Q-1 What are the differences among primary, secondary, and clustering indexes? (4.5M)

Ans: (1.5 M for each)

Primary Index: A primary index is specified on the ordering key field (every record has a unique value for search field) of an ordered file of records.

Clustering Index: like primary index. Used when the ordering field is not a key field (multiple records may be having the same value for search field).

A file can have at most one primary index or one clustering index, but not both.

Secondary Index: can be specified on any non-ordering field of a file. A file can have several secondary indexes.

Q-2 What is the problem of redundancy in database? Give examples of insert, delete, and update anomalies. (4.5M)

Ans: (3M for definitions, 1.5M for examples)

Redundancy: Duplication of same information at several places. May lead to high storage cost, access cost, inconsistency of data.

Insert Anomalies: Multiple unrelated information kept together. Hence cannot insert one without the other (eg., employee, department).

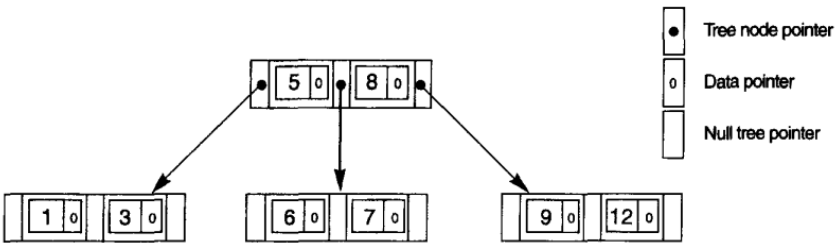
Update Anomaly: multiple copies of the same data may disagree whenever update fail to modify all the copies.

Delete Anomaly: Deleting one information may result in losing the other (eg., employee, department).

Q-3 What is the order p of a B+-tree? Describe the structure of both internal and leaf nodes of a B+-tree using suitable example. How does a B-tree differ from a B+-tree? (8M)

2M + 3M + 3M

B+ tree: Balanced Search tree, all leaf nodes are at the same level, grows bottom up.



A B-tree of order p = 3. The values were inserted in the order 8,5, 1, 7, 3, 12,9,6.

Unlike B-Trees, the structure of leaf nodes differs from the structure of internal nodes in B+ Trees. Unlike B-trees, In B+ tree, Data pointers are stored at leaf nodes only. B+ tree index is highly cost efficient in range queries. In B+ trees, leaf nodes have entry for every value of search field, along with pointer to block containing that search field value record. Some search

field values from the leaf nodes are repeated in the internal nodes to guide the search. The pointers in internal nodes are tree pointers to blocks that are tree nodes, whereas the pointers in leaf nodes are data pointers to the data file records or block. For order p B+ tree, each leaf can hold floor((p-1)/2) to (p-1) values.

Q-4) For the schema R = (A, B, C, D, E) with the following set F of functional dependencies: A → BC, CD → E, B → D, E → A. a) Give a lossless decomposition into BCNF. b) Give a lossless, dependency-preserving decomposition into 3NF. (8M)

4M + 4M

Ans:

a)

result: = {R}; F^+ = {A → ABCDE, B → D, BC → ABCDE, CD → ABCDE, E → ABCDE}.

R is not in BCNF. B → D is a non-trivial FD that holds on R, B ∩ D = ∅, and B → ABCDE is not in F^+.

Therefore, result = (result – R) ∪ (R – D) ∪ (B, D), i.e. (A, B, C, E) ∪ (B, D). (A, B, C, E) and (B, D) are in BCNF. So this is a decomposition of R is in BCNF.

b)

1) Construct a canonical cover of F. In our case F_C = F.

2) Initially we have an empty set of Rj (j = 0). Therefore, none of Rj contains ABC (we take a dependency from the canonical cover A → B). So R_1 = (A, B, C). Consider CD → E. CDE is not in R_1, hence we add R_2 = (C, D, E). Similarly, we add R_3 = (B, D), and R_4 = (E, A).

3) R_1 contains a candidate key for R, therefore we do not need to add a relation consisting of a candidate key. Finally, the received decomposition is (A, B, C), (C, D, E), (B, D), (E, A) which is lossless-join dependency-preserving decomposition into 3NF.

Q-05 Explain about Buffer management in DBMS? Write short notes on the following. (i) Optical disk. (ii) Magnetic tapes.

Distribution:

Buffering of Disk Blocks – (04M)

Write-Ahead Logging (WAL), Steal / No-Steal, Force / No-Force approach (04M)

secondary storage devices : Description or comparison (4.5)

Optical disk - stores data digitally and uses laser beams to read and write data. Most of the optical disks are read-only once written. Mostly used for archival storage.

Magnetic tapes are sequential access devices that do not support direct access unlike magnetic disks and Solid-State Drives. Hence access is slower, but storage cost is less. Useful for archiving and backup storage of data as well as system logs.

Q-6: Explain Open hashing? Discuss their advantages and disadvantages. Compare dynamic hashing with static hashing. (12.5)

Ans:

In open hashing, the set of buckets is fixed, and there are no overflow chains. Instead, if a bucket is full, the system inserts records in some other bucket in the initial set of buckets B (using linear probing). Open hashing gives good insertion and lookup performance, hence extensively used in compilers, assemblers for lookup in symbol tables. Deletion is problematic in open hashing, which is frequent in database systems, hence in DBMS, closed hashing is used.

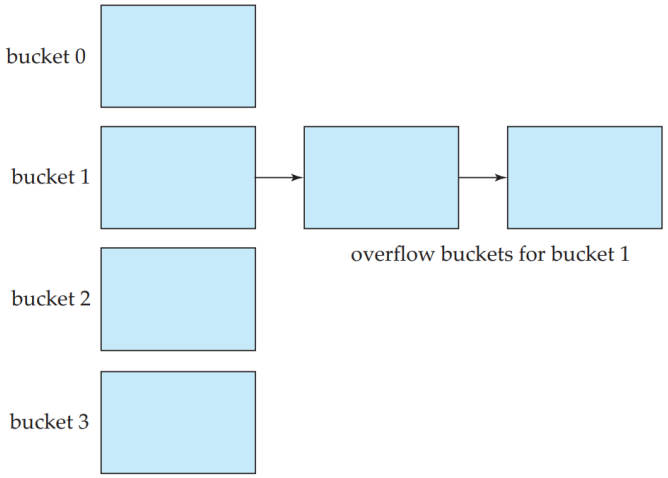


Figure 1Closed Hashing

In static hashing the number of buckets is fixed when the index is created. Handles overflow for buckets by chaining. Bucket may have an overflow chain.

Drawback: Large number of records stored in a single bucket or in one or more overflow buckets (in an overflow chain). Long overflow chains considerably increasing the cost of equality search.

In dynamic hashing, for efficient lookup, hash index can be rebuilt with an increased number of buckets. For example, if the number of records becomes twice the number of buckets, the index can be rebuilt with twice as many buckets as before.

dynamic hashing types a) linear hashing technique b) extendable hashing technique.

8) Discuss the relationship between Extendible and Linear Hashing. What are their relative merits? (4.5M)

(2.5M - Extendible, 2.0M – Linear)

extendable hashing technique -

Advantages - performance does not degrade as the file grows, copes with changes in database size by splitting and coalescing buckets as the database grows and shrinks, buckets can be

allocated dynamically, minimal space overheads, uses information from global depth and local depth.

Disadvantages: system access the bucket address table before accessing the bucket itself. Hence, lookup involves an additional level of indirection and hence the cost. Cost overheads of periodic doubling of the bucket address table.

linear hashing technique - The idea behind linear hashing is to allow a hash file to expand and shrink its number of buckets dynamically without needing a directory. It avoids the extra level of indirection associated with extendable hashing, at the possible cost of more overflow buckets.

Q-7 Explain how shadow paging helps to recover from transaction failure. (4.5M)

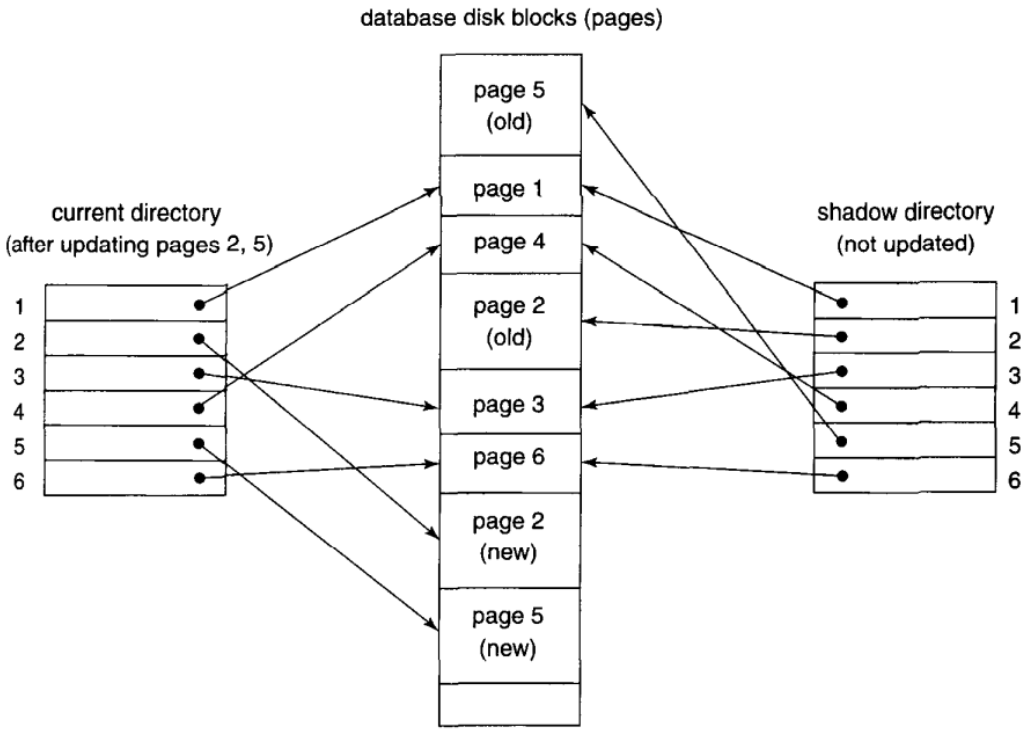


Figure 2Shadow Paging example

To recover from a failure, it is sufficient to free the modified database pages, to discard the current directory, and reinstate the shadow directory.

Q-9 Explain how the serializability is achieved using two- phase locking (2PL) protocol and timestamping protocol. (8M)

Serializable schedule - equivalent to some serial schedule. (1M)

2PL:

Growing phase: new locks can be acquired; none can be released.

Lock point: last lock acquired.

Shrinking phase: locks can be released; none acquired. (3M)

Example (1M)

Timestamping Protocol: Each transaction is assigned a unique timestamp, and protocol ensures conflicting operations are executed in the order of the transaction timestamps. (1M)

Example (2M)

Q-10 What is a schedule in DBMS? Define the concepts of recoverable, cascadeless, and strict schedules, and compare them in terms of their recoverability. (8M)

Schedule, recoverable, cascadeless, and strict schedules definitions (6M)

Comparison: (2M)

Q-11 Discuss how serializability is used to enforce concurrency control in a database system. What is the difference between conflict serializability and view serializability? (12.5 M)

Serializability definition with examples of conflict and view serializability – (8M)

Difference between conflict serializability and view serializability – (4.5M)

Q-12 Describe the write-ahead logging protocol. Explain UNDO-type and REDO-type log entries giving example. What are checkpoints and transaction commit points and why are they important? (12.5M)

(3M + 6.5M + 3M)

Recovery algorithm using redo, undo, checkpoints. Explain redo phase, undo phase, redo lists, undo lists, transaction commit points, checkpointing etc., using suitable example.

Checkpoints controls the expanding size of the undo-redo log, as well the time taken by the recovery process.

Commit point of a Transaction: effect of all the transaction operations on the database have been recorded in the log. In effect can be assumed to be permanently recorded in the data. In case of failure, can be redone from the log.

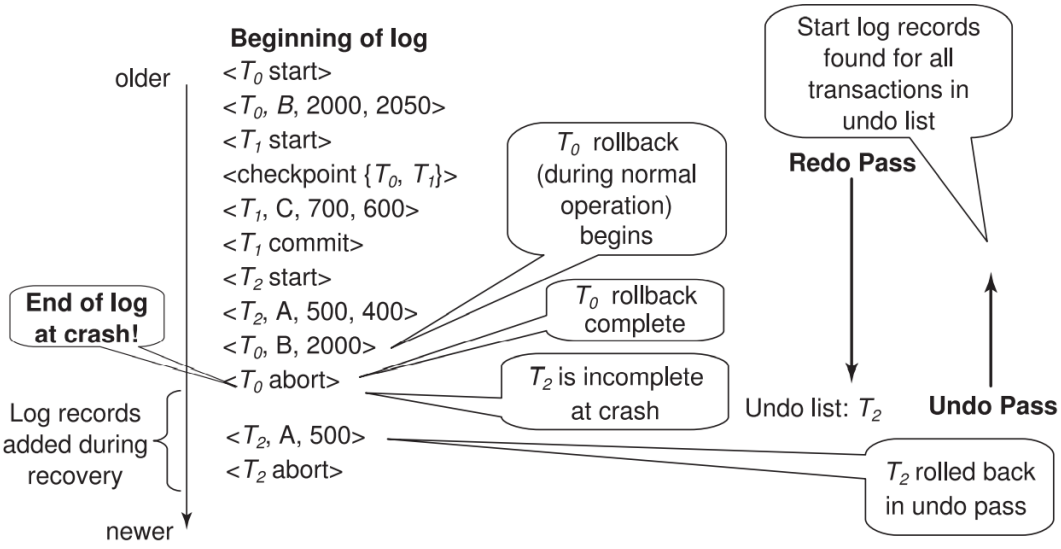


Figure 3:Example of logged actions, and actions during recovery.