



CS 525: Advanced Database Organization

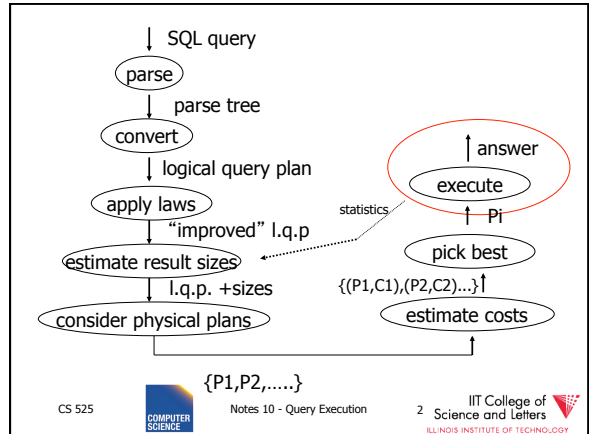
10: Query Execution

Boris Glavic




Slides: adapted from a [course](#) taught by [Hector Garcia-Molina](#), Stanford InfoLab

CS 525

Notes 10 - Query Execution
1 IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY




Query Execution

- Here only:
 - how to implement operators
 - what are the costs of implementations
 - how to implement queries
 - Data flow between operators
- Next part:
 - How to choose good plan

CS 525

Notes 10 - Query Execution
3 IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY


Execution Plan

- A tree (DAG) of physical operators that implement a query
- May use indices
- May create temporary relations
- May create indices on the fly
- May use auxiliary operations such as sorting

CS 525

Notes 10 - Query Execution
4 IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY


How to estimate costs

- If everything fits into memory
 - Standard computational complexity
- If not
 - Assume fixed memory available for buffering pages
 - Count I/O operations
 - Real systems combine this with CPU estimations

CS 525

Notes 10 - Query Execution
5 IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Estimating IOs:

- Count # of disk blocks that must be read (or written) to execute query plan

CS 525

Notes 10 - Query Execution
6 IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

To estimate costs, we may have additional parameters:

$B(R)$ = # of blocks containing R tuples

$f(R)$ = max # of tuples of R per block

M = # memory blocks available

CS 525



Notes 10 - Query Execution

7

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

To estimate costs, we may have additional parameters:

$B(R)$ = # of blocks containing R tuples

$f(R)$ = max # of tuples of R per block

M = # memory blocks available

$HT(i)$ = # levels in index i

$LB(i)$ = # of leaf blocks in index i

CS 525



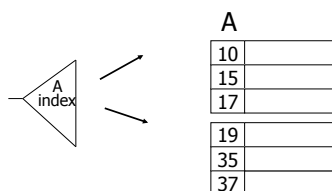
Notes 10 - Query Execution

8

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Clustered index

Index that allows tuples to be read in an order that corresponds to physical order



CS 525



Notes 10 - Query Execution

9

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Operators Overview

- (External) Sorting
- Joins (Nested Loop, Merge, Hash, ...)
- Aggregation (Sorting, Hash)
- Selection, Projection (Index, Scan)
- Union, Set Difference
- Intersection
- Duplicate Elimination

CS 525



Notes 10 - Query Execution

10

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Operator Profiles

- Algorithm
- In-memory complexity: e.g., $O(n^2)$
- Memory requirements
 - Runtime based on available memory
- #I/O if operation needs to go to disk
- Disk space needed
- Prerequisites
 - Conditions under which the operator can be applied

CS 525



Notes 10 - Query Execution

11

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Execution Strategies

- Compiled
 - Translate into C/C++/Assembler code
 - Compile, link, and execute code
- Interpreted
 - Generic operator implementations
 - Generic executor
 - Interprets query plan

CS 525



Notes 10 - Query Execution

12

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Virtual Machine Approach

- Implement virtual machine of low-level DBMS operations
- Compile query into machine-code for that machine

CS 525



Notes 10 - Query Execution

13

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Iterator Model

- Need to be able to combine operators in different ways
 - E.g., join inputs may be scans, or outputs of other joins, ...
 - -> define generic interface for operators
 - be able to arbitrarily compose complex plans from a small set of operators

CS 525



Notes 10 - Query Execution

14

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Iterator Model - Interface

- **Open**
 - Prepare operator to read inputs
- **Close**
 - Close operator and clean up
- **Next**
 - Return next result tuple

CS 525

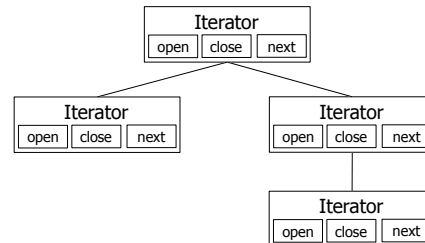


Notes 10 - Query Execution

15

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Query Execution – Iterator Model



CS 525

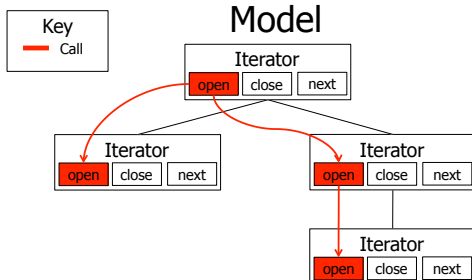


Notes 10 - Query Execution

16

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Query Execution – Iterator Model



CS 525

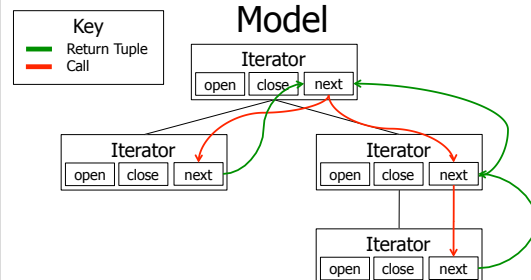


Notes 10 - Query Execution

17

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Query Execution – Iterator Model



CS 525



Notes 10 - Query Execution

18

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Parallelism

- Iterator Model
 - **Pull-based** query execution
- Potential types of parallelism
 - Inter-query (every multiuser system)
 - Intra-operator
 - Inter-operator

CS 525



Notes 10 - Query Execution

19

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Intra-Operator Parallelism

- Execute portions of an operator in parallel
 - Merge-Sort
 - Assign a processor to each merge phase
 - Scan
 - Partition tables
 - Each process scans one partition

CS 525



Notes 10 - Query Execution

20

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Inter-Operator Parallelism

- Each process executes one or more operators
- **Pipelining**
 - **Push-based** query execution
 - Chain operators to directly produce results
 - Pipeline-breakers
 - Operators that need to consume the whole input (or large parts) before producing outputs

CS 525



Notes 10 - Query Execution

21

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Pipelining Communication

- Queues
 - Operators push their results to queues
 - Operators read their inputs from queues
- Direct call
 - Operator calls its parent in the tree with results
 - Within one process

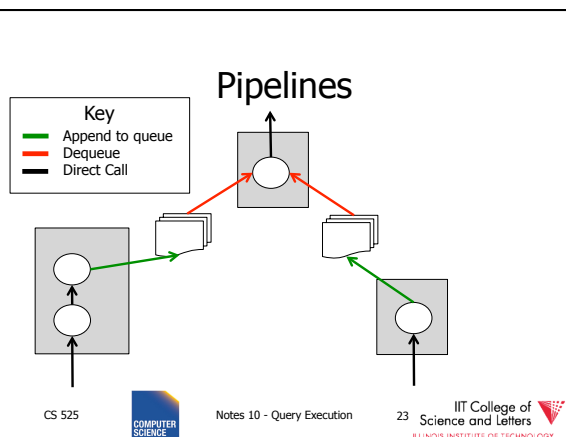
CS 525



Notes 10 - Query Execution

22

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY



Pipeline-breakers

- Sorting
 - All operators that apply sorting
- Aggregation
- Set Difference
- Some implementations of
 - Join
 - Union

CS 525



Notes 10 - Query Execution

24

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Operators Overview

- (External) Sorting
- Joins (Nested Loop, Merge, Hash, ...)
- Aggregation (Sorting, Hash)
- Selection, Projection (Index, Scan)
- Union, Set Difference
- Intersection
- Duplicate Elimination

CS 525



Notes 10 - Query Execution

25

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Sorting

- Why do we want/need to sort
 - Query requires sorting (ORDER BY)
 - Operators require sorted input
 - Merge-join
 - Aggregation by sorting
 - Duplicate removal using sorting

CS 525



Notes 10 - Query Execution

26

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

In-memory sorting

- Algorithms from data structures 101
 - Quick sort
 - Merge sort
 - Heap sort
 - Intro sort
 - ...

CS 525



Notes 10 - Query Execution

27

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

External sorting

- Problem:
 - Sort **N** pages of data with **M** pages of memory
- Solutions?

CS 525



Notes 10 - Query Execution

28

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

First Idea

- Split data into runs of size **M**
- Sort each run in memory and write back to disk
 - $\lceil N/M \rceil$ sorted runs of size **M**
- Now what?



CS 525



Notes 10 - Query Execution

29

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Merging Runs

- Need to create bigger sorted runs out of sorted smaller runs
 - Divide and Conquer
 - Merge Sort?
- How to merge two runs that are bigger than **M**?

CS 525



Notes 10 - Query Execution

30

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Merging Runs using 3 pages

- Merging sorted runs R_1 and R_2
- Need 3 pages
 - One page to buffer pages from R_1
 - One page to buffer pages from R_2
 - One page to buffer the result
 - Whenever this buffer is full, write it to disk

CS 525

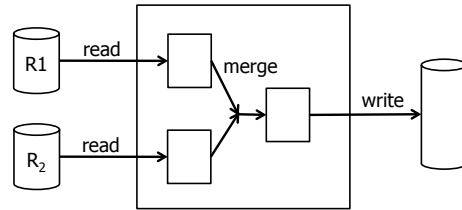


Notes 10 - Query Execution

31

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Merging Runs



CS 525



Notes 10 - Query Execution

32

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

2-Way External Mergesort

- Repeat process until we have one sorted run
- Each iteration (pass) reads and writes the whole table once: $2 B(R)$ I/Os
- Each pass doubles the run size
 - $1 + \lceil \log_2 (B(R) / M) \rceil$ runs
 - $2 B(R) * (1 + \lceil \log_2 (B(R) / M) \rceil)$ I/Os

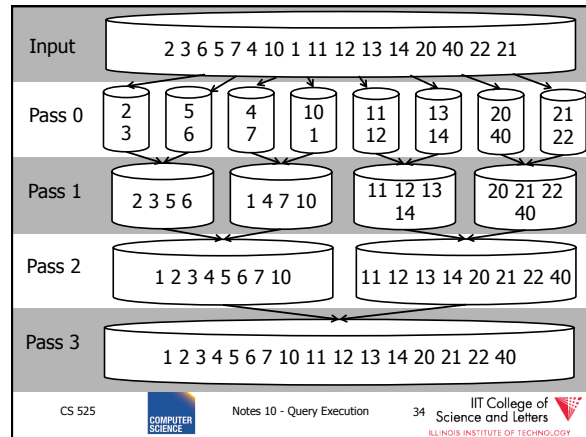
CS 525



Notes 10 - Query Execution

33

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY



CS 525



Notes 10 - Query Execution

34

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

N-Way External Mergesort

- How to utilize M buffer during merging?
- Each pass merges $M-1$ runs at once
 - One memory page as buffer for each run
- #I/Os
 - $1 + \lceil \log_{M-1} (B(R) / M) \rceil$ runs
 - $2 B(R) * (1 + \lceil \log_{M-1} (B(R) / M) \rceil)$ I/Os

CS 525

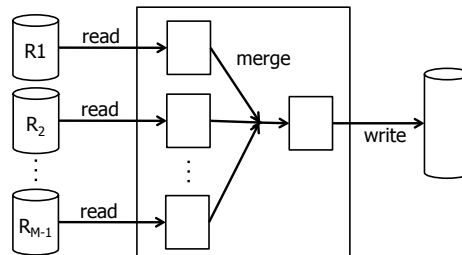


Notes 10 - Query Execution

35

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Merging Runs



CS 525



Notes 10 - Query Execution

36

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

How many passes do we need?

N	M=17	M=129	M=257	M=513	M=1025
100	2	1	1	1	1
1,000	3	2	2	2	1
10,000	4	2	2	2	2
100,000	5	3	3	2	2
1,000,000	5	3	3	3	2
10,000,000	6	4	3	3	3
100,000,000	7	4	4	3	3
1,000,000,000	8	5	4	4	3

CS 525



Notes 10 - Query Execution

37

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

To put into perspective

- Scenario
 - Page size 4KB
 - 1TB of data (250,000,000)
 - 10MB of buffer for sorting (250)
- Passes
 - 4 passes

CS 525



Notes 10 - Query Execution

38

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Merge

- In practice would want larger I/O buffer for each run
- Trade-off between number of runs and efficiency of I/O

CS 525



Notes 10 - Query Execution

39

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Improving in-memory merging

- Merging **M** runs
 - To choose next element to output
 - Have to compare **M** elements
 - \rightarrow complexity linear in **M**: **O(M)**
- How to improve that?
 - Use priority queue to store current element from each run
 - \rightarrow **O(log₂(M))**

CS 525



Notes 10 - Query Execution

40

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Priority Queue

- Queue for accessing elements in some given order
 - **pop-smallest** = return and remove smallest element in set
 - **Insert(e)** = insert element into queue

CS 525



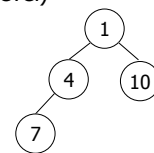
Notes 10 - Query Execution

41

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Min-Heap

- Implementation of priority queue
 - Store elements in a binary tree
 - All levels are full (except leaf level)
 - Heap property
 - Parent is smaller than child
- Example: { 1, 4, 7, 10 }



CS 525



Notes 10 - Query Execution

42

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Min-Heap Insertion

- **insert(e)**

1. Add element at next free leaf node
 - This may invalidate heap property
2. If node smaller than parent then
 - Switch node with parent
3. Repeat until 2) cannot be applied anymore

CS 525



Notes 10 - Query Execution

43

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Min-Heap Dequeue

- **pop-smallest**

1. Return Root and use right-most leaf as new root
 - This may invalidate heap property
2. If node smaller than child then
 - Switch node with smaller child
3. Repeat until 2) cannot be applied anymore

CS 525



Notes 10 - Query Execution

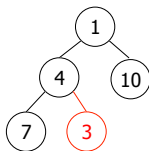
44

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

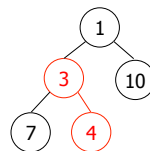
Insertion

- Insert 3

Insert at first free position



Restore heap property



CS 525

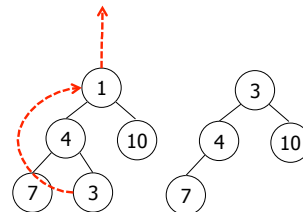


Notes 10 - Query Execution

45

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Dequeue



CS 525

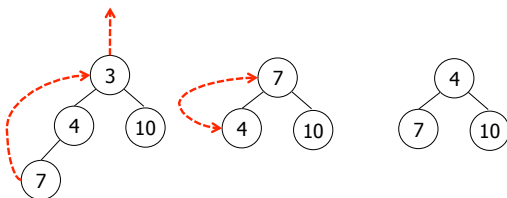


Notes 10 - Query Execution

46

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Dequeue



CS 525



Notes 10 - Query Execution

47

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Min/Max-Heap Complexity

- Heap is a complete tree
 - Height is $O(\log_2(n))$
- Insertion
 - Maximal height of the tree switches
 - $\rightarrow O(\log_2(n))$
- Dequeue
 - Maximal height of the tree switches
 - $\rightarrow O(\log_2(n))$

CS 525



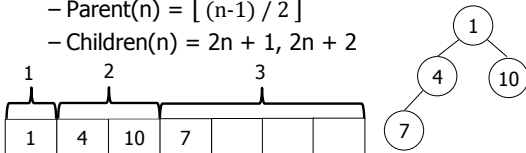
Notes 10 - Query Execution

48

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Min-Heap Implementation

- Full tree
 - Use array to implement tree
- Compute positions
 - $\text{Parent}(n) = \lfloor (n-1) / 2 \rfloor$
 - $\text{Children}(n) = 2n + 1, 2n + 2$



CS 525

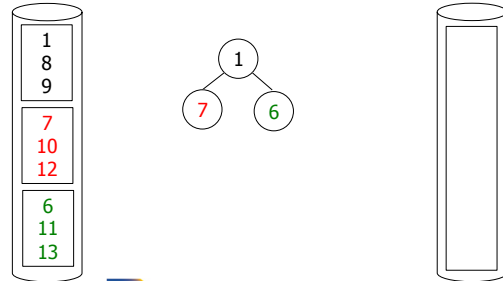


Notes 10 - Query Execution

49

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Merging with Priority Queue



CS 525

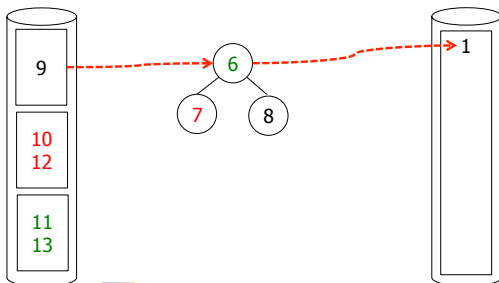


Notes 10 - Query Execution

50

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Merging with Priority Queue



CS 525

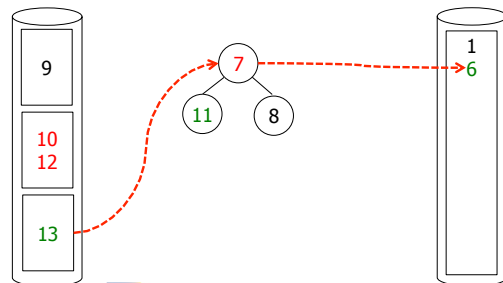


Notes 10 - Query Execution

51

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Merging with Priority Queue



CS 525



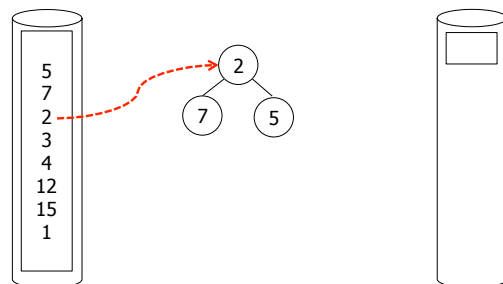
Notes 10 - Query Execution

52

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Using a heap to generate runs

- Read inputs into heap
 - Until available memory is full
- Replace elements
 - Remove smallest element from heap
 - If larger than last element written of current run then write to current run
 - Else create a new run
 - Add new element from input to heap



CS 525



Notes 10 - Query Execution

53

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

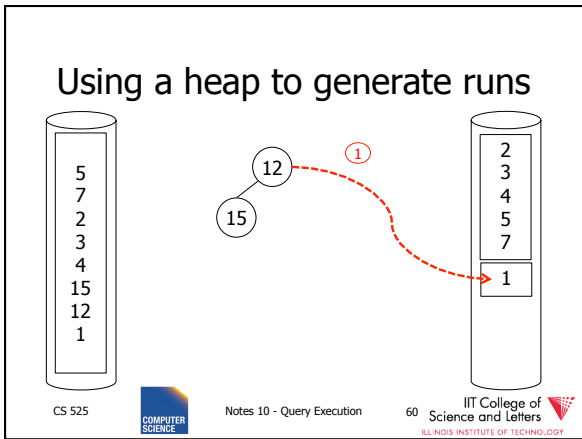
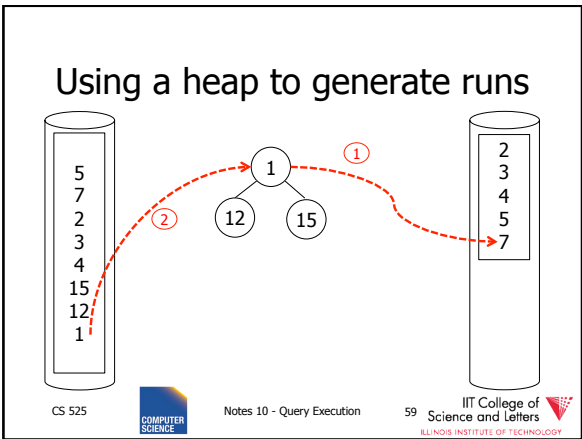
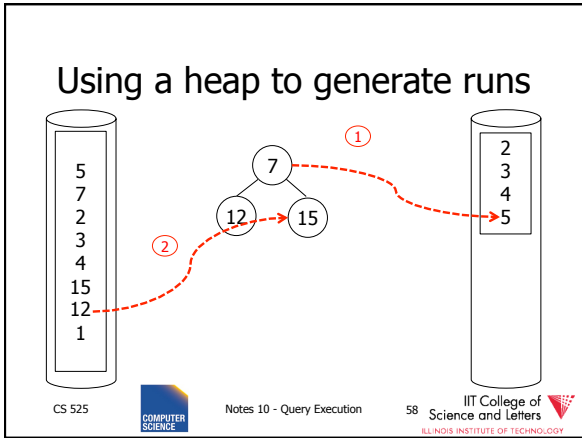
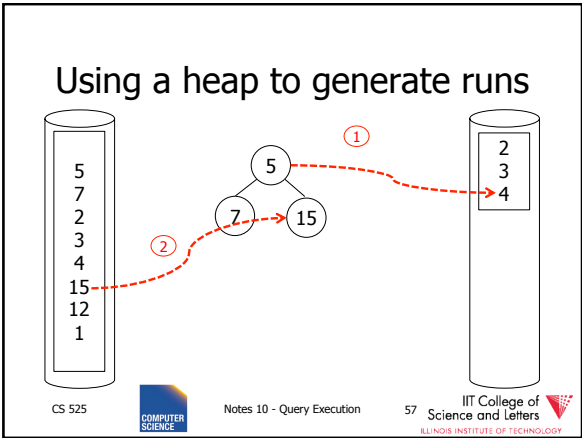
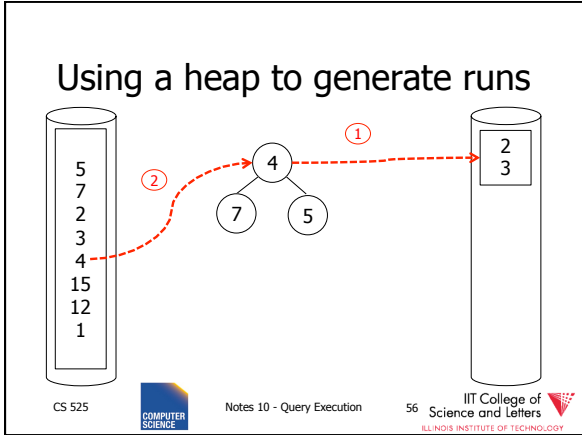
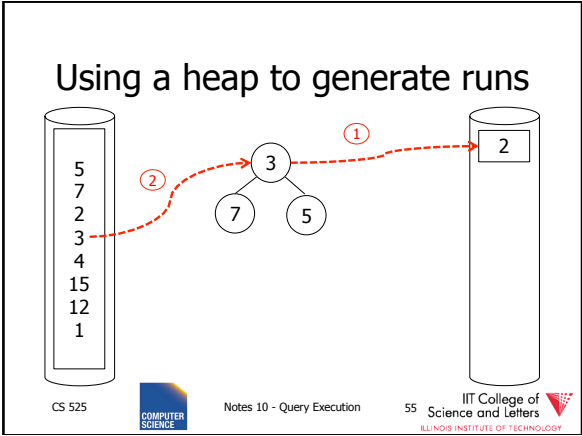
CS 525



Notes 10 - Query Execution

54

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY



Using a heap to generate runs

- Increases the run-length
 - On average by a factor of 2 (see Knuth)

CS 525



Notes 10 - Query Execution

61

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Use clustered B+-tree

- Keys in the B+-tree **I** are in sort order
 - If B+-tree is clustered traversing the leaf nodes is sequential I/O!
 - \mathbf{K} = #keys/leaf node
- Approach
 - Traverse from root to first leaf: **HT(I)**
 - Follow sibling pointers: $\mathbf{|R| / K}$
 - Read data blocks: **B(R)**

CS 525



Notes 10 - Query Execution

62

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

I/O Operations

- HT(I) + |R| / K + B(R)** I/Os
- Less than **2 B(R)** = 1 pass external mergesort
- > Better than external merge-sort!

CS 525



Notes 10 - Query Execution

63

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Unclustered B+-tree?

- Each entry in a leaf node may point to different page of relation R
 - For each leaf page we may read up to **K** pages from relation R
 - Random I/O
- In worst-case we have
 - $\mathbf{K * B(R)}$
 - $\mathbf{K = 500}$
 - $\mathbf{500 * B(R)}$ = 250 merge passes

CS 525



Notes 10 - Query Execution

64

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Sorting Comparison

B(R) = number of block of R
M = number of available memory blocks
#RB = records per page
HT = height of B+-tree (logarithmic)
K = number of keys per leaf node

Property	Ext. Mergesort	B+ (clustered)	B+ (unclustered)
Runtime	$O(N \log_{M-1}(N))$	$O(N)$	$O(N)$
#I/O (random)	$2 B(R) * (1 + \lceil \log_{M-1}(B(R)/M) \rceil)$	$HT + R / K + B(R)$	$HT + R / K + K * \#RB$
Memory	M	1 (better HT + X)	1 (better HT + X)
Disk Space	$2 B(R)$	0	0
Variants	1) Merge with heap 2) Run generation with heap 3) Larger Buffer		

CS 525



Notes 10 - Query Execution

65

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Operators Overview

- (External) Sorting
- Joins (Nested Loop, Merge, Hash, ...)
- Aggregation (Sorting, Hash)
- Selection, Projection (Index, Scan)**
- Union, Set Difference
- Intersection
- Duplicate Elimination

CS 525



Notes 10 - Query Execution

66

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Scan

- Implements access to a table
 - Combined with selection
 - Probably projection too
- Variants
 - **Sequential**
 - Scan through all tuples of relation
 - **Index**
 - Use index to find tuples that match selection

CS 525



Notes 10 - Query Execution

67

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Operators Overview

- (External) Sorting
- Joins (Nested Loop, Merge, Hash, ...)
- Aggregation (Sorting, Hash)
- Selection, Projection (Index, Scan)
- Union, Set Difference
- Intersection
- Duplicate Elimination

CS 525



Notes 10 - Query Execution

68

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Options

- Transformations: $R_1 \bowtie_c R_2, R_2 \bowtie_c R_1$
- Joint algorithms:
 - Nested loop
 - Merge join
 - Join with index
 - Hash join
- Outer join algorithms

CS 525



Notes 10 - Query Execution

69

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Nested Loop Join (conceptually)

```
for each r ∈ R1 do
  for each s ∈ R2 do
    if (r,s) ⊨ C then output (r,s)
```

Applicable to:

- Any join condition C
- Cross-product

CS 525



Notes 10 - Query Execution

70

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Merge Join (conceptually)

- (1) if R_1 and R_2 not sorted, sort them
- (2) $i \leftarrow 1; j \leftarrow 1;$
While $(i \leq T(R_1)) \wedge (j \leq T(R_2))$ do
 if $R_1\{i\}.C = R_2\{j\}.C$ then output Tuples
 else if $R_1\{i\}.C > R_2\{j\}.C$ then $j \leftarrow j+1$
 else if $R_1\{i\}.C < R_2\{j\}.C$ then $i \leftarrow i+1$

Applicable to:

- C is conjunction of equalities or $</>$:
 $A_1 = B_1$ AND ... AND $A_n = B_n$

CS 525



Notes 10 - Query Execution

71

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Procedure Output-Tuples

```
While  $(R_1\{i\}.C = R_2\{j\}.C) \wedge (i \leq T(R_1))$  do
  [  $jj \leftarrow j;$ 
    while  $(R_1\{i\}.C = R_2\{jj\}.C) \wedge (jj \leq T(R_2))$  do
      [output pair  $R_1\{i\}, R_2\{jj\};$ 
         $jj \leftarrow jj+1$  ]
    ]
   $i \leftarrow i+1$  ]
```

CS 525



Notes 10 - Query Execution

72

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Example

i	$R_1\{i\}.C$	$R_2\{j\}.C$	j
1	10	5	1
2	20	20	2
3	20	20	3
4	30	30	4
5	40	30	5
		50	6
		52	7

CS 525



Notes 10 - Query Execution

73

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Index nested loop (Conceptually)

```

For each r ∈ R1 do
    [ X ← index (R2, C, r.C)
    for each s ∈ X do
        output (r,s) pair]
    
```

Note: $X \leftarrow \text{index}(\text{rel}, \text{attr}, \text{value})$
then $X = \text{set of rel tuples with attr} = \text{value}$

CS 525



Notes 10 - Query Execution

74

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Hash join (conceptual)

Hash function h , range $0 \rightarrow k$
Buckets for R_1 : G_0, G_1, \dots, G_k
Buckets for R_2 : H_0, H_1, \dots, H_k

Applicable to:

- C is conjunction of equalities
 $A_1 = B_1 \text{ AND } \dots \text{ AND } A_n = B_n$

CS 525



Notes 10 - Query Execution

75

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Hash join (conceptual)

Hash function h , range $0 \rightarrow k$
Buckets for R_1 : G_0, G_1, \dots, G_k
Buckets for R_2 : H_0, H_1, \dots, H_k

Algorithm

- (1) Hash R_1 tuples into G buckets
- (2) Hash R_2 tuples into H buckets
- (3) For $i = 0$ to k do
match tuples in G_i, H_i buckets

CS 525



Notes 10 - Query Execution

76

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Simple example hash: even/odd

R_1	R_2	Buckets	
2	5	Even: 2 4 8	4 12 8 14
4	4		R_1 R_2
3	12	Odd: 3 5 9	5 3 13 11
5	3		
8	13		
9	8		
	11		
	14		

CS 525



Notes 10 - Query Execution

77

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Factors that affect performance

- (1) Tuples of relation stored physically together?
- (2) Relations sorted by join attribute?
- (3) Indexes exist?

CS 525



Notes 10 - Query Execution

78

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Example 1(a) NL Join $R_1 \bowtie R_2$

- Relations not contiguous
- Recall $\begin{cases} T(R_1) = 10,000 & T(R_2) = 5,000 \\ S(R_1) = S(R_2) = 1/10 \text{ block} \\ \text{MEM} = 101 \text{ blocks} \end{cases}$

CS 525



Notes 10 - Query Execution

79

Example 1(a)

Nested Loop Join $R_1 \bowtie R_2$

- Relations not contiguous
- Recall $\begin{cases} T(R_1) = 10,000 & T(R_2) = 5,000 \\ S(R_1) = S(R_2) = 1/10 \text{ block} \\ \text{MEM} = 101 \text{ blocks} \end{cases}$

Cost: for each R_1 tuple:

[Read tuple + Read R_2]

Total = $10,000 [1 + 500] = 5,010,000$ IOs

CS 525



Notes 10 - Query Execution

80

- Can we do better?

CS 525



Notes 10 - Query Execution

81

- Can we do better?

Use our memory

- (1) Read 100 blocks of R_1
- (2) Read all of R_2 (using 1 block) + join
- (3) Repeat until done

CS 525



Notes 10 - Query Execution

82

Cost: for each R_1 chunk:

Read chunk: 100 IOs
Read R_2 : $\frac{500 \text{ IOs}}{600}$

CS 525



Notes 10 - Query Execution

83

Cost: for each R_1 chunk:

Read chunk: 100 IOs
Read R_2 : $\frac{500 \text{ IOs}}{600}$

Total = $\frac{1,000}{100} \times 600 = 6,000$ IOs

CS 525



Notes 10 - Query Execution

84

- Can we do better?

CS 525



Notes 10 - Query Execution

85

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

- Can we do better?

• Reverse join order: $R_2 \bowtie R_1$

$$\text{Total} = \frac{500}{100} \times (100 + 1,000) =$$

$$5 \times 1,100 = 5,500 \text{ IOs}$$

CS 525



Notes 10 - Query Execution

86

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Cost of Block Nested Loop

• Reverse join order: $R_1 \bowtie R_2$

$$\text{Total} = \left\lceil \frac{B(R_1)}{M-1} \right\rceil \times (\min(B(R_1), M-1) + B(R_2))$$

CS 525



Notes 10 - Query Execution

87

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Block-Nested Loop Join (conceptual)

for each M-1 blocks of R_1 do
 read M-1 blocks of R_1 into buffer
 for each block of R_2 do
 read next block of R_2
 for each tuple r in R_1 block
 for each tuple s in R_2 block
 if $(r,s) \in C$ then output (r,s)

CS 525



Notes 10 - Query Execution

88

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Note

- How much memory for buffering inner and for outer chunks?
 - 1 for inner would minimize I/O
 - But, larger buffer better for I/O

CS 525

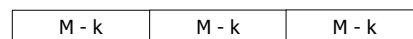


Notes 10 - Query Execution

89

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

R_1



R_2



CS 525



Notes 10 - Query Execution

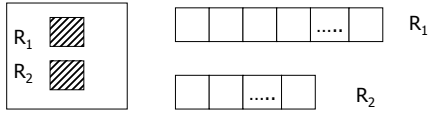
90

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Example 1(b) Merge Join

- Both R_1, R_2 ordered by C; relations contiguous

Memory



CS 525



Notes 10 - Query Execution

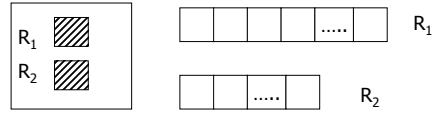
91

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Example 1(b) Merge Join

- Both R_1, R_2 ordered by C; relations contiguous

Memory



Total cost: Read R_1 cost + read R_2 cost
= 1000 + 500 = 1,500 IOs

CS 525



Notes 10 - Query Execution

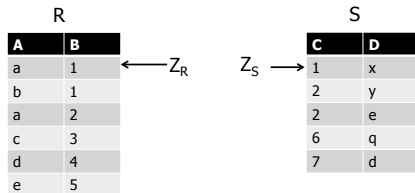
92

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Merge Join Example

$R \bowtie_{B=C} S$

Output: (a,1,1,X)



CS 525



Notes 10 - Query Execution

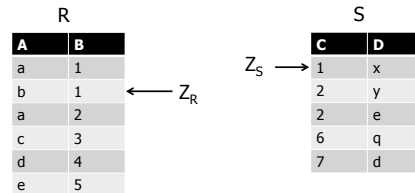
93

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Merge Join Example

$R \bowtie_{B=C} S$

Output: (b,1,1,X)



CS 525



Notes 10 - Query Execution

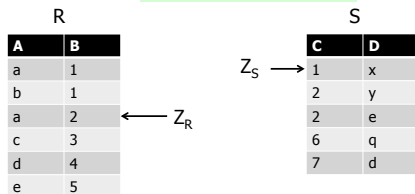
94

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Merge Join Example

$R \bowtie_{B=C} S$

R.B > S.C: advance Z_S



CS 525



Notes 10 - Query Execution

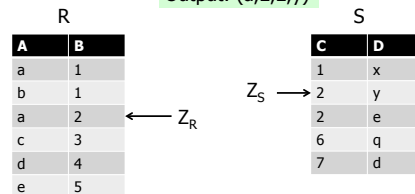
95

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Merge Join Example

$R \bowtie_{B=C} S$

Output: (a,2,2,y)



CS 525



Notes 10 - Query Execution

96

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY


Merge Join Example

$R \bowtie_{B=C} S$

Output: (a,2,2,e)

R		S	
A	B	C	D
a	1	1	x
b	1	2	y
a	2	2	e
c	3	6	q
d	4	7	d
e	5		

Diagram showing pointers Z_R and Z_S at row 2 of R and S respectively.

CS 525  Notes 10 - Query Execution 97 IIT College of Science and Letters ILLINOIS INSTITUTE OF TECHNOLOGY


Merge Join Example

$R \bowtie_{B=C} S$

R.B > S.C: advance Z_S

R		S	
A	B	C	D
a	1	1	x
b	1	2	y
a	2	2	e
c	3	6	q
d	4	7	d
e	5		

Diagram showing pointer Z_S advanced to row 3 of S, and Z_R at row 2 of R.

CS 525  Notes 10 - Query Execution 98 IIT College of Science and Letters ILLINOIS INSTITUTE OF TECHNOLOGY


Merge Join Example

$R \bowtie_{B=C} S$

R.B < S.C: advance Z_R

R		S	
A	B	C	D
a	1	1	x
b	1	2	y
a	2	2	e
c	3	6	q
d	4	7	d
e	5		

Diagram showing pointer Z_R advanced to row 3 of R, and Z_S at row 2 of S.

CS 525  Notes 10 - Query Execution 99 IIT College of Science and Letters ILLINOIS INSTITUTE OF TECHNOLOGY


Merge Join Example

$R \bowtie_{B=C} S$

R.B < S.C: advance Z_R

R		S	
A	B	C	D
a	1	1	x
b	1	2	y
a	2	2	e
c	3	6	q
d	4	7	d
e	5		

Diagram showing pointer Z_R advanced to row 4 of R, and Z_S at row 3 of S.

CS 525  Notes 10 - Query Execution 100 IIT College of Science and Letters ILLINOIS INSTITUTE OF TECHNOLOGY


Merge Join Example

$R \bowtie_{B=C} S$

R.B < S.C: **DONE**

R		S	
A	B	C	D
a	1	1	x
b	1	2	y
a	2	2	e
c	3	6	q
d	4	7	d
e	5		


Diagram showing pointer Z_R at row 4 of R, and Z_S at row 3 of S.

CS 525  Notes 10 - Query Execution 101 IIT College of Science and Letters ILLINOIS INSTITUTE OF TECHNOLOGY

Example 1(c) Merge Join

- R_1, R_2 not ordered, but contiguous

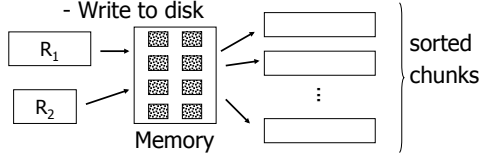
--> Need to sort R_1, R_2 first

CS 525  Notes 10 - Query Execution 102 IIT College of Science and Letters ILLINOIS INSTITUTE OF TECHNOLOGY

One way to sort: Merge Sort

(i) For each 100 blk chunk of R:

- Read chunk
- Sort in memory
- Write to disk



CS 525

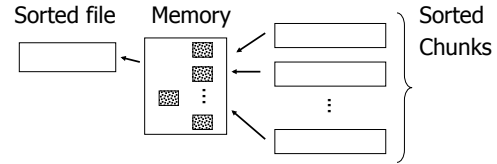


Notes 10 - Query Execution

103

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

(ii) Read all chunks + merge + write out



CS 525



Notes 10 - Query Execution

104

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Cost: Sort

Each tuple is read, written,
read, written

so...

Sort cost R_1 : $4 \times 1,000 = 4,000$

Sort cost R_2 : $4 \times 500 = 2,000$

CS 525



Notes 10 - Query Execution

105

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Example 1(d) Merge Join (continued)

R_1, R_2 contiguous, but unordered

Total cost = sort cost + join cost
= $6,000 + 1,500 = 7,500$ IOs

CS 525



Notes 10 - Query Execution

106

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Example 1(c) Merge Join (continued)

R_1, R_2 contiguous, but unordered

Total cost = sort cost + join cost
= $6,000 + 1,500 = 7,500$ IOs

But: Iteration cost = 5,500
so merge join does not pay off!

CS 525



Notes 10 - Query Execution

107

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

But say $R_1 = 10,000$ blocks contiguous
 $R_2 = 5,000$ blocks not ordered

Iterate: $\frac{5000}{100} \times (100 + 10,000) = 50 \times 10,100$
= 505,000 IOs

Merge join: $5(10,000 + 5,000) = 75,000$ IOs

Merge Join (with sort) WINS!

CS 525



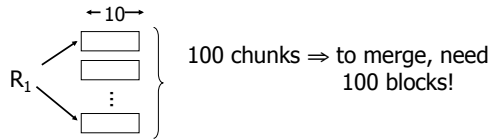
Notes 10 - Query Execution

108

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

How much memory do we need for merge sort?

E.g: Say I have 10 memory blocks



CS 525



Notes 10 - Query Execution

109

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

In general:

Say k blocks in memory

x blocks for relation sort

chunks = (x/k) size of chunk = k

CS 525



Notes 10 - Query Execution

110

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

In general:

Say k blocks in memory

x blocks for relation sort

chunks = (x/k) size of chunk = k

chunks < buffers available for merge

CS 525



Notes 10 - Query Execution

111

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

In general:

Say k blocks in memory

x blocks for relation sort

chunks = (x/k) size of chunk = k

chunks < buffers available for merge

so... $(x/k) \leq k$

or $k^2 \geq x$ or $k \geq \sqrt{x}$

CS 525



Notes 10 - Query Execution

112

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

In our example

R_1 is 1000 blocks, $k \geq 31.62$

R_2 is 500 blocks, $k \geq 22.36$

Need at least 32 buffers

Again: in practice we would not want to use only one buffer per run!

CS 525



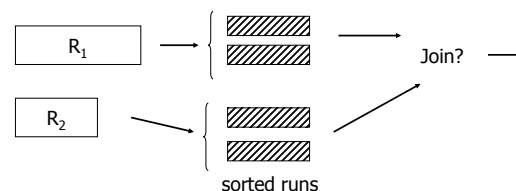
Notes 10 - Query Execution

113

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Can we improve on merge join?

Hint: do we really need the fully sorted files?



CS 525



Notes 10 - Query Execution

114

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Cost of improved merge join:

$C = \text{Read } R_1 + \text{write } R_1 \text{ into runs}$
 $+ \text{read } R_2 + \text{write } R_2 \text{ into runs}$
 $+ \text{join}$
 $= 2,000 + 1,000 + 1,500 = 4,500$

--> Memory requirement?

CS 525



Notes 10 - Query Execution

115

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Example 1(d) Index Join

- Assume $R_1.C$ index exists; 2 levels
- Assume R_2 contiguous, unordered
- Assume $R_1.C$ index fits in memory

CS 525



Notes 10 - Query Execution

116

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Cost: Reads: 500 IOs
for each R_2 tuple:

- probe index - free
- if match, read R_1 tuple: 1 IO

CS 525



Notes 10 - Query Execution

117

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

What is expected # of matching tuples?

(a) say $R_1.C$ is key, $R_2.C$ is foreign key
then expect = 1

(b) say $V(R_1.C) = 5000$, $T(R_1) = 10,000$
with uniform assumption
expect = $10,000/5,000 = 2$

CS 525



Notes 10 - Query Execution

118

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

What is expected # of matching tuples?

(c) Say $\text{DOM}(R_1, C) = 1,000,000$
 $T(R_1) = 10,000$
with alternate assumption
Expect = $\frac{10,000}{1,000,000} = \frac{1}{100}$

CS 525



Notes 10 - Query Execution

119

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Total cost with index join

(a) Total cost = $500 + 5000(1)1 = 5,500$

(b) Total cost = $500 + 5000(2)1 = 10,500$

(c) Total cost = $500 + 5000(1/100)1 = 550$

CS 525



Notes 10 - Query Execution

120

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

What if index does not fit in memory?

Example: say $R_1.C$ index is 201 blocks

- Keep root + 99 leaf nodes in memory
- Expected cost of each probe is

$$E = (0)\frac{99}{200} + (1)\frac{101}{200} \approx 0.5$$

CS 525



Notes 10 - Query Execution

121



Total cost (including probes)

$$\begin{aligned} &= 500+5000 \text{ [Probe + get records]} \\ &= 500+5000 [0.5+2] \quad \text{uniform assumption} \\ &= 500+12,500 = 13,000 \quad \text{(case b)} \end{aligned}$$

CS 525



Notes 10 - Query Execution

122



Total cost (including probes)

$$\begin{aligned} &= 500+5000 \text{ [Probe + get records]} \\ &= 500+5000 [0.5+2] \quad \text{uniform assumption} \\ &= 500+12,500 = 13,000 \quad \text{(case b)} \end{aligned}$$

For case (c):

$$\begin{aligned} &= 500+5000[0.5 \times 1 + (1/100) \times 1] \\ &= 500+2500+50 = 3050 \text{ IOs} \end{aligned}$$

CS 525



Notes 10 - Query Execution

123



So far

{	Nested Loop	5500	
	Merge join	1500	
	Sort+Merge Join	7500	→ 4500
	$R_1.C$ Index	5500	→ 3050 → 550
	$R_2.C$ Index	_____	

CS 525



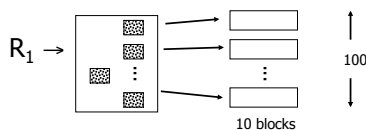
Notes 10 - Query Execution

124



Example 1(e) Partition Hash Join

- R_1, R_2 contiguous (un-ordered)
- Use 100 buckets
- Read R_1 , hash, + write buckets



CS 525

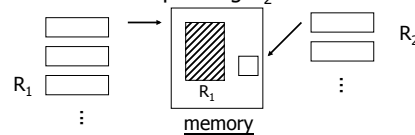


Notes 10 - Query Execution

125



- > Same for R_2
- > Read one R_1 bucket; build memory hash table
 - using different hash function h'
- > Read corresponding R_2 bucket + hash probe



☞ Then repeat for all buckets

CS 525



Notes 10 - Query Execution

126



Cost:

“Bucketize:” Read R_1 + write

Read R_2 + write

Join: Read R_1, R_2

Total cost = $3 \times [1000+500] = 4500$

CS 525



Notes 10 - Query Execution

127

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Cost:

“Bucketize:” Read R_1 + write

Read R_2 + write

Join: Read R_1, R_2

Total cost = $3 \times [1000+500] = 4500$

Note: this is an approximation since buckets will vary in size and we have to round up to blocks

CS 525

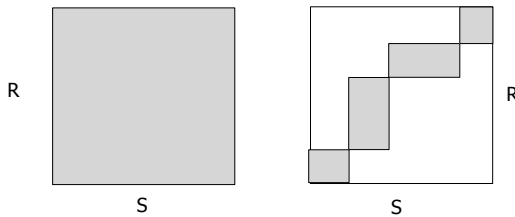


Notes 10 - Query Execution

128

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Why is Hash Join good?



CS 525



Notes 10 - Query Execution

129

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Minimum memory requirements:

Size of R_1 bucket = (x/k)

k = number of memory buffers

x = number of R_1 blocks

So... $(x/k) < k$

$k > \sqrt{x}$ need: $k+1$ total memory buffers

CS 525



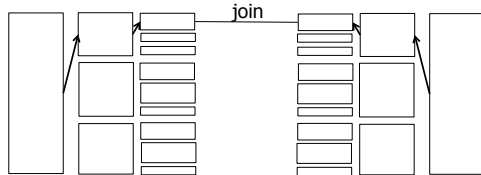
Notes 10 - Query Execution

130

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Can we use Hash-join when buckets do not fit into memory?:

- Treat buckets as relations and apply Hash-join recursively



CS 525



Notes 10 - Query Execution

131

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Duality Hashing-Sorting

- Both partition inputs
- Until input fits into memory
- Logarithmic number of phases in memory size

CS 525



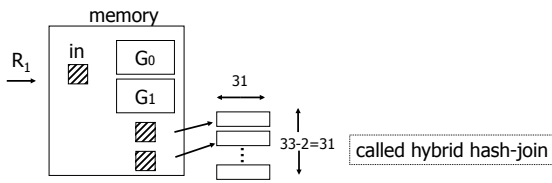
Notes 10 - Query Execution

132

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Trick: keep some buckets in memory

E.g., $k' = 33$ R_1 buckets = 31 blocks
keep 2 in memory



CS 525



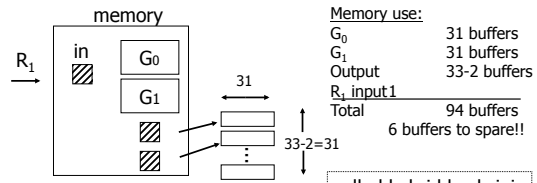
Notes 10 - Query Execution

133

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Trick: keep some buckets in memory

E.g., $k' = 33$ R_1 buckets = 31 blocks
keep 2 in memory



CS 525



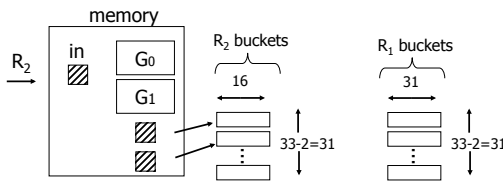
Notes 10 - Query Execution

134

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Next: Bucketize R_2

- R_2 buckets = $500/33 = 16$ blocks
- Two of the R_2 buckets joined immediately with G_0, G_1



CS 525



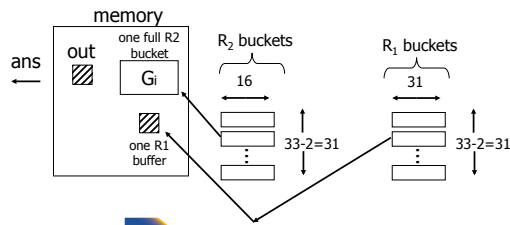
Notes 10 - Query Execution

135

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Finally: Join remaining buckets

- for each bucket pair:
 - read one of the buckets into memory
 - join with second bucket



CS 525



Notes 10 - Query Execution

136

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Cost

- Bucketize $R_1 = 1000 + 31 \times 31 = 1961$
- To bucketize R_2 , only write 31 buckets:
so, cost = $500 + 31 \times 16 = 996$
- To compare join (2 buckets already done)
read $31 \times 31 + 31 \times 16 = 1457$

Total cost = $1961 + 996 + 1457 = 4414$

CS 525

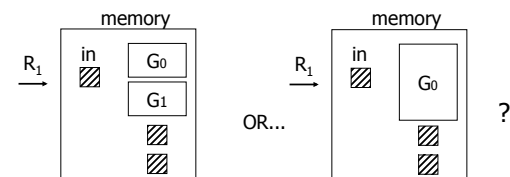


Notes 10 - Query Execution

137

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

• How many buckets in memory?



• See textbook for answer...

CS 525



Notes 10 - Query Execution

138

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Another hash join trick:

- Only write into buckets <val,ptr> pairs
- When we get a match in join phase, must fetch tuples

CS 525



Notes 10 - Query Execution

139



- To illustrate cost computation, assume:
 - 100 <val,ptr> pairs/block
 - expected number of result tuples is 100

CS 525



Notes 10 - Query Execution

140



- To illustrate cost computation, assume:
 - 100 <val,ptr> pairs/block
 - expected number of result tuples is 100
- Build hash table for R_2 in memory
5000 tuples \rightarrow $5000/100 = 50$ blocks
- Read R_1 and match
- Read $\sim 100 R_2$ tuples

CS 525



Notes 10 - Query Execution

141



- To illustrate cost computation, assume:
 - 100 <val,ptr> pairs/block
 - expected number of result tuples is 100
- Build hash table for R_2 in memory
5000 tuples \rightarrow $5000/100 = 50$ blocks
- Read R_1 and match
- Read $\sim 100 R_2$ tuples

<u>Total cost</u> =	Read R_2 :	500
	Read R_1 :	1000
	Get tuples:	<u>100</u>
		1600

CS 525



Notes 10 - Query Execution

142



So far:

Iterate	5500
Merge join	1500
Sort+merge joint	7500
$R_1.C$ index	5500 \rightarrow 550
$R_2.C$ index	_____
Build $R_1.C$ index	_____
Build $R_2.C$ index	_____
Hash join	4500+
with trick, R_1 first	4414
with trick, R_2 first	_____
Hash join, pointers	1600

CS 525



Notes 10 - Query Execution

143



Yet another hash join trick:

- Combine the ideas of
 - block nested-loop with hash join
- Use memory to build hash-table for one chunk of relation
- Find join partners in $O(1)$ instead of $O(M)$
- Trade-off
 - Space-overhead of hash-table
 - Time savings from look-up

CS 525



Notes 10 - Query Execution

144



Summary

- Nested Loop ok for “small” relations (relative to memory size)
 - Need for complex join condition
- For equi-join, where relations not sorted and no indexes exist, hash join usually best

CS 525



Notes 10 - Query Execution

145

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

- Sort + merge join good for non-equi-join (e.g., $R_1.C > R_2.C$)
- If relations already sorted, use merge join
- If index exists, it could be useful (depends on expected result size)

CS 525



Notes 10 - Query Execution

146

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Join Comparison

N_i = number of tuples in R_i
 $B(R_i)$ = number of blocks of R_i
 $\#P$ = number of partition steps for hash join
 P_{ij} = average number of join partners

Algorithm	#I/O	Memory	Disk Space
Nested Loop (block)	$B(R_1) / (M-1) * [\min(B(R_1), M-1) + B(R_2)]$	3	0
Index Nested Loop	$B(R_1) + N_1 * P_{12}$	$B(\text{Index}) + 2$	0
Merge (sorted)	$B(R_1) + B(R_2)$	Max tuples =	0
Merge (unsorted)	$B(R_1) + B(R_2) + (\text{sort} - 1 \text{ pass})$	sort	$B(R_1) + B(R_2)$
Hash	$(2\#P + 1) (B(R_1) + B(R_2))$	$\text{root}(\max(B(R_1), B(R_2)), \#P + 1)$	$\sim B(R_1) + B(R_2)$

CS 525



Notes 10 - Query Execution

147

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Why do we need nested loop?

- Remember not all join implementations work for all types of join conditions

Algorithm	Type of Condition	Example
Nested Loop	any	a LIKE '%hello%'
Index Nested Loop	Supported by index: Equi-join (hash) Equi or range (B-tree)	a = b a < b
Merge	Equalities and ranges	a < b, a = b AND c = d
Hash	Equi-join	a = b

CS 525



Notes 10 - Query Execution

148

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Outer Joins

- How to implement (left) outer joins?
- Nested Loop and Merge
 - Use a flag that is set to true if we find a match for an outer tuple
 - If flag is false fill with NULL
- Hash
 - If no matching tuple fill with NULL

CS 525



Notes 10 - Query Execution

149

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Merge Left Outer Join

$R \bowtie_{B=C} S$

Output: (a,1,1,X)

R		S	
A	B	C	D
a	1	1	x
d	4	2	y
e	5	2	e
		6	q
		7	d

Z_R points to the row (a, 1) in R. Z_S points to the row (1, x) in S.

CS 525



Notes 10 - Query Execution

150

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Merge Left Outer Join

$R \bowtie_{B=C} S$

R

A	B
a	1
d	4
e	5

No match for (d,4)
Output: (d,4,NULL,NULL)

S

C	D
1	x
2	y
2	e
6	q
7	d

CS 525



Notes 10 - Query Execution

151

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Merge Left Outer Join

$R \bowtie_{B=C} S$

R

A	B
a	1
d	4
e	5

No match for (e,5)
Output: (e,5,NULL,NULL)

S

C	D
1	x
2	y
2	e
6	q
7	d

CS 525



Notes 10 - Query Execution

152

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Operators Overview

- (External) Sorting
- Joins (Nested Loop, Merge, Hash, ...)
- **Aggregation (Sorting, Hash)**
- Selection, Projection (Index, Scan)
- Union, Set Difference
- Intersection
- Duplicate Elimination

CS 525



Notes 10 - Query Execution

153

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Aggregation

- Have to compute aggregation functions
 - for each group of tuples from input
- Groups
 - Determined by equality of group-by attributes

CS 525



Notes 10 - Query Execution

154

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Aggregation Example

```
SELECT sum(a), b
FROM R
GROUP BY b
```

a	b
3	1
4	2
3	1
1	2
1	2

sum(a)	b
6	1
6	2

CS 525



Notes 10 - Query Execution

155

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Aggregation Function Interface

- `init()`
 - Initialize state
- `update(tuple)`
 - Update state with information from tuple
- `close()`
 - Return result and clean-up

CS 525



Notes 10 - Query Execution

156

IIT College of Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Implementation SUM(A)

- `init()`
 - `sum := 0`
- `update(tuple)`
 - `sum += tuple.A`
- `close()`
 - **return** `sum`

CS 525



Notes 10 - Query Execution

157

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Aggregation Implementations

- **Sorting**
 - Sort input on group-by attributes
 - On group boundaries output tuple
- **Hashing**
 - Store current aggregated values for each group in hash table
 - Update with newly arriving tuples
 - Output result after processing all inputs

CS 525



Notes 10 - Query Execution

158

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Grouping by sorting

- Similar to Merge join
- Sort R on group-by attribute
- Scan through sorted input
 - If group-by values change
 - Output using `close()` and call `init()`
 - Otherwise
 - Call `update()`

CS 525



Notes 10 - Query Execution

159

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Aggregation Example

```
SELECT sum(a), b
FROM R
GROUP BY b
```

sort

a	b	a	b
3	1	3	1
4	2	3	1
3	1	4	2
1	2	1	2
1	2	1	2

`init()`
0

CS 525



Notes 10 - Query Execution

160

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Aggregation Example

```
SELECT sum(a), b
FROM R
GROUP BY b
```

a	b
3	1
3	1
4	2
1	2
1	2

`update(3,1)`
3

CS 525



Notes 10 - Query Execution

161

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Aggregation Example

```
SELECT sum(a), b
FROM R
GROUP BY b
```

a	b
3	1
3	1
4	2
1	2
1	2

`update(3,1)`
6

CS 525



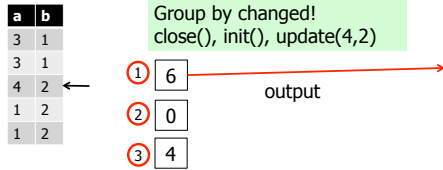
Notes 10 - Query Execution

162

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Aggregation Example

```
SELECT sum(a), b
FROM R
GROUP BY b
```



CS 525



Notes 10 - Query Execution

163

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Grouping by Hashing

- Create in-memory hash-table
- For each input tuple probe hash table with group by values
 - If no entry exists then call init(), update(), and add entry
 - Otherwise call update() for entry
- Loop through all entries in hash-table and output calling close()

CS 525



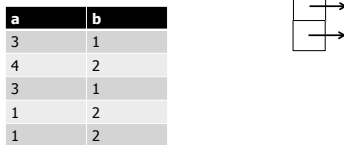
Notes 10 - Query Execution

164

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Aggregation Example

```
SELECT sum(a), b
FROM R
GROUP BY b
```



CS 525



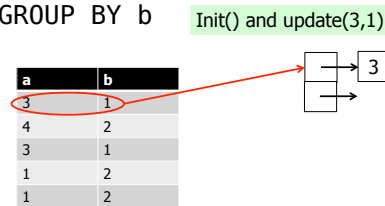
Notes 10 - Query Execution

165

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Aggregation Example

```
SELECT sum(a), b
FROM R
GROUP BY b
```



CS 525



Notes 10 - Query Execution

166

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Aggregation Example

```
SELECT sum(a), b
FROM R
GROUP BY b
```

Init() and update(4,2)



CS 525



Notes 10 - Query Execution

167

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Aggregation Example

```
SELECT sum(a), b
FROM R
GROUP BY b
```

update(3,1)



CS 525



Notes 10 - Query Execution

168

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Aggregation Example

```
SELECT sum(a), b
FROM R
GROUP BY b
```

- Loop through hash table entries
- Output tuples

a	b
3	1
4	2
3	1
1	2
1	2



CS 525



Notes 10 - Query Execution

169

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Aggregation Summary

- Hashing
 - No sorting -> no extra I/O
 - Hash table has to fit into memory
 - No outputs before all inputs have been processed
- Sorting
 - No memory required
 - Output one group at a time

CS 525



Notes 10 - Query Execution

170

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Operators Overview

- (External) Sorting
- Joins (Nested Loop, Merge, Hash, ...)
- Aggregation (Sorting, Hash)
- Selection, Projection (Index, Scan)
- Union, Set Difference
- Intersection
- Duplicate Elimination

CS 525



Notes 10 - Query Execution

171

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Duplicate Elimination

- Equivalent to group-by on all attributes
- -> Can use aggregation implementations
- Optimization
 - Hash
 - Directly output tuple and use hash table only to avoid outputting duplicates

CS 525



Notes 10 - Query Execution

172

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Operators Overview

- (External) Sorting
- Joins (Nested Loop, Merge, Hash, ...)
- Aggregation (Sorting, Hash)
- Selection, Projection (Index, Scan)
- Union, Set Difference
- Intersection
- Duplicate Elimination

CS 525



Notes 10 - Query Execution

173

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Set Operations

- Can be modeled as join
 - with different output requirements
- As aggregation/group by on all columns
 - with different output requirements

CS 525



Notes 10 - Query Execution

174

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Union

- Bag union
 - Append the two inputs
 - E.g., using three buffers
- Set union
 - Apply duplicate removal to result

CS 525



Notes 10 - Query Execution

175

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Intersection

- Set version
 - Equivalent to join + project + duplicate removal
 - 3-state aggregate function (found left, found right, found both)
- Bag version
 - Join + project + $\min(i,j)$
 - Aggregate $\min(\text{count}(i), \text{count}(j))$

CS 525



Notes 10 - Query Execution

176

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Set Difference

- Using join methods
 - Find matching tuples
 - If no match found, then output
- Using aggregation
 - $\text{count}(i) - \text{count}(j)$ (**bag**)
 - $\text{true}(i) \text{ AND } \text{false}(j)$ (**set**)

CS 525



Notes 10 - Query Execution

177

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Summary

- Operator implementations
 - Joins!
 - Other operators
- Cost estimations
 - I/O
 - memory
- Query processing architectures

CS 525



Notes 10 - Query Execution

178

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY

Next

- Query Optimization Physical
- -> How to **efficiently** choose an **efficient** plan

CS 525



Notes 10 - Query Execution

179

IIT College of
Science and Letters
ILLINOIS INSTITUTE OF TECHNOLOGY