Ajay Kumar Garg engineering College, Ghaziabad

Department of CSE

PUT Solution-ODD Semester (2021-22)

Course	:	B.Tech
Subject code	:	KCS-501
Subject Name	:	Database Management System
Semester	:	V
Written By	:	Ms. Arti Pandey
Reviewed By	:	Mamta Bhusry
HOD signature	:	Dr. Sunita Yadav 07/01/2022

PUT Solution

(Section-A)

Q.1. Define DBMS. Write any two advantages?

Ans: DBMS stands for Database Management System. It is software designed to store, retrieve, define and manage data in a database.

Advantages:—

(1) Improved data sharing.
(2) Reduced data redundancy.

Q.2. Differentiate between weak entity & strong entity set.

Ans- (1) A weak entity is dependent on a strong entity to ensure its existence whereas a strong entity is not dependent on any other entity.

(2) A strong entity always a primary key whereas weak entity does not have any primary key.

Strong entity

weak ~~strong~~ entity
(Double Rectangle)

Q.3. Explain 2 DDL & 2 DML commands in brief.

Ans: DDL Commands—

(i) CREATE:— It is used to create the database or its objects like table.

(iii) DROP :- It is used to delete objects from the database.

DML Commands —

(i) INSERT :- It is used to insert data into a table.

(ii) DELETE :- It is used to delete records from a database table.

Q.4. Explain refrential integrity constraints?

Ans. A refrential integrity constraint is specified between two tables. If a foreign key in Table1 refers to the P.K of Table2, then every value of the foreign key in Table1 must be null or be available in Table2.

Q.5. Define the term Join dependency.

Ans. Join dependency is related to 5NF. If the join of R1 & R2 over Q is equal to relation R then we can say that join dependency exists.

Q6. What do you understand by Multivalued dependency?

Ans Multivalued dependency occurs when two attributes in a table are independent of each other but, both depend on a third attribute. It is related to 4NF.

Q.7. List ACID properties and define any one.

Ans. A → Atomicity
C → Consistency
I → Isolation
D → Durability

Atomicity:- It means 'All or None'. It means a transaction completed sucessfuly or it is not.

Q.8. Differentiate serial and concurrent schedule.

Ans. Serial schedule - It is a type of schedule where one transaction is executed before starting another transaction.

Concurrent Schedule - It is a schedule in which several transaction execute concurrently.

Q.9. Define Concurrency Control?

Ans. It is a process of managing the concurrent execution of transaction without conflicts between each other.

Q.10. Define Exclusive lock?

Ans. It is a lock in which data item can be read as well as written. It can be owned by only one transaction at a time. It is denoted as Lock - X

Section - B

Q.11. What is data abstraction ? How the Data abstraction is achieved in DBMS ?

Ans. Database systems are made-up of complex data structures. To ease the user interaction with database, the developers hide internal irrelevant details from users. This process of hiding irrelevant details from user is called data abstraction.

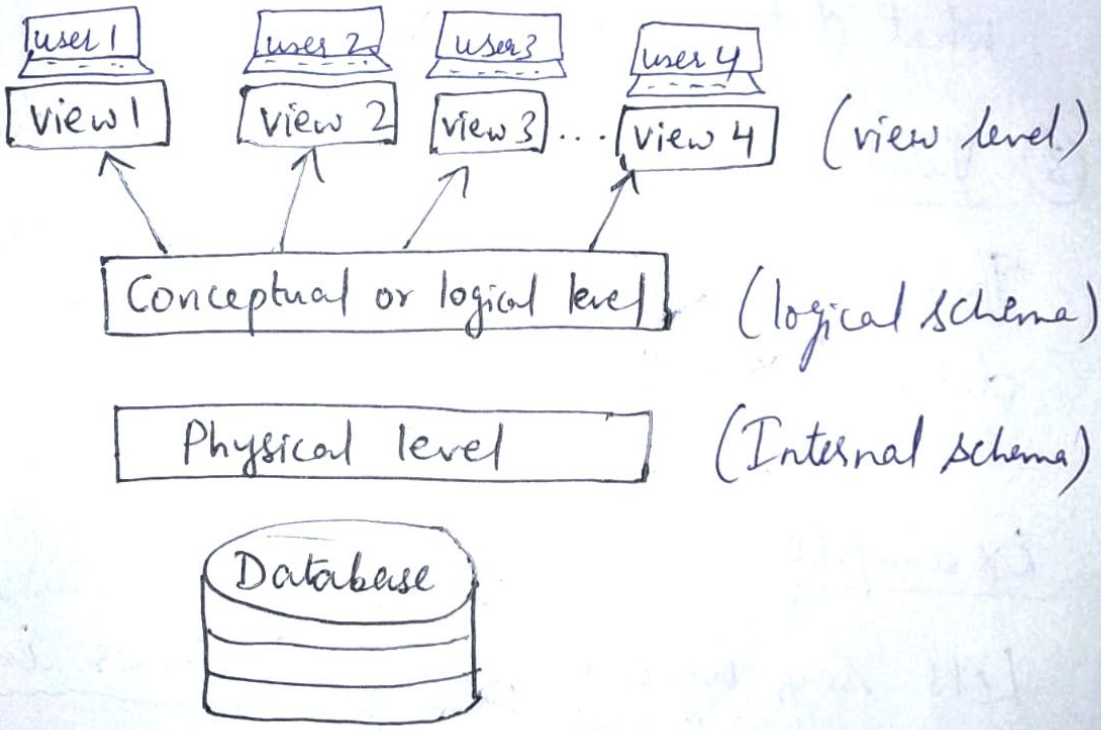


Figure : Three levels of Data Abstraction

There are 3 levels of abstraction:

(1) Physical level — This is the lowest level of data abstraction. It describes how data is actually stored in database. Here complex data structures are used.

(2) Conceptual or logical level: — This is the middle level of 3-level data abstraction architecture. It describes what data is stored in database.

(3) View level — Highest level of data abstraction. This level describes the user interaction with database system.

Example

Let's say we are storing Customer information in a Customer table.

At physical level these records can be described as blocks of storage (bytes, gigabytes, terabytes etc.) in memory. These details are hidden from the programmers.

At logical level these records can be described as fields and attributes along with their data types, their relationship among each other can be logically implemented. The programmers generally work at this level because they are aware of such things about database systems.

At view level, users just interact with system with the help of GUI and enter the details at the screen, they are not aware of how the data is stored and what data is stored; such details are hidden from them.

Q.12. Conside the following, relational database. Give an expression in SQL for each of the following queries. Underlined attributes are primary key

Employee (person_name, street, city)

Works (person_name, company_name, salary)

Company (company_name, city)

Manages (person_name, manager_name)

i) Find the name of all employees who work for the SBI bank.

Ans. Select person_name from Works where company_name = 'SBI';

ii) Find the name of all employees who live in the same city and on same street as do their manager.

Ans.

```
select e.person_name
from Employee e
Join manages m ON m.person_name = e.person_name
where Exists (
    Select 1 from Employee where person_name =
    m.manager_name and street = e.street
    and city = e.city);
```

(iii) Find the name, street, and city of all employees who work for SBI bank and earn more than Rs. 8000 per annum.

Ans.
```
Select e.person_name, e.street, e.city
from Employee e
Join Works w ON e.person_name = w.person_name
where w.company_name = 'SBI' and w.salary > 8000 ;
```

(iv) Find all employee's name who earn more than every employee of Kotak Bank.

Ans.
```
Select person_name from Works
Where salary > (select max(salary) from Works
where company_name = 'Kotak' ;
```

(V) Give all Employees of SBI bank a 8% salary raise

Ans.
```
Update Works
Set salary = salary * 1.08
where company_name = 'SBI' ;
```

(Vi) Delete all tuples in the works relation for employee of SBI bank.

Ans:
```
delete from Works
where Company_name = 'SBI';
```

(vii) Find the name of all employees in this database who live in the same city as the company for which they work.

Ans:
```
Select E.person_name
from Employee as E, Works as W, Company as C
where E.person_name = W.person_name and
E.city = C.city and W.company_name = C.company_name;
```

Q.13. Define the term decomposition and explain its types with example. Consider a relation schema R(A, B, C, D) with following FDs:

$A \rightarrow B$
$B \rightarrow C$
$C \rightarrow D$
$D \rightarrow B$

Determine whether the decomposition of R into R1(A, B), R2(B, C), R3(B, D) is lossless or lossy.

Ans: Decomposition is the process of breaking down in parts or elements. It replaces a relation with a collection of smaller relations. It breaks tables into multiple tables in a relation.

The decomposition should be lossless because it confirms that the information in the original relation can be accurately reconstructed based on decomposed relation. If there is no proper decomposition of the relation, then it may lead to problems like loss of information.

Properties of Decomposition

1. lossless Decomposition
2. Dependency Preservation
3. lack of Data Redundancy

Types of Decomposition

1. **lossless Decomposition**: It means the information should not get lost from the relation that is decomposed. It gives a guarantee that the join will result in the same relation as it was decomposed.

for example: <EmpInfo>

Emp_Id	Emp_Name	Emp_Age	Emp_Location	Dept_ID	Dept_Name
E001	Jacob	29	Alabama	Dpt1	Operations
E002	Henry	32	Alabama	Dpt2	HR
E003	Tom	22	Texas	Dpt3	Finance

Decompose the table <EmpInfo> into

<EmpDetails>

Emp_Id	Emp_Name	Emp_Age	Emp_location
E001	Jacob	29	Alabama
E002	Henry	32	Alabama
E003	Tom	22	Texas

<DeptDetails>

Dept_ID	Emp_Id	Dept_Name
Dpt1	E001	Operations
Dpt2	E002	HR
Dpt3	E003	Finance

Now Natural Join of the above 2 tables:-

<Empdetails> Join <DeptDetails>

Emp_ID	Emp_Name	Emp_Age	Emp_location	Dept_ID	Dept_Name
E001	Jacob	29	Alabama	Dpt1	Operations
E002	Henry	32	Alabama	Dpt2	HR
E003	Tom	22	Texas	Dpt3	Finance

Therefore, the above relation had lossless decomposition

i.e. no loss of information.

② lossy Decomposition : Here when the relation is decomposed into 2 or more relation schemas, the loss of information is unavoidable when original relation is retrieved.

for example :

<EmpInfo>

Emp_Id	Emp_Name	Emp_Age	Emp_location	Dept_ID	Dept_Name
E001	Jacob	29	Alabama	Dpt 1	Operations
E002	Henry	32	Alabama	Dpt 2	HR
E003	Tom	22	Texas	Dpt 3	Finance

Decompose the above table into 2 tables :-

<Empdetails>

Emp_Id	Emp_Name	Emp_Age	Emp_location
E001	Jacob	29	Alabama
E002	Henry	32	Alabama
E003	Tom	22	Texas

<Dept details>

Dept_Id	Dept_Name
Dpt 1	Operations
Dpt 2	HR
Dpt 3	Finance

Here, we cannot join <Empdetails> and <Deptdetails> tables as they donot have common attribute. So this is lossy decomposition.

Q.14. Define BCNF. How it is different from 3rd NF Find Normal form of relation R(A, B, C, D, E) having FD set F = {A → B, BC → E, ED → A}, Decompose upto BCNF.

Ans: BCNF stands for Boyce-Codd Normal Form and was made by R.f Boyce and E.f Codd in 1974. A functional Dependency is in BCNF if these properties hold:

(1) It should already be in 3NF.

(2) for a FD say P → Q, P should be a superkey.

Difference b/w 3rd NF and BCNF

3rd NF	BCNF
(1) In 3rd NF there should be no transitive dependency that is no non prime attribute should be transitively dependent on the candidate key.	(1) In BCNF, for any relation A → B A should be a superkey of relation.
(2) It is less stronger than BCNF	(2) It is comparatively more stronger than 3NF.
(3) Here FD such as Prime or Non Prime → Prime Attribute is allowed.	(3) Here FD such as Prime or Non-Prime Attribute → Prime Attribute is Not allowed.

Numerical.

R(A, B, C, D)

FDs: $A \rightarrow B$
$B \rightarrow C$
$C \rightarrow D$
$D \rightarrow B$

R is decomposed in R1(A, B), R2(B, C), R3(B, D)

For relations R1, R2, R3 ∧ they have to be lossless decomposition to satisfy 3 conditions.

(1) The union of all the attributes of R1, R2, R3 is same as original Relation R. Here this is true.

(2) R1 and R2 relations have common attribute B. Similarly R3 also has B as common attribute with R1 ∩ R2. So this condition is also true as ~~B is attribute~~ B is common attribute present in the sub relation R1, R2, R3.

(3) In R1(A, B), B is not a key
but in R2(B, C) B is a key
so R1 ⋈ R2 is lossless.

Now when we further join (R1 ⋈ R2) with R3 as R1 ⋈ R2 has attributes (A, B, C) here R3 also has common attribute B with R1 ⋈ R2 Here B is key in relation R3 as $B^+ = \{B, C, D\}$ from above FD. ∴ This is lossless Decomposition.

3NF	BCNF
(4) The redundancy is high in 3NF	(4) The redundancy is comparatively less in BCNF
(5) Here all FDs are prescored	(5) Here some of the FDs may be sacrificed.
(6) It is comparatively easier to achieve	(6) It is difficult to achieve

Numerical

FDs: $F = \{ A \rightarrow B$

$BC \rightarrow E$

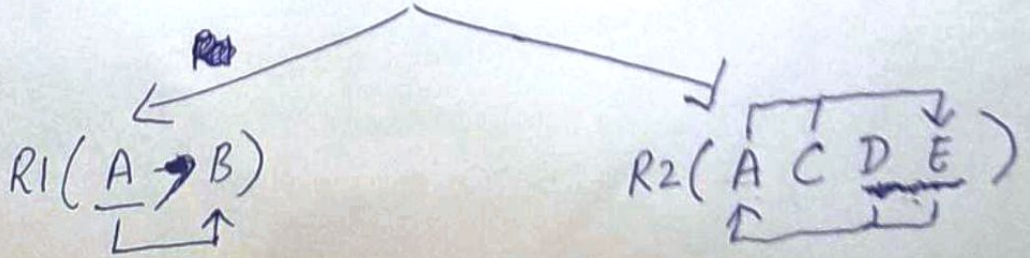
$ED \rightarrow A \}$

Here the Candidate keys are ACD, BCD, ECD

So all FDs are prime Attribute → prime Attribute

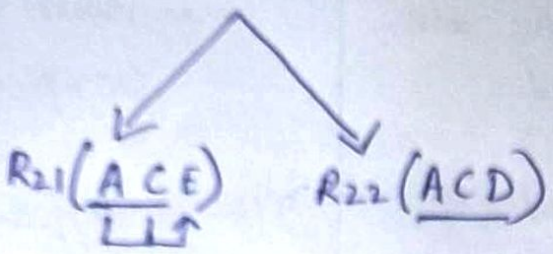
So all FDs satisfy 3NF. But donot satisfy BCNF

So decompose Relation R(ABCDE)

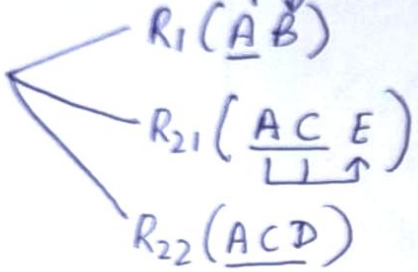


Now R_1 is in BCNF.

But R_2 is still Not in BCNF as both FDs in R_2 violate BCNF so decompose $R_2(ACDE)$



∴ Relation $R(ABCDE)$



Now all relations are in BCNF.

Q.15. Consider following schema:

Suppliers(sid, sname, address)

Parts(pid, pname, colour)

Catalog(sid, pid, cost)

Write Relational Algebra queries for following:—

Q.15 Supplier (sid, sname, address) Catalog (sid, pid, cost)
Parts (pid, pname, colour)

a) find the name of suppliers who supply some blue parts.

Ans: $R1 = \Pi_{pid}(\sigma_{color='blue'}(Parts))$

$R2 = \Pi_{sid}(R1 \bowtie Catalog)$

$R3 = \Pi_{sname}(R2 \bowtie Suppliers)$

b) Find the sid of suppliers who supply some red or green parts.

Ans: $R1 = \Pi_{pid}(\sigma_{color='red' \text{ or } color='green'}(Parts))$

$R2 = \Pi_{sid}(R1 \bowtie Catalog)$

c) Find sid of suppliers who supply some red part or are at sector 128 Noida.

Ans: $R1 = \Pi_{pid}(\sigma_{color='red'}(Parts))$

$R2 = \Pi_{sid}(R1 \bowtie Catalog)$

$R3 = \Pi_{sid}(\sigma_{address='sector\ 128\ Noida'}(suppliers))$

$Ans = R2 \cup R3$

d) Find the sid of supplier who supply every part.

Ans.

$$R1 = \Pi_{sid, pid}(Catalog)$$

$$R2 = \Pi_{pid}(Parts)$$

$$Ans = R1 / R2$$

e) Find the sid of supplier who supply every red or green part.

Ans.

$$R1 = \Pi_{sid, pid}(Catalog)$$

$$R2 = \Pi_{pid}(\sigma_{color='red' \text{ or } color='green'}(Parts))$$

$$Ans = R1 \div R2$$

Section-C

Q.16 (a). What is Conflict Serializable Schedule? Check the given schedule S1 is Conflict serializable and view serializable or not?

S1: R1(x), R2(x), R2(y), W2(y), R1(y), W1(x)

Ans. :— Conflict serializable schedule —

If one Non-serial schedule can be converted into serial schedule by swapping of Non-conflicting Instructions of 2 different transactions in a schedule. Then this Non-serial schedule is called Conflict serializable (and hence it is consistent).

I_i and I_j are conflict Instructions if they satisfy: ——

(1) If they are operations by different transactions on same data items.

(2) At least one of these instruction is write operation.

★ Conflict Instructions can not be swapped.

★ Non-Conflict Instructions can be swapped.

Example of Conflict Instruction

①

T_1	T_2
R(A)	
	W(A)

②

T_1	T_2
W(A)	
	W(A)

③

T_1	T_2
W(A)	
	R(A)

Here, T_1, T_2 are Transactions

R(A) is Read Instruction on data item A

& W(A) is Write Instruction on data item A

Example of Non-conflicting Instruction

①

T_1	T_2
R(A)	
	W(B)

②

T_1	T_2
R(A)	
	R(A)

b.

Numerical : Prove whether S1 is conflict serializable or Not?

Schedule S1

T_1	T_2
$R_1(x)$	
	$R_2(x)$
	$R_2(y)$
	$W_2(y)$
$R_1(y)$	
$W_1(x)$	

(1) Make a Precedence Graph

(2) The Instruction $R_1(x)$ is Non-conflicting Instruction. Do Nothing in Precedence Graph.

(3) $R_2(x)$ is conflicting with $W_1(x)$. So make a directed edge from transaction T_2 to T_1.

(4) Then Next $R_2(y)$ is Non-conflicting Instructions. So do nothing.

(5) Then Next $W_2(y)$ is conflicting Instruction with $R_1(y)$. We already have directed edge from T_2 to T_1 in Precedence Graph.

(6) Then Next R1(y) is Non-conflicting. So do nothing.

(7) Then Next W1(x) is Non-conflicting. So do nothing.

∴ final Precedence Graph is

As there is No cycle in Precedence Graph. So the Schedule S1 is Conflict Serializable. And order of serializability is

$T_2 \longrightarrow T_1$

As schedule S1 is conflict serialiable. So 100% the schedule S1 is View Serializable also.

Q.16 (b) Discuss about Deadlock Handling with example.

Ans. Deadlock is a condition where two or more transactions are waiting indefinitely for one and another to give up locks.

for example – In student table(S), transaction T1 holds a lock on some rows and needs to update some rows in Grade table(G).

Simultaneously, transaction T_2 holds lock on some rows in Grade Table(G) and needs to update rows in Student table held by Transaction T1.

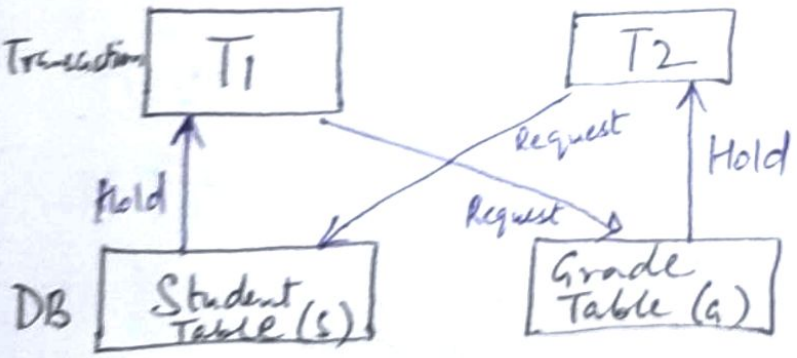


Figure: Deadlock in DBMS

T1	T2
lock-X on S	
Write(S)	
	lock-X on G
	Write(G)
	Wait for lock-X on S
Wait for lock-X on G	

Here lock-X is Exclusive lock

Deadlock Avoidance

(1) When a database is stuck in deadlock state, then it is better to avoid the database rather than aborting or restarting the database. This is waste of time and resources.

(2) Deadlock avoidance mechanism is used to detect any deadlock situation in advance. A method like 'wait for graph' is used for detecting the deadlock situation for smaller databases. For larger databases, deadlock prevention methods are used- such as (1) Wait-Die Scheme

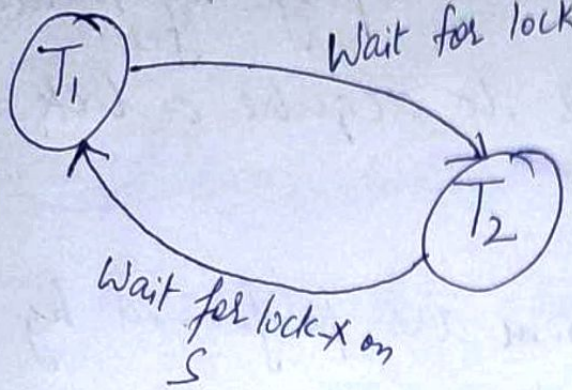
(2) Wound wait Scheme

Deadlock Detection

↳ Here, when a transaction waits indefinitely for resources, then DBMS should detect whether the transaction is involved in deadlock or not.

The lock Manager maintains Wait-for Graph to detect deadlock cycle in Database.

Wait-for Graph for above example.



So lock Manager would not allow a cycle to be formed. If the cycle is formed. Then deadlock is there.

Deadlock Prevention

- This is suitable for large databases.
- If resources are allocated in such a way that deadlock never occurs, then deadlock can be prevented.
- DBMS analyzes the operations of the transaction whether they can create a deadlock situation or not. If they do, then the DBMS never allow such transactions to be executed.

Deadlock Prevention Techniques
- Wait-Die scheme
- Wound wait scheme

Wait-Die scheme

Here TS(T) is time stamp of any transaction T.

Suppose T_i attempts to acquire a lock already held by T_j:

Then following actions are performed by DBMS –

(1) Check if $TS(T_i) < TS(T_j)$ – If T_i is older transaction and T_j has held some resources, then T_i is allowed to wait until dataitem is available for execution. So older transaction T_i waits for younger transaction T_j to free the resources.

(2) Check if $TS(T_i) < TS(T_j)$ – If T_i is older transaction & has held some resources & if T_j is waiting for it, then T_j is killed and restarted later with random delay but with the same timestamp.

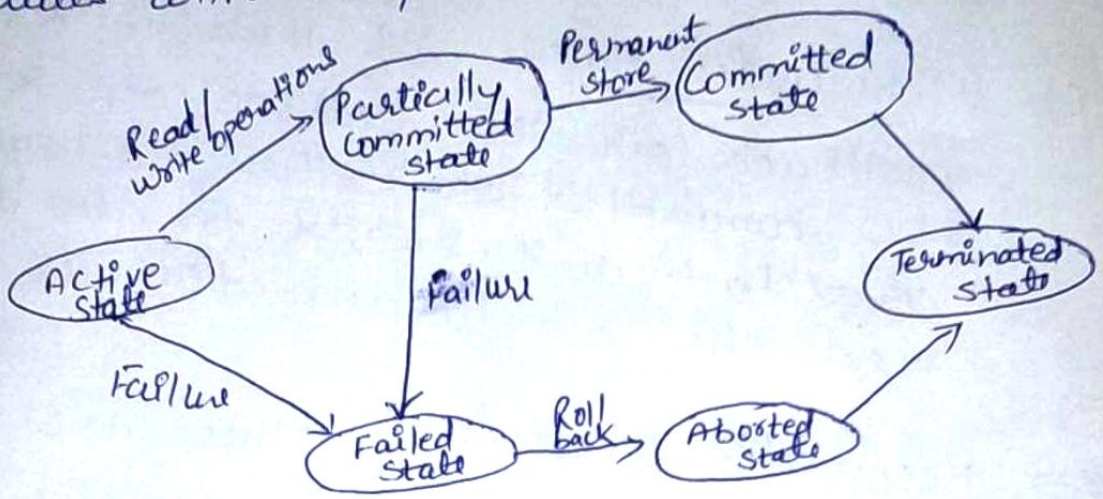
Wound-Wait scheme –

↳ Here if older transaction requests for resource held by younger transaction. Then older transaction forces younger transaction to wound & release the resources. After the minute delay, the younger transaction is restarted with the same timestamp.

↳ Here if older transaction has held some resources required by younger transaction. Then younger transaction is asked to wait for older to release it.

8.17.(a) Explain the state diagram of transaction in detail. Explain recoverable and cascadeless schedules with example.

Ans:



Transaction states

There are different types of Transaction States :-

(1) Active State - When the instructions of the transaction are running then the transaction is in active state.

(2) Partially Committed - After completion of all operations. If the changes are made in memory or local buffer.

(3) Failed state :- When any instruction of transaction fails, it goes to Failed state

(4) Aborted state :- After having any failure the transaction goes from failed state to "aborted state".

(5) Committed State :- It is the state when the changes are made permanent on the database and transaction is complete.

(6) Terminated State :- If there is not any rollback or the transactions comes from the committed state then the system is consistent and ready for new transaction and the old transaction is terminated.

b ⇒ Recoverable Schedule :- There are the schedules in which transaction commit only after all transactions whose changes they read commit.

example:

T_1	T_2
R(A)	
W(A)	
	W(A)
	R(A)
commit	
	commit

⇒ Cascadeless schedule :- Schedules in which transactions read values only after all transactions whose changes they are going to read commit are called Cascadeless Schedules.

→ Avoids all possible dirty Read problem
→ Allows only committed read operations & uncommitted write operations

example:

T_1	T_2	T_3	T_4
R(A)			
W(A)			
committed	R(A)		
	W(A)	R(A)	
	committed	W(A)	R(A)
		committed	W(A)

(b) What is log? How it is maintained? Discuss the features of deferred modification & Immediate modification in brief.

Ans: Log:- A log is a sequence of records, recordings all the update activities in the database. In a stable storage, logs for each transactions are maintained.

If any operation is performed in the database, then it will be recorded in the log. But the process of storing the logs should be done before actual transaction is applied in the database.

The foll. logs are written for the transaction-

(1) When the transaction is initiated, then it writes 'start' log.

$< T_n, \text{Start} >$

(2) When the transaction modifies, then another log is written-

$< T_n, \text{Field Name}, \text{Old Value}, \text{New Value} >$

(3) When transaction finished, then –
< Tn, Commit >.

⇒ Deferred modification :- This technique occurs if the transaction does not modify the database until it has commited. In this method, all logs are created & stored in the stable storage and the d/b updated when a transaction commits.

⇒ Immediate modification:- This technique occurs if database modification occurs while the tansaction is still active. The d/b is modified immediately after every operation. It follows actual d/b modification

Q.18. (a) Explain Time stamp based protocol for concurrency control.

Ans. Time-stamp based Protocol :- It is a protocol which uses system time or logical counter as a Timestamp to serialize the execution of concurrent transactions. This protocol ensures that every conflicting read and write operations are executed in a timestamp order.

⇒ The older transactions is always given

priority in this method. It uses system time to determine the time stamp of the transaction. This is the most commonly used concurrency protocol.

→ Time-stamp based protocols manage conflicts as soon as operation is created.

Advantages:-

(i) Schedules are serializable just like 2PL protocols.

(ii) No waiting for the transaction, which eliminates the possibility of deadlocks.

Disadvantages:-

(i) Starvation is possible if the same transaction is restarted & continually aborted.

(b) Explain Multi version schemes of concurrency control in detail.

Ans: Multi version Concurrency Control provides concurrent access to the database without locking the data. This feature improves the performance of database applications in a multiuser environment. Applications will no longer hang because a read cannot acquire a lock.

Multi-version schemes keep old version of data item to increase concurrency.

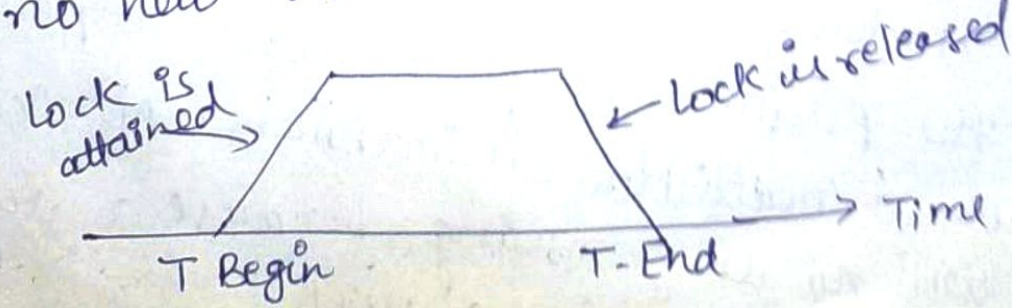
→ Multiversion 2 Phase locking :- Each successful write results in the creation of new version of the data item written. Timestamps are used to label the versions. When a read(X) operation is issued, select an appropriate version of X based on the timestamp of transaction.

Q.19. (a) What is 2PL? Describe with the help of example.

Ans: 2PL :- A transaction is said to follow the Two-Phase locking protocol if locking and unlocking can be done in two phases :-

(i) Growing Phase :- In this phase, new locks on data items may be acquired but none can be released.

(ii) Shrinking Phase :- In this phase, the existing locks may be released but no new locks can be acquired.



Example:

	T_1	T_2
P.	lock-S(A)	
1.		lock-S(A)
2.	lock X(B)	—
3.	—	—
4.	unlock(A)	
5.		lock-X(C)
6.	unlock(B)	
7.		unlock(A)
8.		unlock(C)
9.	—	—

Growing Phase (P, 1, 2, 3) — Lock point (3) — Shrinking Phase (4, 5, 6, 7, 8)

T_2: Growing phase (1, 2, 3, 4, 5); Shrinking phase (6, 7, 8)

(b) What do you mean by Multiple Granularity? How it is implemented in transaction system.

Ans:- Multiple Granularity:-

→ It can be defined as hierarchically breaking up the database into blocks which can be locked.

→ The Multiple Granularity protocole enhances concurrency & reduces lock overhead.

→ It maintains the track of what to lock & how to lock.

→ This type of hierarchy can be graphically

represented as a tree. The levels of the tree starting from the top level are as follows :-

(1) Database :- It is the higher level, shows the entire database.

(2) Area :- The second level represents a node of type area. The higher level database consists of exactly these areas.

(3) File :- The third level is File. The area consists of children nodes are known as Files. No file can be present in more than one area.

(4) Record :- Each file contains child nodes known as records. The files has exactly those records that are its child nodes

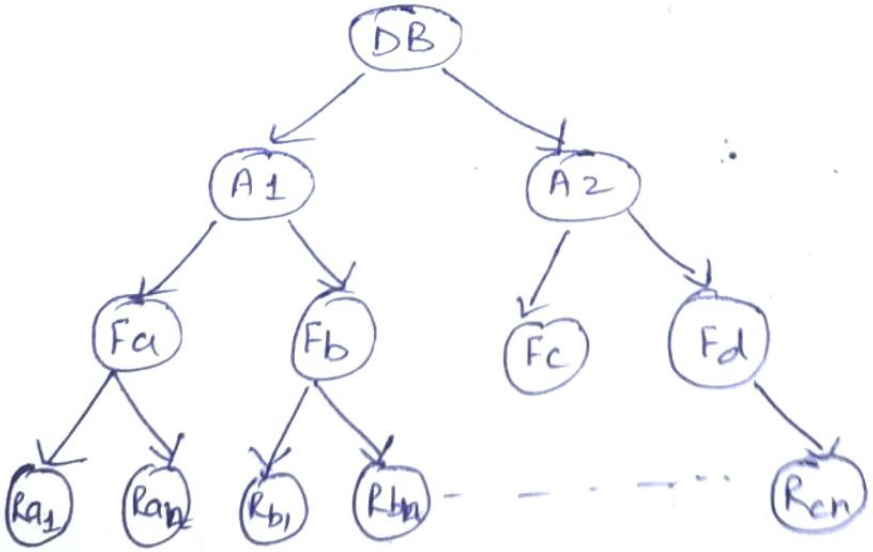
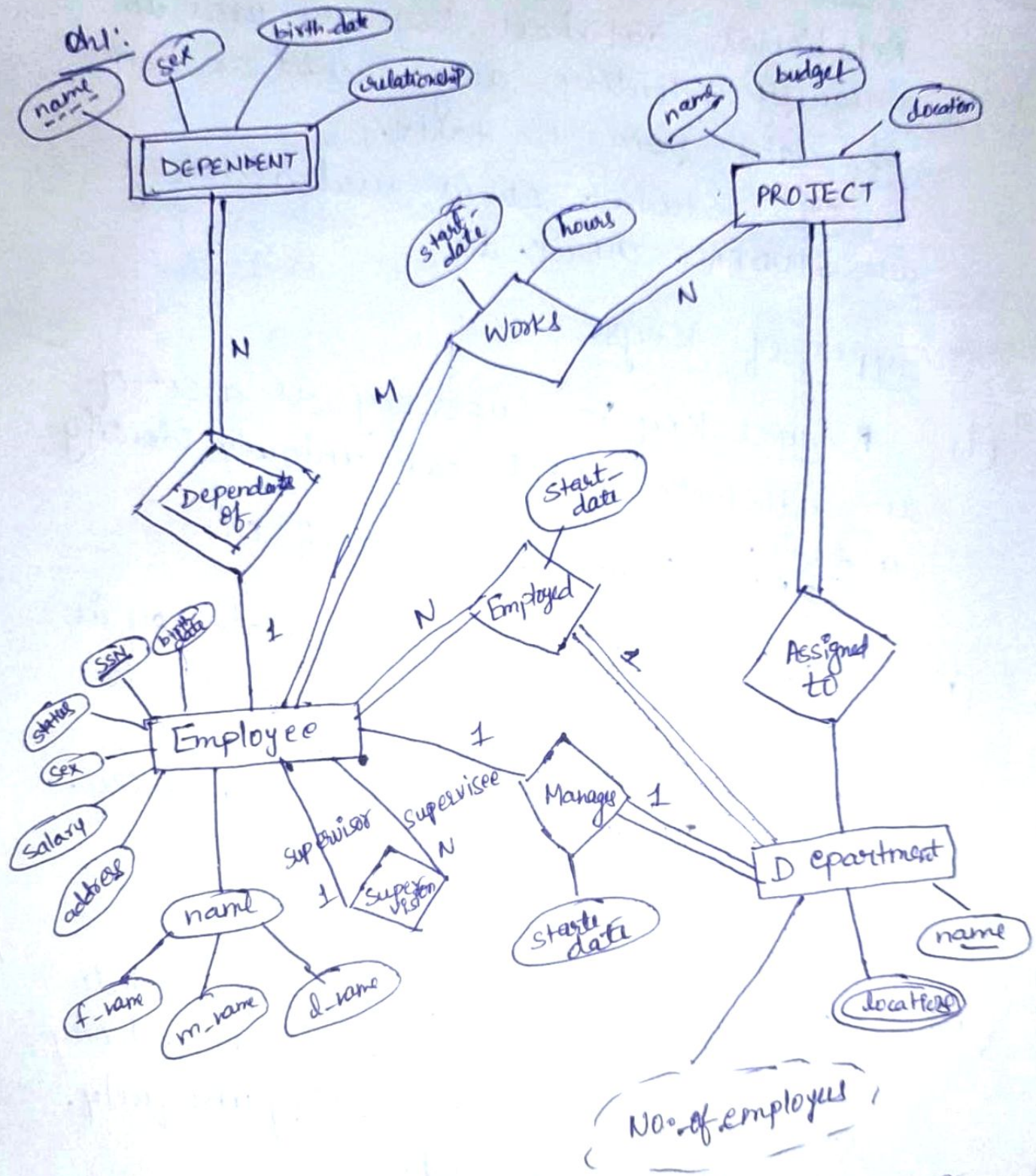


Fig :- Multi Granularity Tree Hierarchy.

Q.20. (a) Draw the E-R diagram for small marketing company database, assuming your own data requirements.

Ans:



(b) Explain the concept of Keys in detail with example.

Ans: **Keys:-** Keys play an important role in Relational database. Keys are used to uniquely identify any record or row of data from the table.

e.g: Student_ID is used as a key in STUDENT TABLE.

Types of Keys:-

(i) **Super Key:-** Super key is a set of an attribute which can uniquely identify a tuple.

ex:- {Emp_Id, Emp_Name} etc.

(ii) **Candidate Key:-** A candidate key is an attribute or set of an attribute which can uniquely identify the tuple. It is a subset of Super Key.

e.g:- {Passport_NO}, License NO.

(iii) **Primary key:-** It is a key used to identify one and only one instance of an entity uniquely.

e.g: Emp_Id.

(iv) Alternate key:- These are the keys ~~remaining~~ are those candidate keys that were not chosen to be Primary key of table.

e.g: Emp_SSN.

(v) Foreign key:- A foreign key is the one that is used to link two tables together via the primary key.

e.g: In the Employee table, Dept_ID is the foreign key and both the tables are related.