CSE 444: Database Internals

Lecture 7
Query Execution and
Operator Algorithms (part 1)

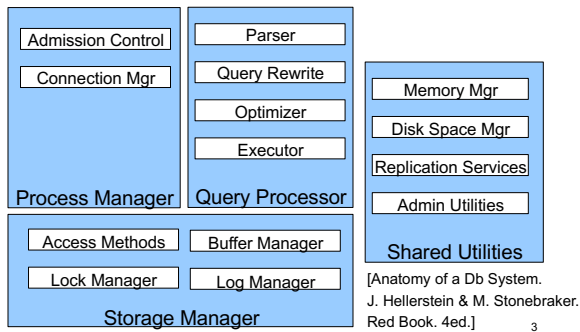CSE 444 - Winter 2017   1

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 - Winter 2017   2

DBMS Architecture

Process Manager	Query Processor	Shared Utilities
Admission Control	Parser	Memory Mgr
Connection Mgr	Query Rewrite	Disk Space Mgr
	Optimizer	Replication Services
	Executor	Admin Utilities

Storage Manager

Access Methods	Buffer Manager
Lock Manager	Log Manager

[Anatomy of a Db System. J. Hellerstein & M. Stonebraker. Red Book. 4ed.]   3

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
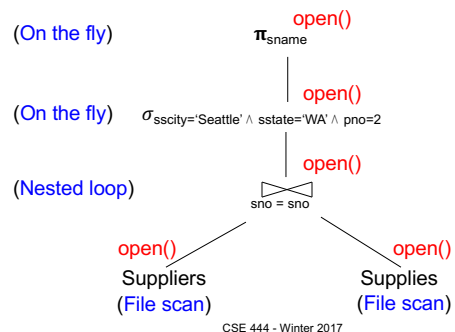 - i.e., Query optimization

CSE 444 - Winter 2017   4

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

- Execution of the physical plan is pull-based

- Operators given a limited amount of memory
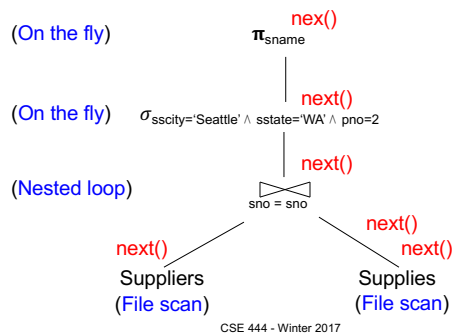
CSE 444 - Winter 2017   5
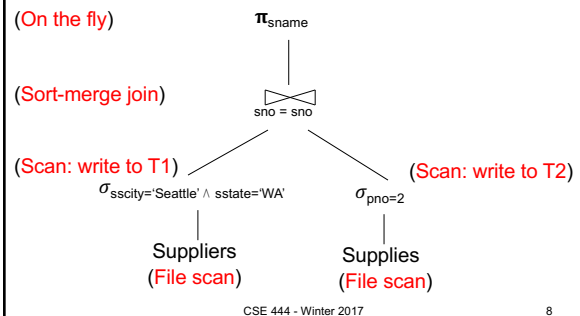

Pipelined Query Execution

(On the fly)   open() π_{sname}

(On the fly)   open() $\sigma_{sscity='Seattle' \wedge sstate='WA' \wedge pno=2}$

(Nested loop)   open()   ⋈ sno = sno

open()   Suppliers (File scan)   open()   Supplies (File scan)

CSE 444 - Winter 2017   6

1

Pipelined Query Execution

(On the fly)   nex()
           π_{sname}

(On the fly)   next()
     $\sigma_{sscity='Seattle' \wedge sstate='WA' \wedge pno=2}$

         next()
(Nested loop)
     sno = sno

            next()
  next()   next()

Suppliers   Supplies
(File scan)   (File scan)

CSE 444 - Winter 2017   7

Intermediate Tuple Materialization

(On the fly) π_{sname}

(Sort-merge join)
     sno = sno

(Scan: write to T1)   (Scan: write to T2)
  $\sigma_{sscity='Seattle' \wedge sstate='WA'}$ $\sigma_{pno=2}$

Suppliers   Supplies
(File scan)   (File scan)

CSE 444 - Winter 2017   8

Memory Management

Each operator:
- Pre-allocates heap space for tuples
  - Pointers to base data in buffer pool
  - Or new tuples on the heap
- Allocates memory for its internal state
  - Either on heap or buffer pool (depends on system)

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

CSE 444 - Winter 2017   9

Operator Algorithms

CSE 444 - Winter 2017   10

Operator Algorithms

Design criteria

- Cost: IO, CPU, Network

- Memory utilization

- Load balance (for parallel operators)

CSE 444 - Winter 2017   11

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 - Winter 2017   12

Convention

- Cost = the cost of reading operands from disk

- Cost of writing the result to disk is *not included*; need to count it separately when applicable

CSE 444 - Winter 2017   13

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: read the book

CSE 444 - Winter 2017   14

Join Algorithms

- Hash join

- Nested loop join

- Sort-merge join

CSE 444 - Winter 2017   15

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

CSE 444 - Winter 2017   16

Hash Join Example

Patient(pid, name, address)

Insurance(pid, provider, policy_nb)

Patient ⋈ Insurance

Two tuples per page

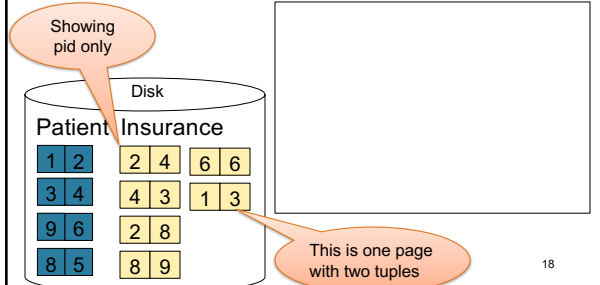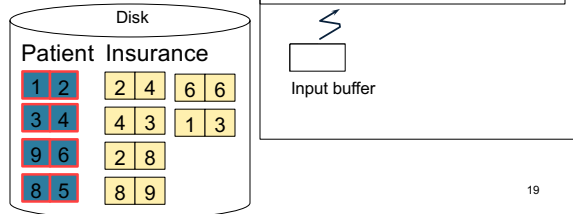| Patient			Insurance		
1	'Bob'	'Seattle'	2	'Blue'	123
2	'Ela'	'Everett'	4	'Prem'	432
3	'Jill'	'Kent'	4	'Prem'	343
4	'Joe'	'Seattle'	3	'GrpH'	554

17

Hash Join Example

Patient ⋈ Insurance

Some large-enough nb

Memory M = 21 pages

Showing pid only

Disk

Patient Insurance

1	2		2	4		6	6
3	4		4	3		1	3
9	6		2	8			
8	5		8	9			

This is one page with two tuples

18

3

Hash Join Example

Step 1: Scan Patient and build hash table in memory

Can be done in method open()

Memory M = 21 pages

Hash h: pid % 5

| 5 | | 1 | 6 | 2 | | 3 | 8 | 4 | 9 |

Input buffer

Disk

Patient Insurance

1	2		2	4	6	6
3	4		4	3	1	3
9	6		2	8
8	5		8	9

19

Hash Join Example

Step 2: Scan Insurance and probe into hash table

Done during calls to next()

Memory M = 21 pages

Hash h: pid % 5

| 5 | | 1 | 6 | 2 | | 3 | 8 | 4 | 9 |

| 2 | 4 | | 2 | 2 |

Input buffer   Output buffer

Write to disk or pass to next operator

Disk

Patient Insurance

1	2		2	4	6	6
3	4		4	3	1	3
9	6		2	8
8	5		8	9

20

Hash Join Example

Step 2: Scan Insurance and probe into hash table

Done during calls to next()

Memory M = 21 pages

Hash h: pid % 5

| 5 | | 1 | 6 | 2 | | 3 | 8 | 4 | 9 |

| 2 | 4 | | 4 | 4 |

Input buffer   Output buffer

Disk

Patient Insurance

1	2		2	4	6	6
3	4		4	3	1	3
9	6		2	8
8	5		8	9

21

Hash Join Example

Step 2: Scan Insurance and probe into hash table

Done during calls to next()

Memory M = 21 pages

Hash h: pid % 5

| 5 | | 1 | 6 | 2 | | 3 | 8 | 4 | 9 |

| 4 | 3 | | 4 | 4 |

Input buffer   Output buffer

Keep going until read all of Insurance

Disk

Patient Insurance

1	2		2	4	6	6
3	4		4	3	1	3
9	6		2	8
8	5		8	9

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

22
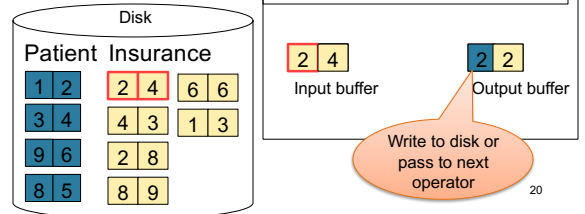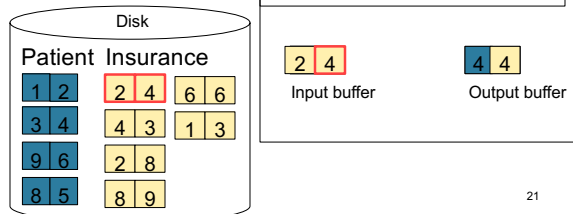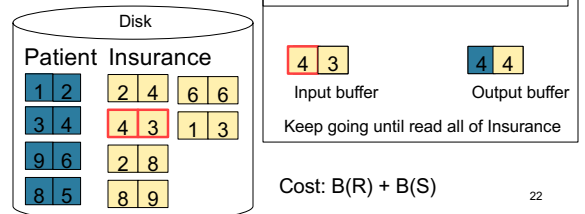
Nested Loop Joins

- Tuple-based nested loop $R \bowtie S$
- 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?

Nested Loop Joins

- Tuple-based nested loop $R \bowtie S$
- 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

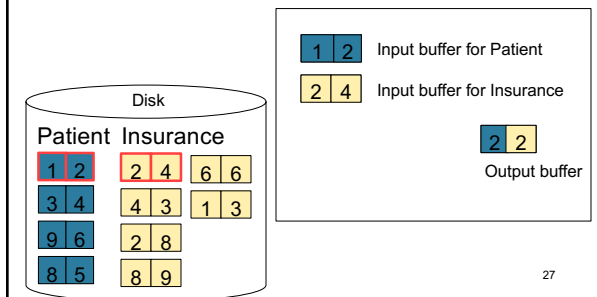## Page-at-a-time Refinement

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
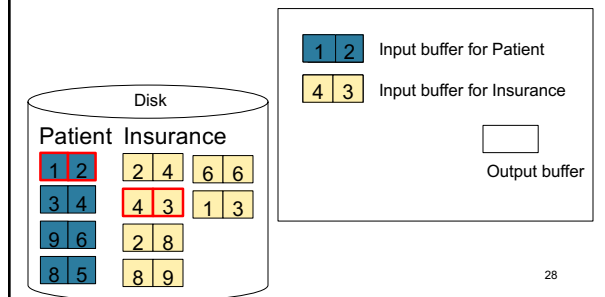   if t_1 and t_2 join then output (t_1, t_2)

What is the Cost?

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?

Page-at-a-time Refinement

27

Page-at-a-time Refinement

28

Page-at-a-time Refinement

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

29
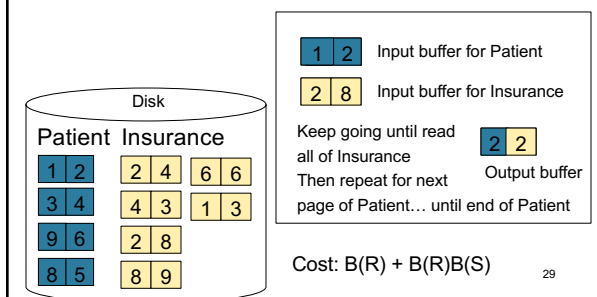
Block-Nested-Loop Refinement

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 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?

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 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)

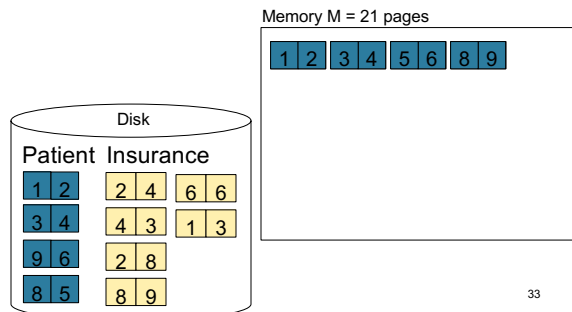What is the Cost?

CSE 444 - Winter 2017   31

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
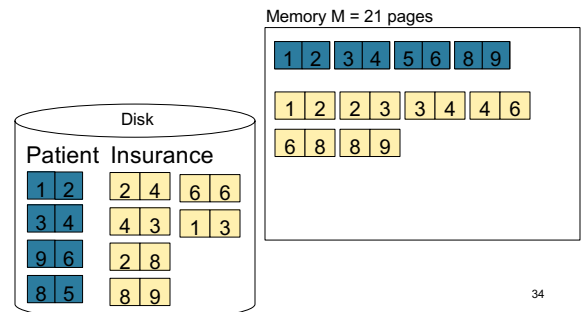- Typically, this is NOT a one pass algorithm

CSE 444 - Winter 2017   32

Sort-Merge Join Example
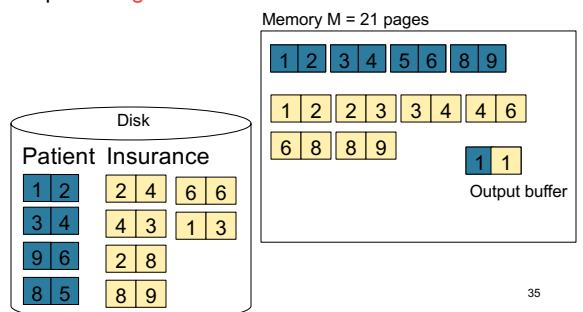
Step 1: Scan Patient and sort in memory

33

Sort-Merge Join Example

Step 2: Scan Insurance and sort in memory

34

Sort-Merge Join Example

Step 3: Merge Patient and Insurance

35

Sort-Merge Join Example

Step 3: Merge Patient and Insurance

Keep going until end of first relation

36