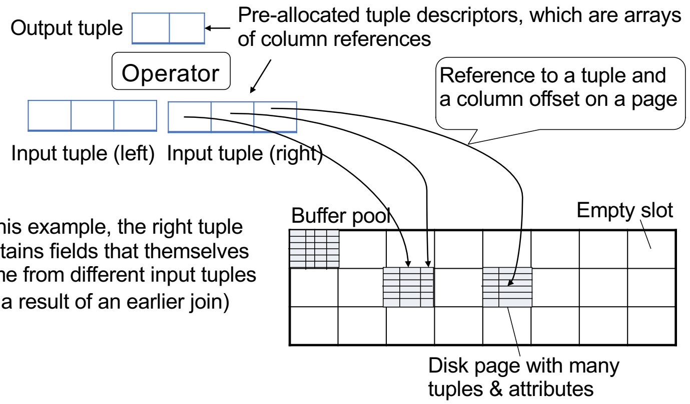
Each operator:

- 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)

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

8

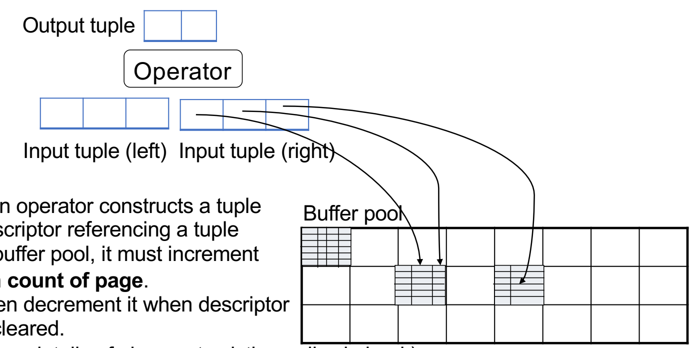
In Flight Tuples (option 1)



In this example, the right tuple contains fields that themselves come from different input tuples (as a result of an earlier join)

9

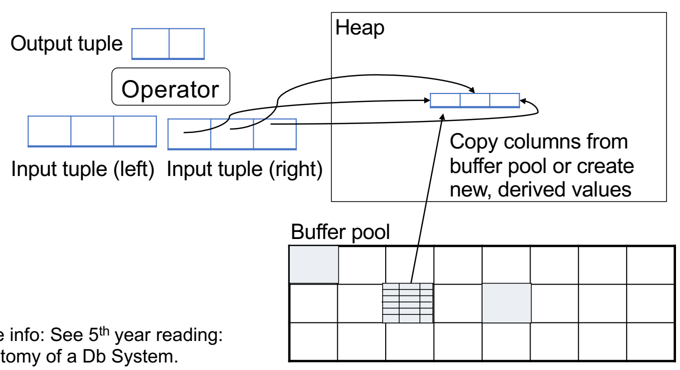
In Flight Tuples (option 1)



If an operator constructs a tuple descriptor referencing a tuple in buffer pool, it must increment **pin count of page**.
Then decrement it when descriptor is cleared.
(more details of pin count eviction policy in book)

10

In Flight Tuples (option 2)



More info: See 5th year reading:
[Anatomy of a Db System.
J. Hellerstein & M. Stonebraker.
Red Book. 4ed.]

11

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

12

Operator Algorithms

Design criteria

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

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)**

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

15

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)
- Note about readings:
  - In class, we discuss only algorithms for joins
  - Other operators are easier: book has extra details

16

Join Algorithms

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

17

Hash Join

Hash join: R ⋈ S

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

18

Hash Join Example

Patient(pid, name, address)

Insurance(pid, provider, policy_nb)

Patient ⋈ Insurance

Two tuples per page

Patient

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

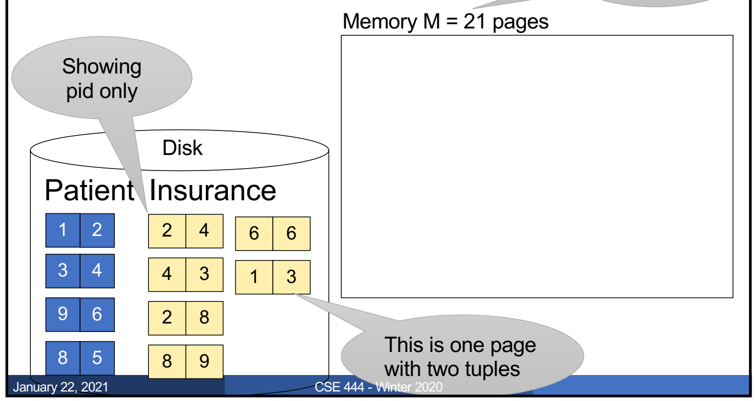
Insurance

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

19

Hash Join Example

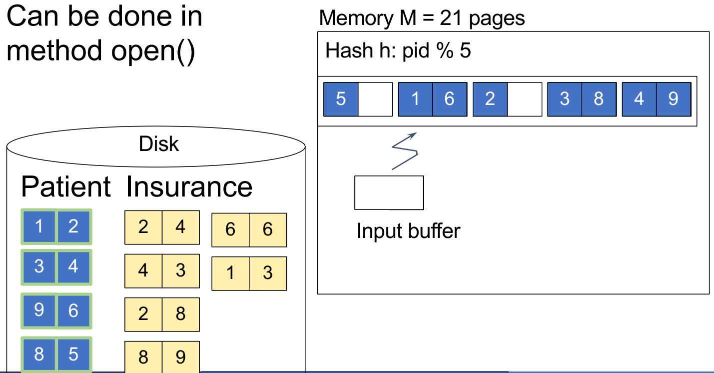
Patient ⋈ Insurance



20

Hash Join Example

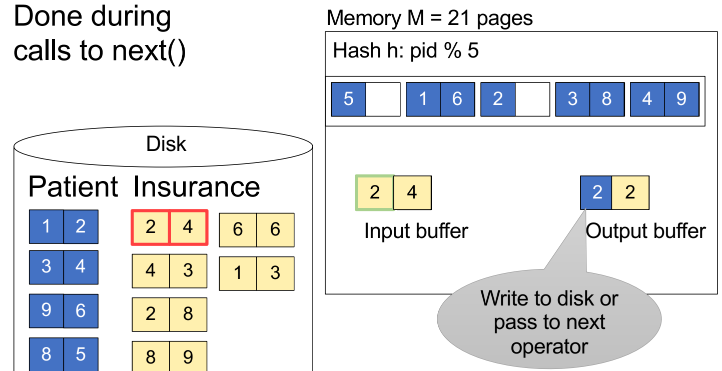
Step 1: Scan Patient and build hash table in memory
Can be done in method open()



21

Hash Join Example

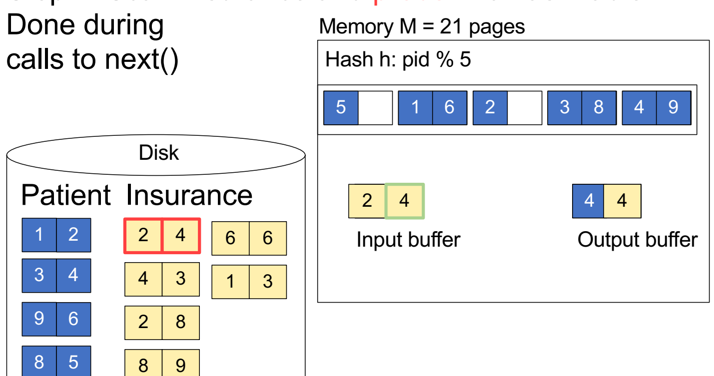
Step 2: Scan Insurance and probe into hash table
Done during calls to next()



22

Hash Join Example

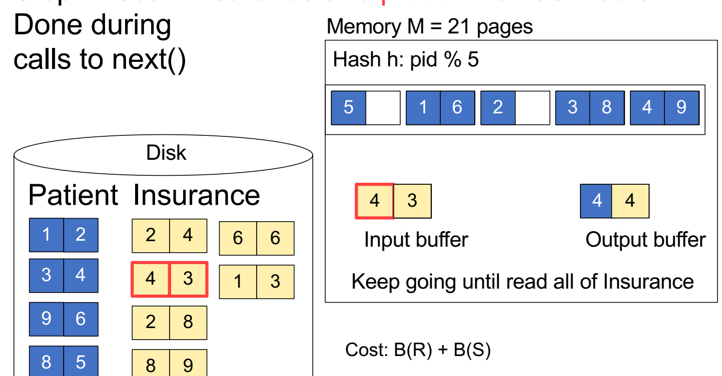
Step 2: Scan Insurance and probe into hash table
Done during calls to next()



23

Hash Join Example

Step 2: Scan Insurance and probe into hash table
Done during calls to next()



24

Nested Loop Joins

- Tuple-based nested loop R ⋈ S
- R is the outer relation, S is the inner relation

```
for each tuple t1 in R do
  for each tuple t2 in S do
    if t1 and t2 join then output (t1,t2)
```

What is the Cost?

25

Nested Loop Joins

- Tuple-based nested loop R ⋈ S
- R is the outer relation, S is the inner relation

```
for each tuple t1 in R do
  for each tuple t2 in S do
    if t1 and t2 join then output (t1,t2)
```

What is the Cost?

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

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 t1 in r, t2 in s
      if t1 and t2 join then output (t1,t2)
```

What is the Cost?

27

Page-at-a-time Refinement

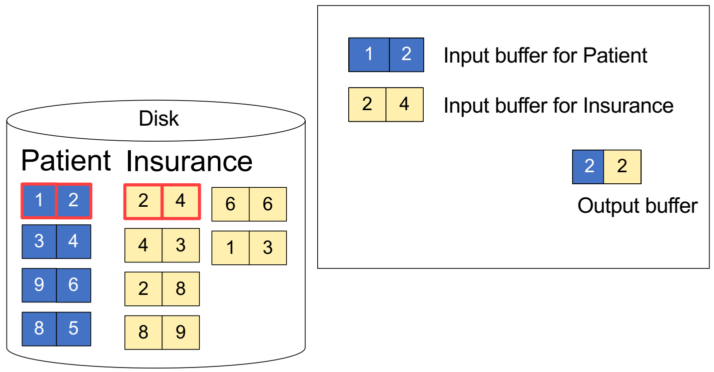
```
for each page of tuples r in R do
  for each page of tuples s in S do
    for all pairs of tuples t1 in r, t2 in s
      if t1 and t2 join then output (t1,t2)
```

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

What is the Cost?

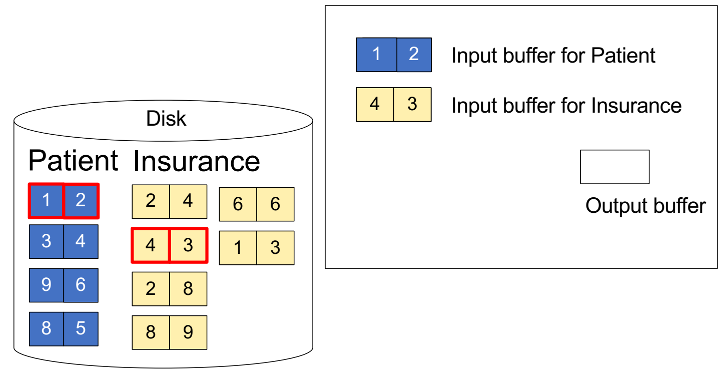
28

Page-at-a-time Refinement



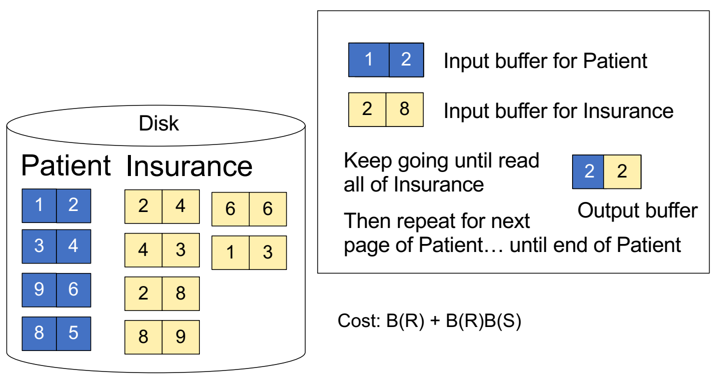
29

Page-at-a-time Refinement



30

Page-at-a-time Refinement



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

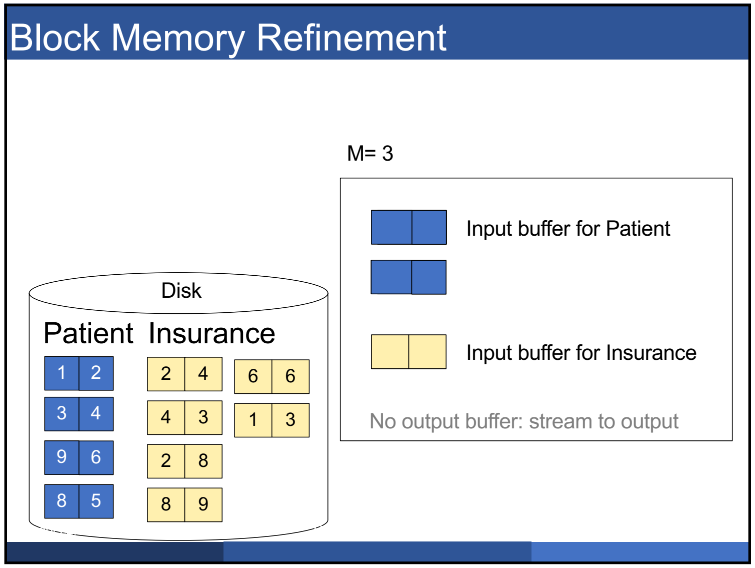
31

Block-Memory Refinement

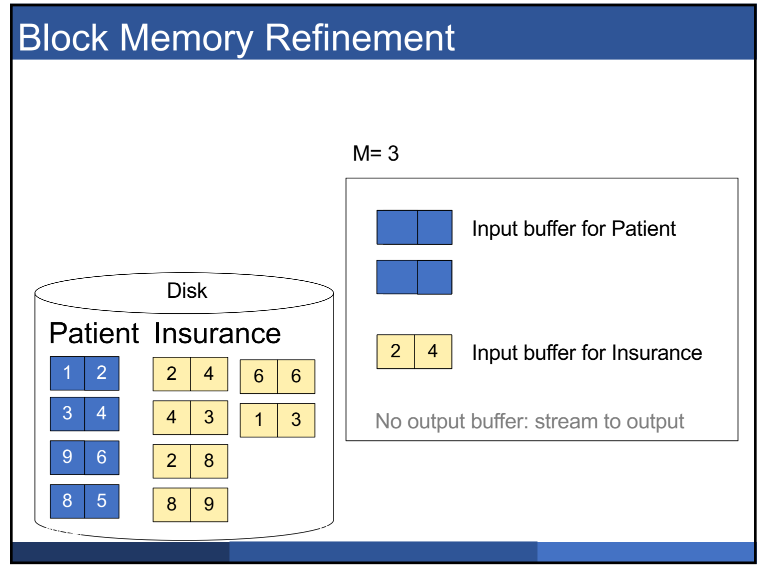
```
for each group of M-1 pages r in R do
  for each page of tuples s in S do
    for all pairs of tuples t1 in r, t2 in s
      if t1 and t2 join then output (t1,t2)
```

What is the Cost?

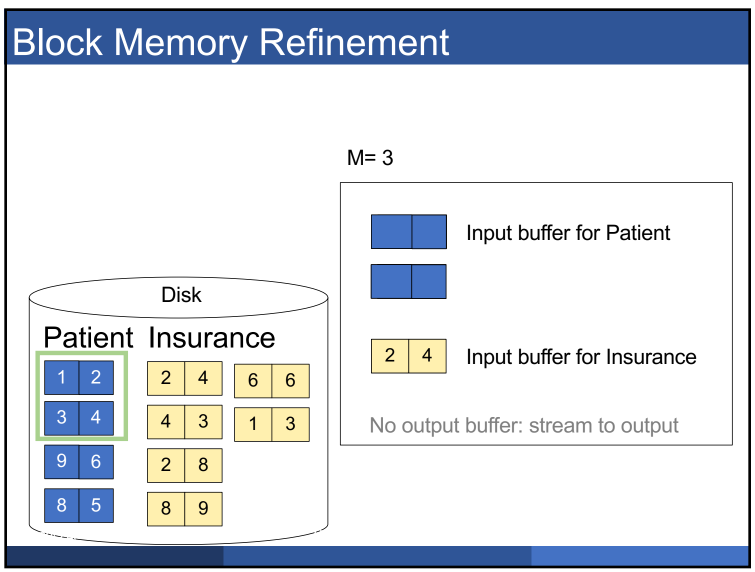
32


Block Memory Refinement
M= 3
Input buffer for Patient
Input buffer for Insurance
No output buffer: stream to output
Disk
Patient
Insurance

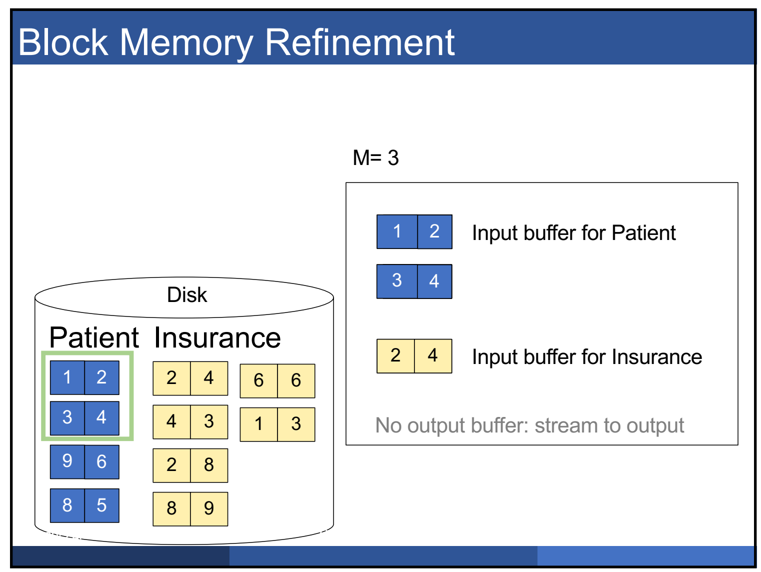
33


Block Memory Refinement
M= 3
Input buffer for Patient
Input buffer for Insurance
No output buffer: stream to output
Disk
Patient
Insurance

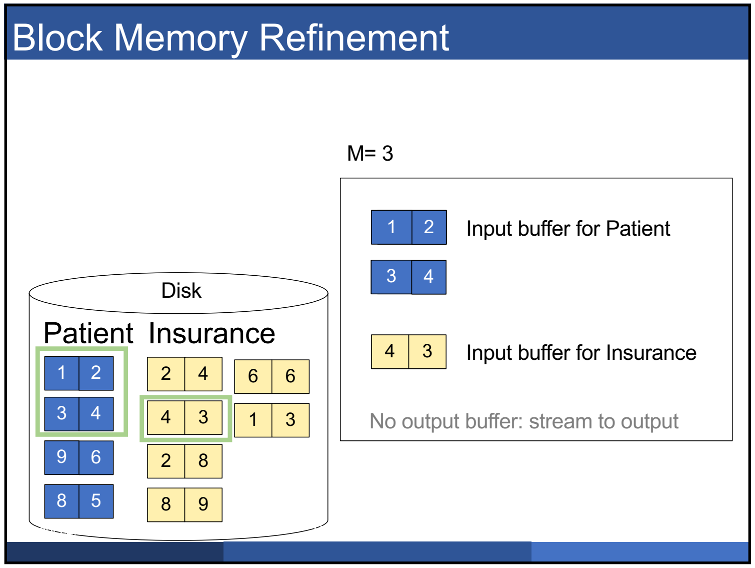
34

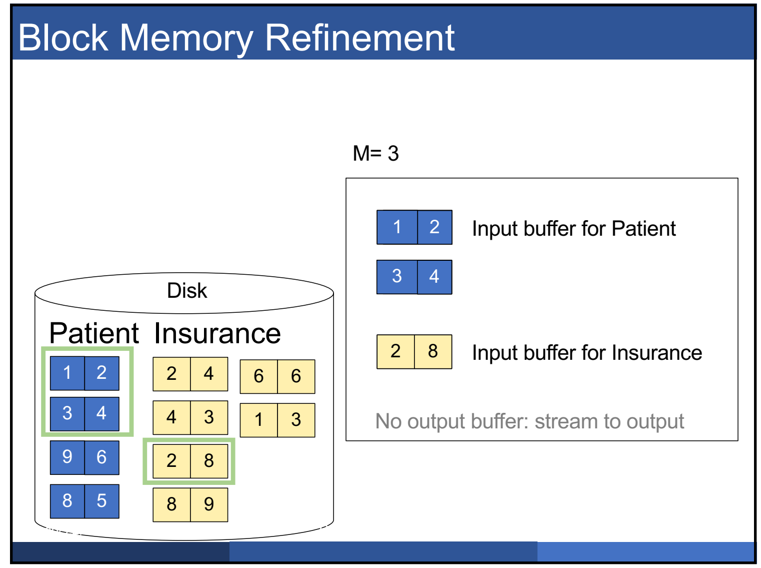

Block Memory Refinement
M= 3
Input buffer for Patient
Input buffer for Insurance
No output buffer: stream to output
Disk
Patient
Insurance

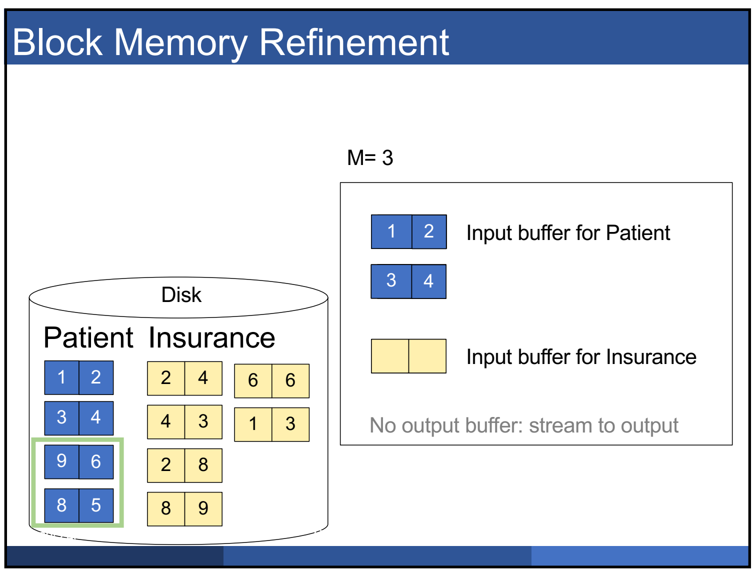
35

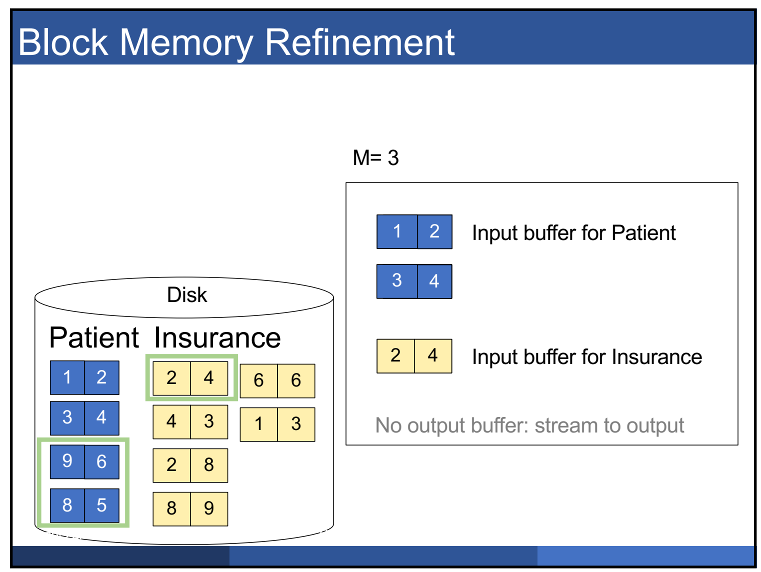

Block Memory Refinement
M= 3
Input buffer for Patient
Input buffer for Insurance
No output buffer: stream to output
Disk
Patient
Insurance

36


Block Memory Refinement
M= 3
1 2
3 4
Input buffer for Patient
4 3
Input buffer for Insurance
No output buffer: stream to output
Disk
Patient
Insurance
1 2
3 4
9 6
8 5
2 4
4 3
2 8
8 9
6 6
1 3
37


Block Memory Refinement
M= 3
1 2
3 4
Input buffer for Patient
2 8
Input buffer for Insurance
No output buffer: stream to output
Disk
Patient
Insurance
1 2
3 4
9 6
8 5
2 4
4 3
2 8
8 9
6 6
1 3
38


Block Memory Refinement
M= 3
1 2
3 4
Input buffer for Patient
Input buffer for Insurance
No output buffer: stream to output
Disk
Patient
Insurance
1 2
3 4
9 6
8 5
2 4
4 3
2 8
8 9
6 6
1 3
39


Block Memory Refinement
M= 3
1 2
3 4
Input buffer for Patient
2 4
Input buffer for Insurance
No output buffer: stream to output
Disk
Patient
Insurance
1 2
3 4
9 6
8 5
2 4
4 3
2 8
8 9
6 6
1 3
40

Block Memory 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 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?

January 22, 2021 CSE 444 - Winter 2020 41

41

Block Memory 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 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)/(M-1)

What is the Cost?

January 22, 2021 CSE 444 - Winter 2020 42

42

Sort-Merge Join

Sort-merge join: R ⋈ S

- Scan R and sort in main memory
- Scan S and sort in main memory
- Merge R and S

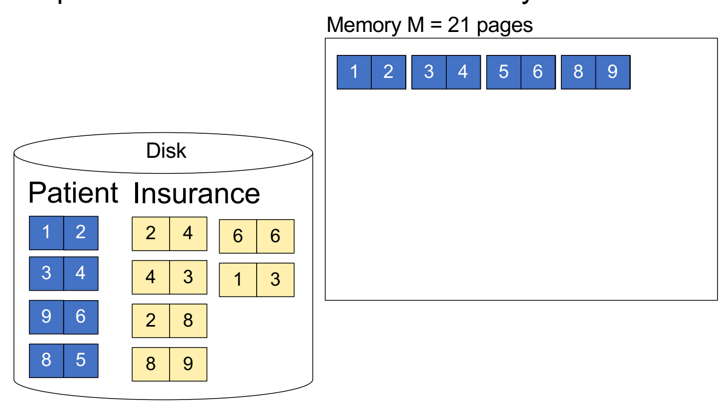
- Cost: B(R) + B(S)
- One pass algorithm when B(S) + B(R) <= M
- Typically, this is NOT a one pass algorithm,
  - We'll see the multi-pass version next lecture

January 22, 2021 CSE 444 - Winter 2020 43

43

Sort-Merge Join Example

Step 1: Scan Patient and sort in memory

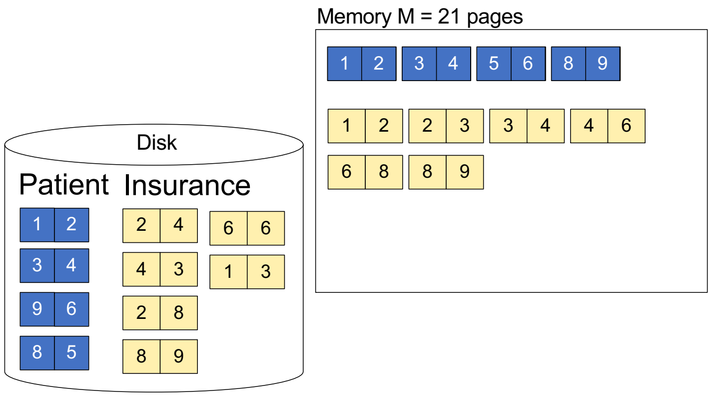


January 22, 2021 CSE 444 - Winter 2020 44

44

Sort-Merge Join Example

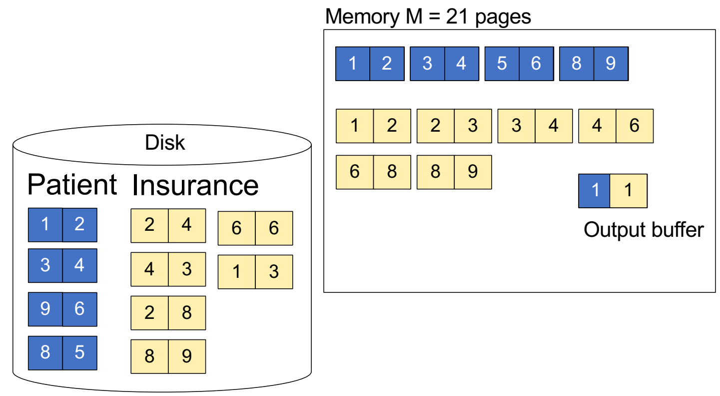
Step 2: Scan Insurance and sort in memory



45

Sort-Merge Join Example

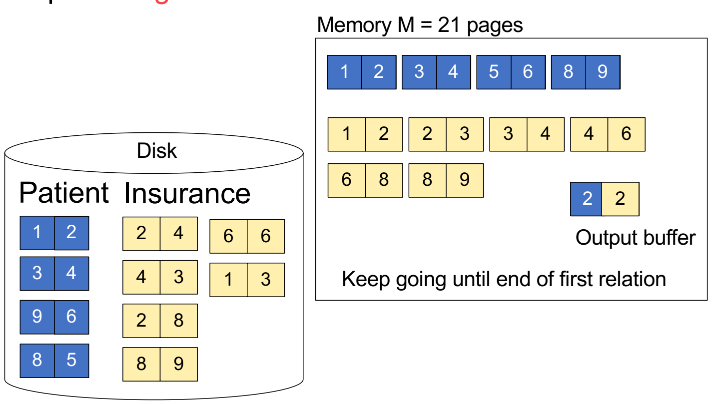
Step 3: Merge Patient and Insurance



46

Sort-Merge Join Example

Step 3: Merge Patient and Insurance



47

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)

48

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

49

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:

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

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:

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:

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:

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:

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

January 22, 2021 CSE 444 - Winter 2020 57

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 !

January 22, 2021 CSE 444 - Winter 2020 58

58

Index Nested Loop Join

R ⋈ 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)

January 22, 2021 CSE 444 - Winter 2020 59

59