Recovery Techniques

Log-based recovery: The log is a sequence of log records, recording all update activities in the database. An update log record describes a single database write

** For log records to be useful for recovery from system and disk failures, the log must reside in stable storage.

The log record has several fields:

- Transaction identifier → write operation
- Data-item identifier → location on disk
- Old value → prior to the write
- New value → after the write

Checkpoints: The presence of a <checkpoint> record in the log allows the system to determine its recovery procedure. Consider a transaction T_i that committed prior to the checkpoint. For such a transaction the $\langle T_i \text{ commit} \rangle$ record appears in the log before the checkpoint record. Any database modifications made by T_i must have been written to the database either prior to the checkpoint or as part of the checkpoint itself.

- undo(T_k): For all transactions T_k in T that have no $\langle T_k \text{ commit} \rangle$ record in the log, execute undo(T_k).

- redo(T_k): For all transactions T_k in T such that the record $\langle T_k \text{ commit} \rangle$ appears in the log, execute redo(T_k).

shadow

Deferred update: The deferred update protocol as follows:

(1) A transaction can not change the database on disk until it reaches its commit point.

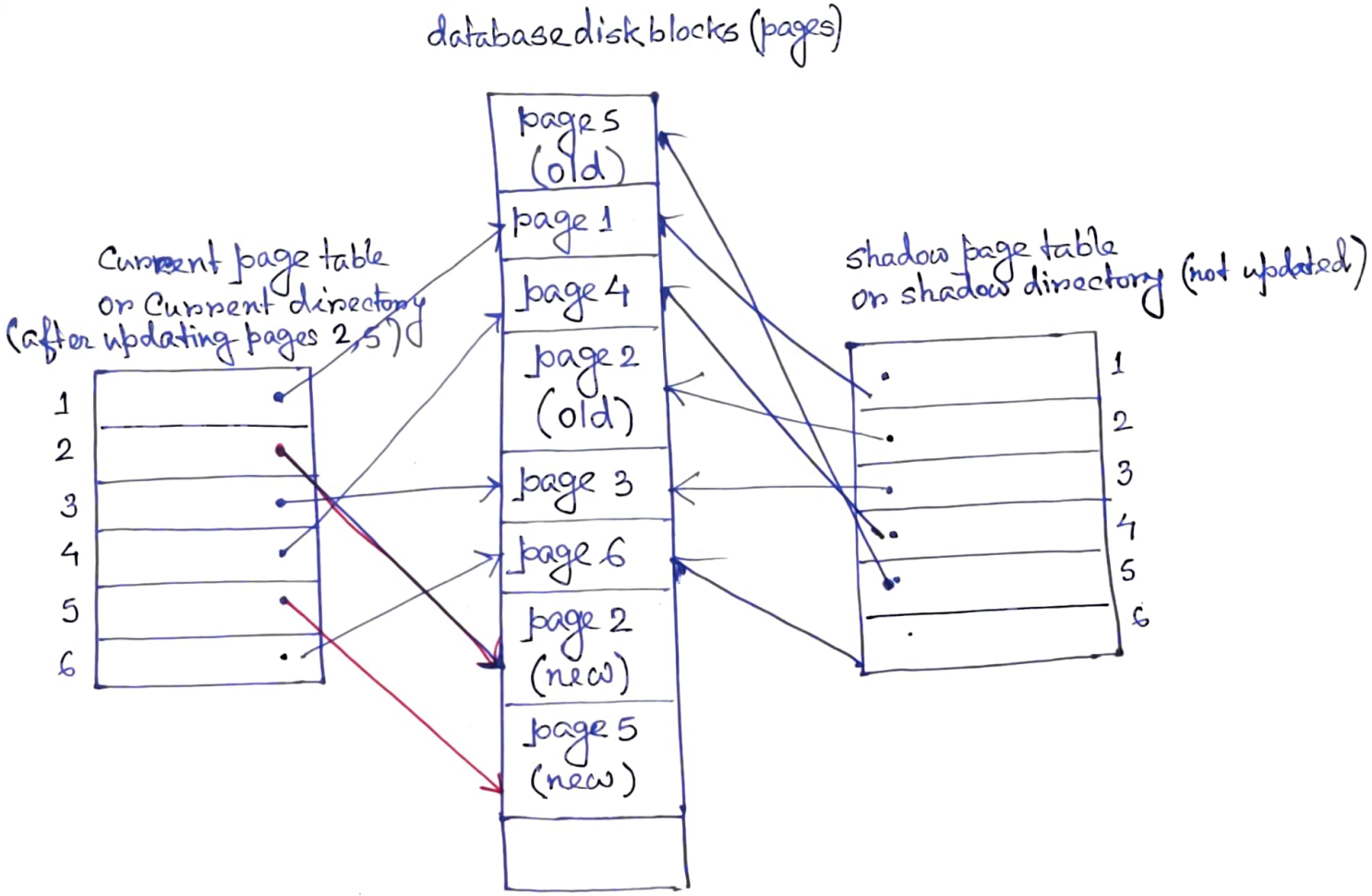
(2) A transaction does not reach its commit point until all its update operations are recorded in the log and the log is force-written to disk.

Immediate update: The immediate update protocol as follows:

(1) When a transaction issues an update command, the database can be updated "immediately", without any need to wait for transaction to reach its commit point.

(2) An update operation must still be recorded in the log (on disk) before it is applied to the database — so that we can recover in case of failure.

Shadow Paging :- The database is partitioned into some number of fixed-length blocks, which are referred to as pages.



When transaction begins execution, the current page table / current directory — whose entries point to the most recent on current database pages on disk — is copied into a shadow page table / shadow directory. The shadow directory is then saved on disk while the current directory is used by the transaction.

File and Indexing

Fixed length Records: Every record in the file has exactly the same size (in bytes).

- It is difficult to delete a record from the file. The space occupied by the record to be deleted must be filled with some other record of the file.
- Because of fixed block size of records, some records will cross block boundaries.

Variable-length Records: Different records in the file have different sizes.

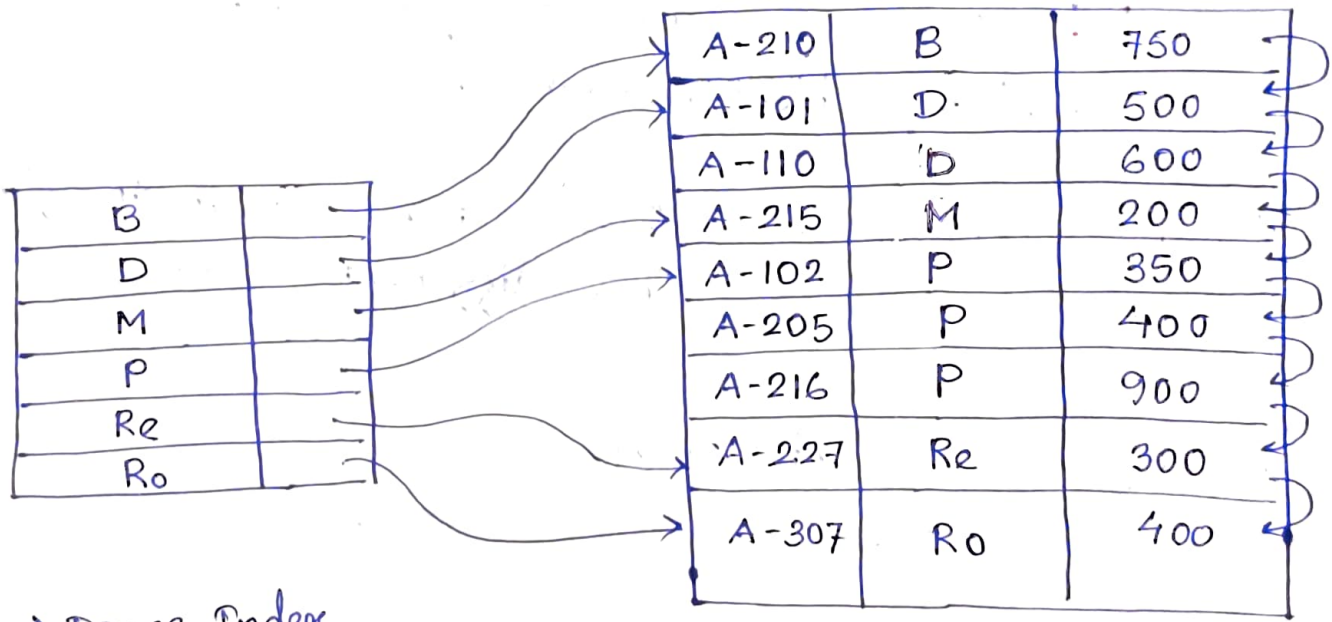
Indexing:
→ Sorted order.
→ easy to find the record.

Indices

- Ordered indices
  - primary index
    - Dense and Sparse indices
    - Multilevel indices
  - Secondary indices
- Hash indices

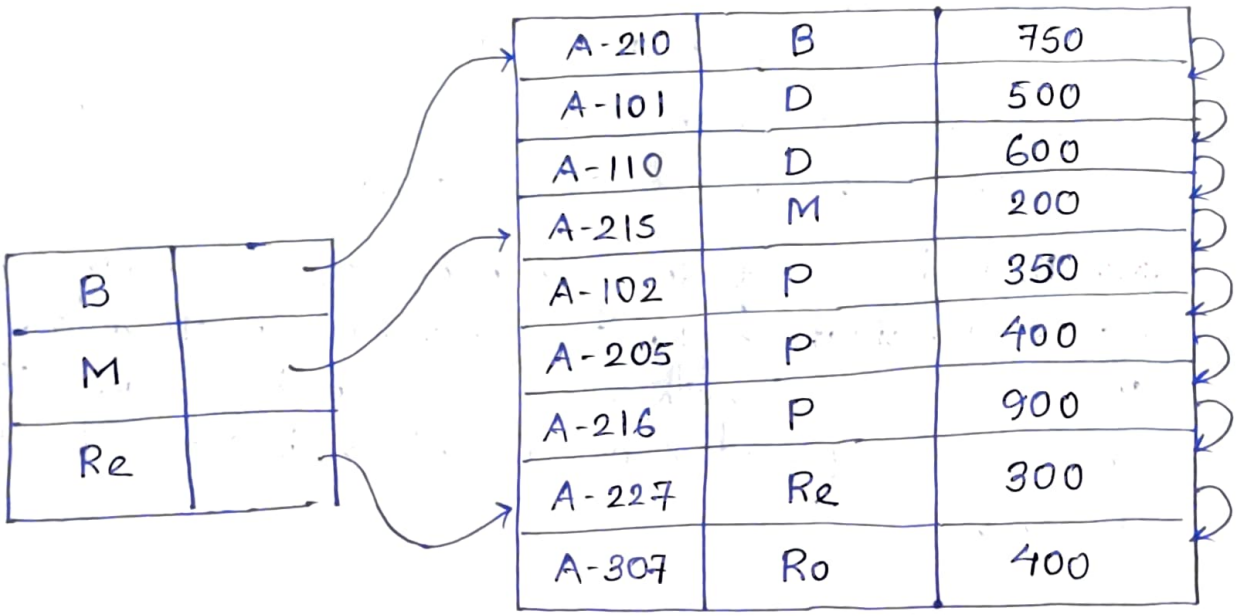
Dense Index : An index record consists of a search key value, and pointers to one or more records with that value as their search key value.

→ An index record appears for every search-key value in the file. In a dense primary index, the index record contains the search key value and a pointer to the first data record with that search key value.

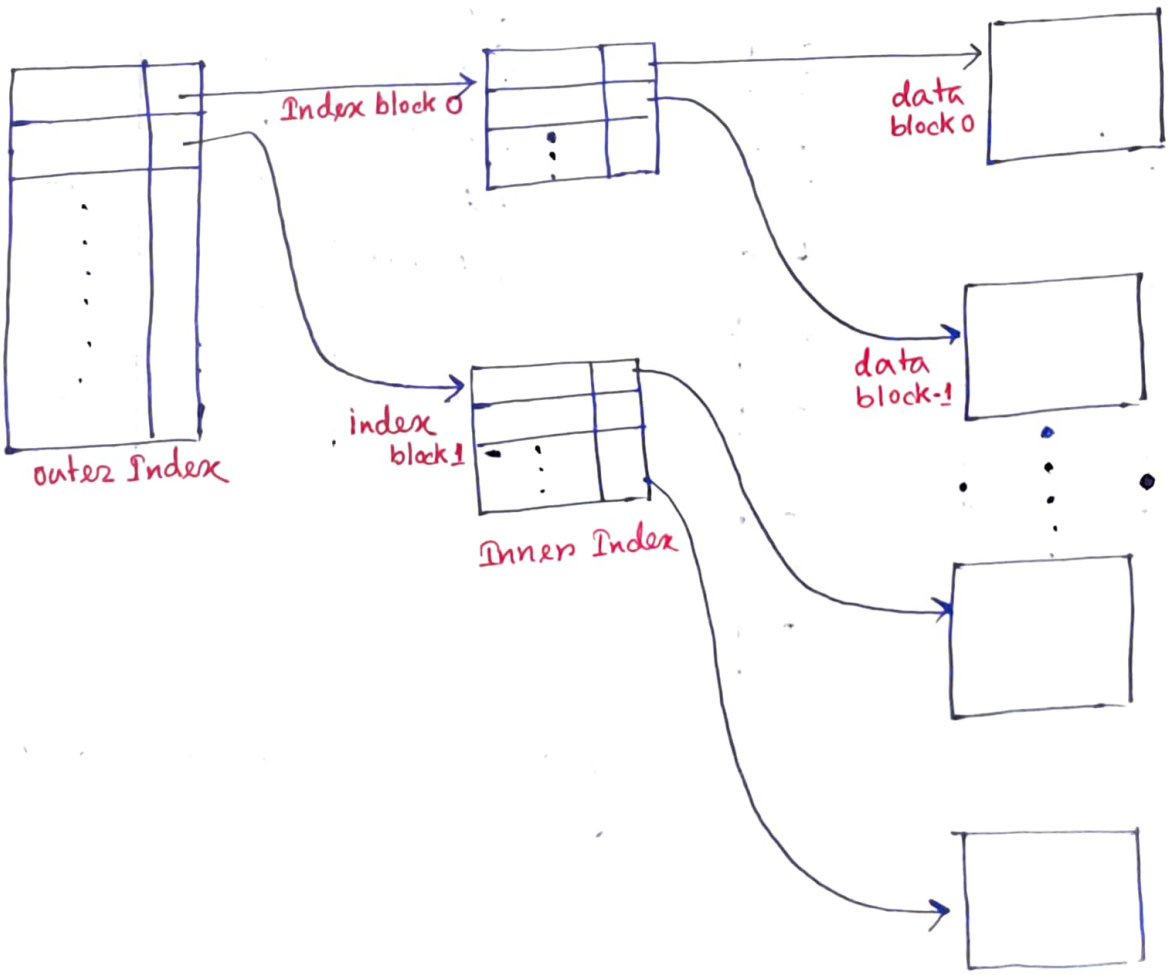


→ Dense Index
→ Sparse Index

Sparse Index : An index record appears for only some of the search-key values. To locate a record, we find the index entry with the largest search-key value that is less than or equal to the search-key value for which we are looking. We start at the record pointed to by that index entry, and follow the pointers in the file until we find the desired record.

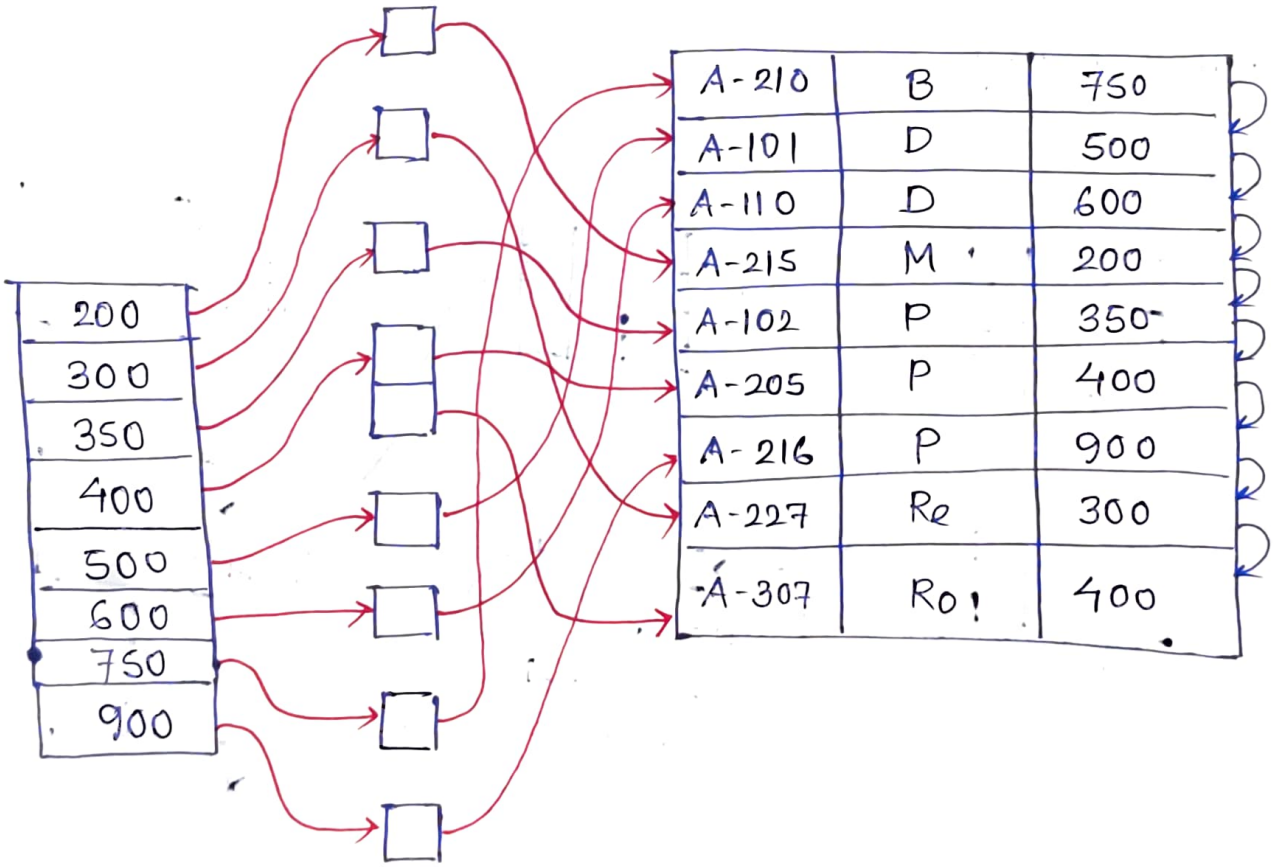


Multilevel Indices



Secondary Indices : A secondary index on a candidate key looks just like a dense primary index, except that the records pointed to by successive values in the index are not stored sequentially.

** In contrast, if the search key of a secondary index is not a candidate key, it is not enough to point to the just first record with each search key value. Therefore the secondary index must contain pointers to all the records.



Secondary index on non-candidate key