Optimization of join operation

- Merge scan: order the relation R and S on disk in ahead, then we can compare their tuples in order, and both relation only need to scan one time. If R and S have not ordered in ahead, must consider the ordering cost to see if it is worth to use this method (p122)
- Using index or hash to look for mapping tuples: in nested loop method, if there is suitable access route on I (say B+ tree index), it can be used to substitute sequence scan. It is best when there is cluster index or hash on join attributes.
- Hash join: because the join attributes of R and S have the same domain, R and S can be hashed into the same hash file using the same hash function, then R ⋈ S can be computed based on the hash file.

4.5 Recovery

4.5.1 Introduction

The main roles of recovery mechanism in DBMS are:

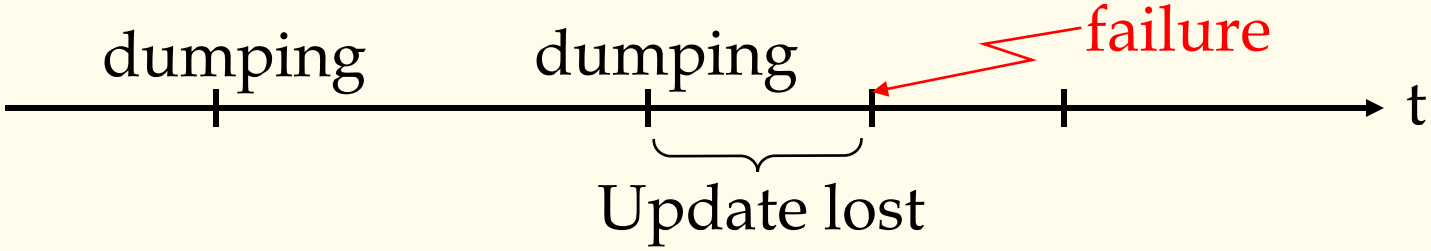
(1) Reducing the likelihood of failures (prevention)

(2) Recover from failures (solving)

Restore DB to a consistent state after some failures.

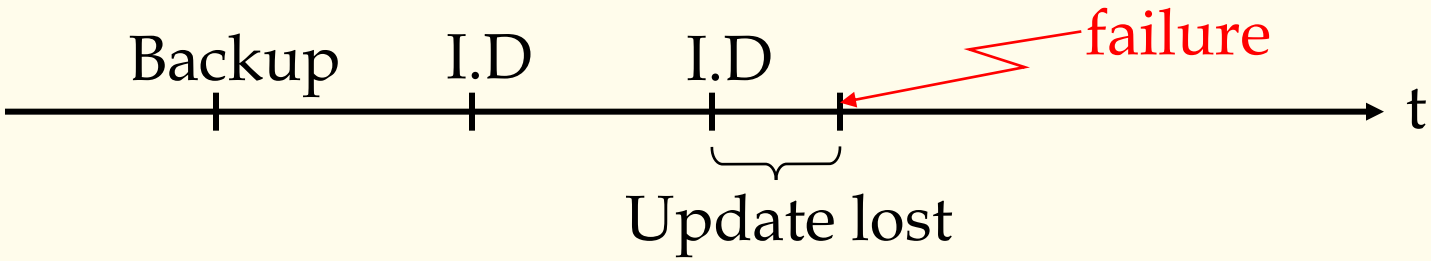
- Redundancy is necessary.
- Should inspect all possible failures.
- General method:

1) Periodical dumping



- Variation : Backup + Incremental dumping

  I.D --- updated parts of DB



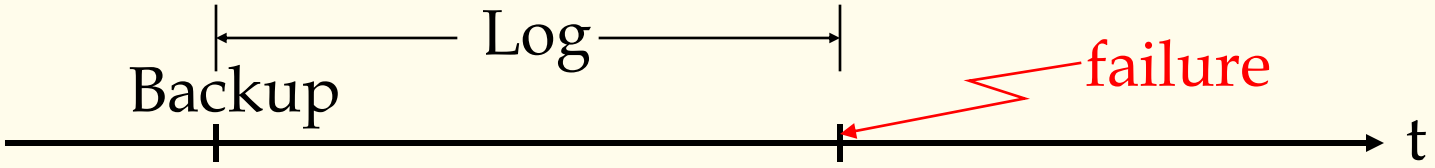
This method is easy to be implemented and the overhead is low, but the update maybe lost after failure occurring. So it is often used in file system or small DBMS.

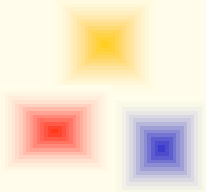
2) Backup + Log

Log : record of all changes on DB since the last backup copy was made.

Change: { Old value (before image --- B.I) / New value (after image --- A.I) } Recorded into Log

For	update op.	: B.I	A.I
	insert op.	: ----	A.I
	delete op.	: B.I	----





While recovering:

- Some transactions maybe half done, should undo them with B.I recorded in Log.
- Some transactions have finished but the results have not been written into DB in time, should redo them with A.I recorded in Log. (finish writing into DB)

  It is possible to recover DB to the most recent consistent state with Log.