CS3402 Database Systems

Assignment 2 (due: 21/Nov/2022 @11:59pm!)

Question A.

1) What are the advantages and disadvantages of hash functions relative to B-tree indices? How might the type of index available influence the choice of a query-processing strategy? **[15 marks]**

Answer:

Hash functions have a few advantages over B-tree:

- requires no indices, hence no extra space cost (3 marks)
- can support selection-type operations very fast (3 marks)

Hash functions have some disadvantages relative to B-tree:

- collisions must be handled by hash function look-ups (3 marks)
- being non-clustering, range search will be very slow (3 marks)

Thus, depending on the type of query, if selection-type operations are involved, we may choose hashing if it is available, or if range queries are to be processed, we may go for B-tree if it's available. (3 marks)

2) When is it preferable to use a dense index rather than a sparse index? **[8 marks]**

Answer:

It is preferable to use a dense index rather than a sparse index when

- the file is not ordered on the indexed field (such as when the index is a secondary index) (4 marks)
- or, when the index file is small enough to fit in the main memory (4 marks)

3) What is the difference between primary index and a secondary index? **[7 marks]**

Answer:

The difference between primary index and a secondary index lies in the facts that:

- The primary index is on the field which specifies the sequential order of the data file (4 marks)
- There can only be one primary index while many secondary indices (3 marks)

4) If a hash structure is used on a search key for which range queries are likely, what property should the hash functions have? **[10 marks]**

Answer:

The hash function should preserve the order of the data.
That is, if h is the hash function, and $x < y$, then it implies that $h(x) < h(y)$.

Question B.

Consider the relations:

STUDENT(SNAME, SID, BDATE, ADDRESS, DNUM)
COURSE(CNAME, CID, LEVEL, LECTURER)
COURSE_TAKING(STUDENTID, COURSEID, GRADE)

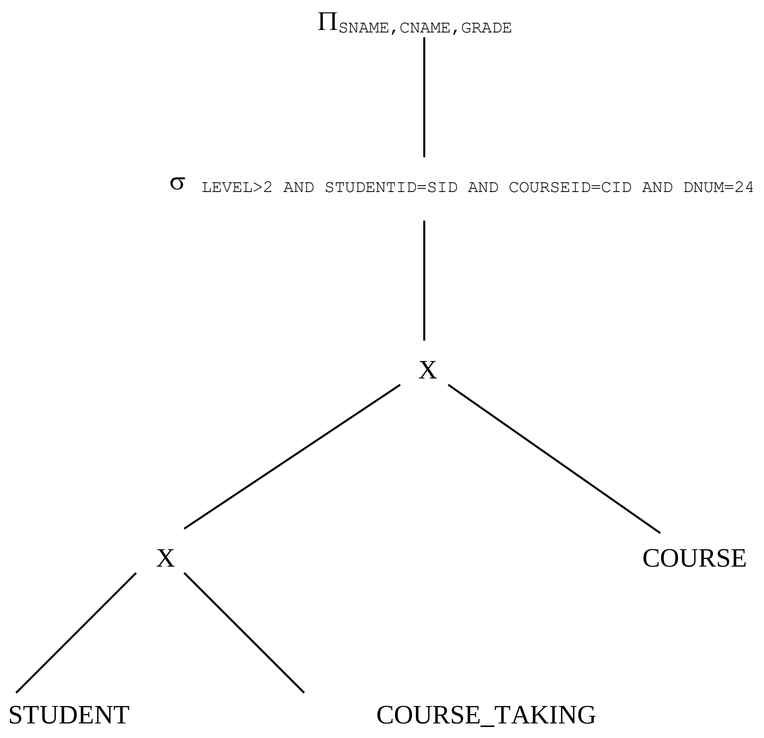
as well as the following SQL query:

```
SELECT SNAME, CNAME, GRADE
FROM STUDENT, COURSE_TAKING, COURSE
WHERE LEVEL>2 AND STUDENTID=SID
        AND COURSEID=CID AND DNUM = 24;
```

a) Draw a canonical query tree for the above SQL query. **[9 marks]**

b) Apply the optimization rules to the above query tree and come up with the most optimized query tree using those rules. State the necessary assumptions for your decision. **[21 marks]**

Answer:

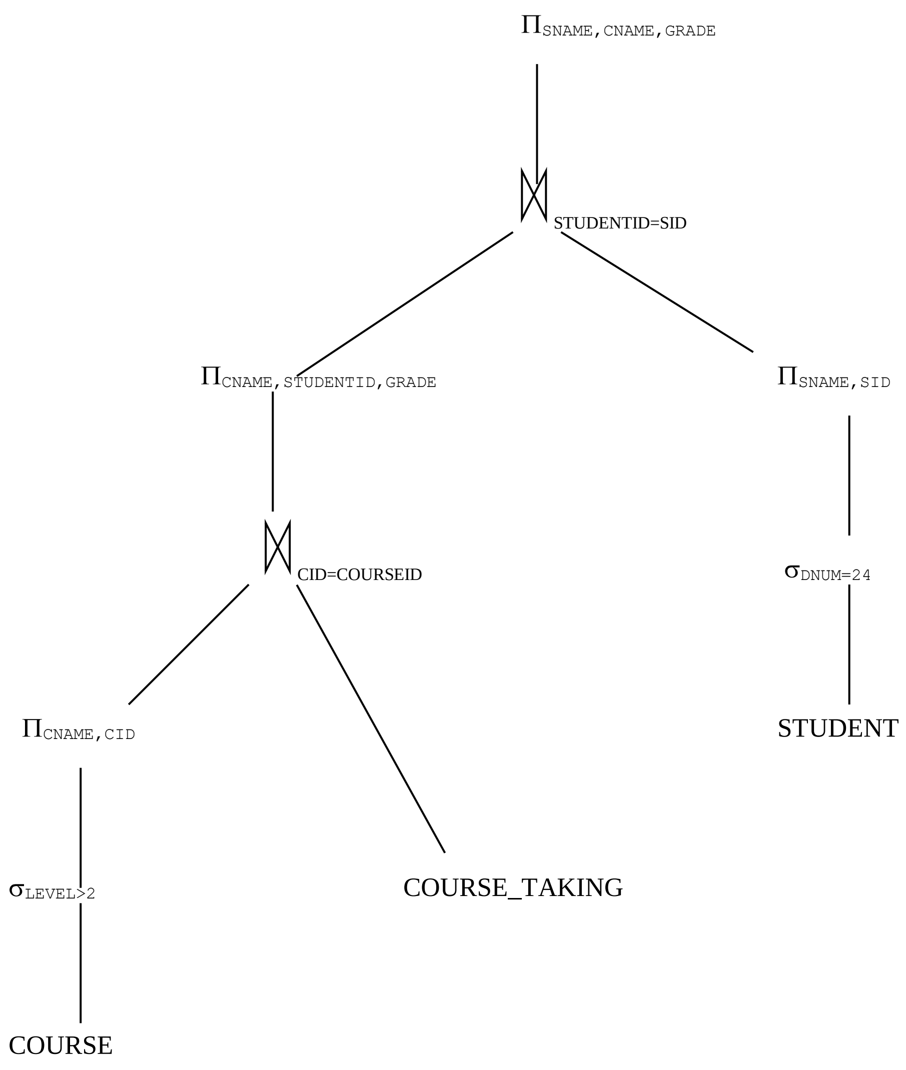


Marking scheme: partially correct gets =<3 points.

b) Apply the optimization rules to the above query tree and come up with the most optimized query tree using those rules. State the necessary assumptions for your decision.

(21 marks)

Assumption: *There are less courses with level > 2, compared to the number of students in department 24.*



Marking scheme: One optimization step missing will get 2 points deducted and some typos or missing symbols will also cause 2-4 points deducted.

Question D.

[30 marks]

1) Do you agree that a nonrecoverable schedule results in a loss of transaction atomicity? Explain your answer.

2) When the system recovers from a crash, in what order must transactions be undone and redone?

3) Suppose that the system crashes during the time it is recovering from a prior crash. When the system again recovers, what action must be taken?

Answer:

1) Yes: If a failure occurs, it may be necessary to *abort a committed transaction*, because in a nonrecoverable schedule a transaction T is allowed to commit ***before*** all the transactions from which T has read some data have committed. **(8 marks)**

Answer:

2) First, undo all transactions that began prior to the crash but have not committed (ie, there is a **start** record but no **commit** record in the log). These transactions are undone *in reverse temporal order* (newest to oldest);

Next, redo all transactions that committed prior to the crash (ie, there is a **commit** record in the log). These transactions are redone *in temporal order* (from oldest to newest). **(14 marks)**

Answer:

3)The answer is *exactly the same* as for that of Question D 2) above, because undo and redo are idempotent (so it is acceptable to undo or to redo a transaction several times). The database will be consistent once the recovery activity has been completed. **(8 marks)**