NYU, Tandon School of Engineering

December 13, 2024

CS-GY 6083

Principles of Database Systems

Section A, Fall 2024

Homework #6

Submitted by:

Khwaab Thareja

N15911999

Kt3180

Guided By: Prof Phyllis Frankl

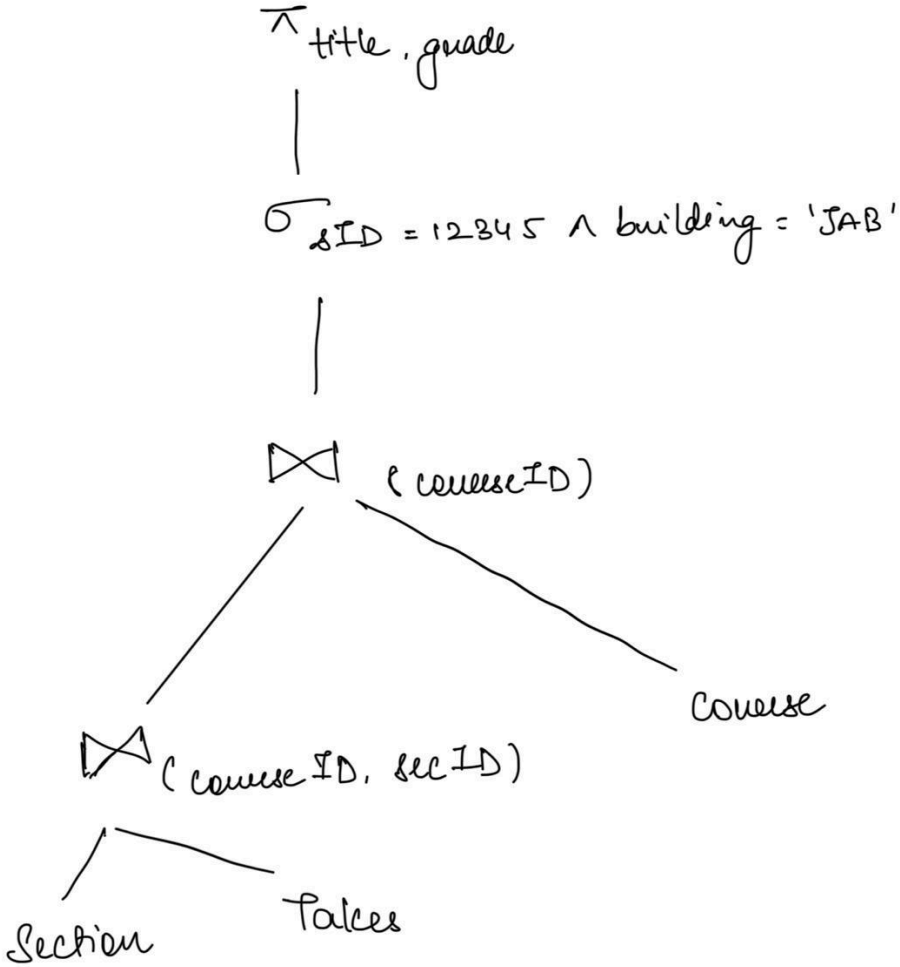
Answer 1.1

SELECT C.title, T.grade FROM Course C

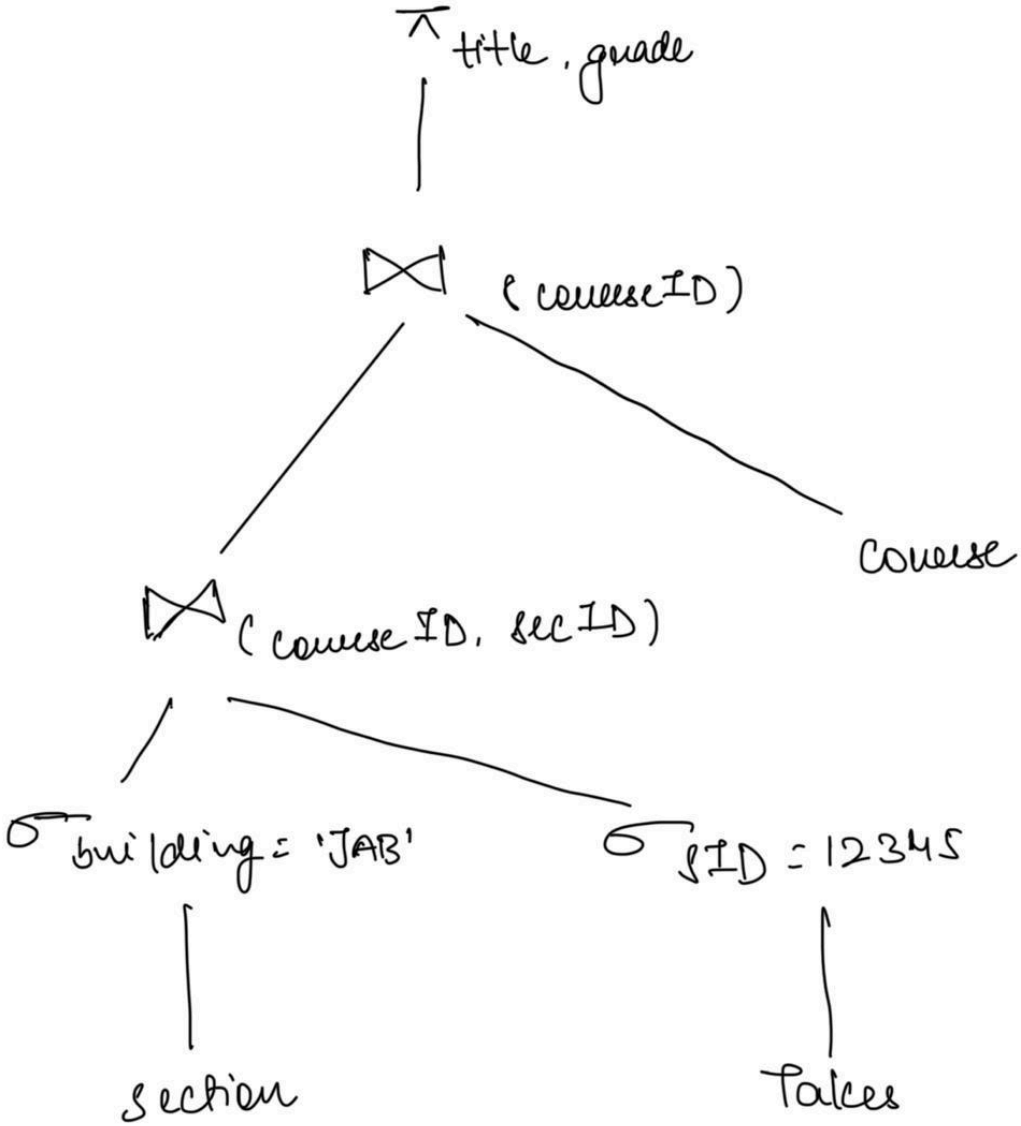
JOIN Section S ON C.courseID = S.courseID

JOIN Takes T ON S.courseID = T.courseID AND S.secID = T.secID AND S.sem = T.sem AND S.year = T.year WHERE T.sID = '12345' AND S.building = 'JAB';

Answer 1.2



Answer 1.3


title, grade
(courseID)
Course
(courseID, secID)
building = 'JAB'
ID = 12345
Section
Takes

Answer 1.4

Section 1

In this part, Scan takes to find the courses taken by student and their grades in those courses.

a.) The part of the expression that is being evaluated is : **σ sID=12345(Takes)**

b.) A full scan takes place since there is no indexing

- Here relation size is **40,000,000** records
- Each 64 bytes which makes it → **2560** MB
- Since table takes **2560** MB, it doesnt fit in the memory (100MB)

c.) Since it is assumed to be sequential read, only one seek is required ,total seek time: 10ms

d.) Since it is sequential read, Block Transfers:

- Takes size: 2560MB = **640,000** blocks (4KB each)
- Transfer time: 2560MB / 50MBps = **51.2 second**

e.) Since students has taken 10 courses, hence

- Result Size:- 10 records - Each record 64 bytes- therefore 10*64 → Total: **640 bytes**

f.) Now size of each block is <100M, So there is no need to write it out, it can stay in the main memory.

g.) Summary : 1 block is 640 Bytes

- Seeks: 10ms → Transfer: 51.2s → Total: ~**51.21 seconds**

Section 2:

a.) The part of the expression that is being evaluated is: σbuilding="JAB"

b.) A full scan takes place since there is no indexing.

- Relation size: 1,000,000 records.
- Each record size: 512 bytes → Total: 500MB Since the table is 500MB

c.) Number of seeks: 1.

- Total seek time: 10ms

d.) Block Transfers:

- Block size: 4KB
- Total blocks: 500MB/4KB=125,000
- Transfer time: 500MB/50MBps=10s

e.) Since 50% of sections are in the "JAB" building:

- Result Size:
    - 1,000,000×50%=500,000 records.
    - Each record size: 512 bytes.
    - Total: 500,000×512 bytes=250MB.

f.) Since the result size is 250MB>100MB, it must be written to disk.

- Write Time: 250MB/50MBps=5s.

g.) Summary:

- Seeks: 10ms.
- Transfer: 10s.
- Write: 5s.
- Total Time: ~15.01s.

Section 3:

a.) The part of the expression that is being evaluated is:

- **(σ sID=12345(Takes))⋈(σ building="JAB"(Section))**

b.) Join Method: Block Nested Loop Join.

- Outer Relation: Filtered Section (250MB).
- Inner Relation: Filtered Takes (640 bytes, fits in memory).
- Number of passes: ⌈250MB/100MB⌉=3

c.) Number of seeks: 3 passes for the outer relation.

- Total Seek Time: 3×10ms=30ms.
-

d.) Block Transfers:

- Outer relation size: 250MB×3=750MB.
- Transfer time: 750MB/50MBps=15s.

e.) Result Size:

- Student has 10courses, 50% are in "JAB".
- Result: 10×50% =5 records.
- Each record size: 576 bytes.
- Total: 5×576 bytes=2.88KB.

f.) Since the result size is 2.88KB, it fits in memory.

g.) Summary:

- Seeks: 30ms.
- Transfer: 15s.
- Total Time: ~15.03s.

Section 4:

a.) The part of the expression that is being evaluated is: (Filtered Results)⋈Course

b.) Join Method:

- Outer Relation: Filtered Results (2.88KB, fits in memory).
- Inner Relation: Course (10MB).
- Single pass needed.

c.) Number of seeks: 1.

- Total Seek Time: **10ms**.

d.) Block Transfers:

- Inner relation size: **10MB**.
- Transfer time: 10MB/50MBps=**0.2s.**

e.) Result Size:

- Assume 1-to-1 match for 5 records.
- Result size: negligible (<1KB).

f.) Since the result size is negligible, it fits in memory.

g.) Summary:

- Seeks: **10ms.**
- Transfer: **0.2s**.
- Total Time: ~**0.21s.**

Answer 2

1. How is this index potentially helpful?

The unclustered B+ tree index on the Takes table uses the sID column, with each key (sID) pointing to a specific record in the table. This index helps by allowing direct access to the records for student 12345, so there's no need to scan the whole table. This makes the first step of the query plan, filtering the Takes table for σ sID=12345, much faster. However, because the index is unclustered, the records are spread out across different blocks, which means extra steps are needed to fetch them.

2. Cost of Using the Index

Step 1: Using the Index on Takes

a.) Expression Being Evaluated: σ sID=12345

b.) Index Structure and Traversal:

- The B+ tree index is assumed to have a height (H) of 4 due to the large size of the Takes table (40Mrecords).
- The index lookup involves traversing 4 levels of the tree to locate sID=12345

c.) Number of Seeks:

- Index Traversal: H=4 → 4 seeks.
- Data File Access: 10 seekss (one for each of the student's records).
- Total Seeks: 4+10=14
- Total Seek Time: 14×10ms=140ms

d.) Data Transfers:

- Index Nodes: Traversing 4 index nodes (4KB each) → 16KB.
- Record Fetches: 10 records, each in a separate block (4KB each) → 40KB.
- Total Data Transferred: 16KB+40KB=56KB.
- Transfer Time: 56KB/50MBps=0.00112s.

e.) Fits in Memory:

- The fetched records (640 bytes) fit easily into memory. No need to write intermediate results to disk.

f.) Summary:

- Seeks: 14→ 140ms.
- Transfers: 0.00112s.
- Total Time: 140.1ms.

Step 1: Without Index (Sequential Scan)

- A full table scan of Takes (size 2560MB):
    - Seeks: 1 → 10ms.
    - Transfers: 2560MB/50MBps=51.2s.
    - Total Time: ~51.21.

Steps 2 and 3 (Unaffected):

Using the index does not impact Steps 2 and 3:

- Step 2: Filtering Section for the "JAB" building still requires a full scan.
- Step 3: Joining with Course involves sequential scans and is unaffected by the Takes index.

3. Is using the index beneficial for this query?

Yes, using the index is beneficial for the following reasons:

1. Reduction in Step 1 Cost:
    - Without Index: ~51.21s.
    - With Index: ~140.1ms.
2. Using the index reduces the cost of Step 1 by over 50 seconds, making it significantly more efficient.
3. Data Reduction:
    - The index allows us to directly access the 10 relevant records for the student 12345, reducing the amount of data passed to Step 2.

Hence using secondary(Unclustered) B+ Tree index is highly beneficial for this query