End Semester Examination (5th sem), 2021

→ Subject Name :→ Database Management Systems.

→ Subject Code :→ IT 3103.

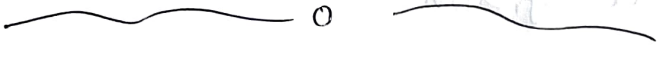
→ Date of Examination :→ 06.12.2021

→ Name :→ Aniket Majee.

→ Examination Roll Number :→ 510819019.

→ G suite ID :→ 510819019.aniket@students.iiests.ac.in

→ Number of sheets uploaded :→ 14



(1.) Data dictionary :→

A data dictionary is a collection of names, definitions and attributes about the data elements in the database that is it contains the meta data. It is a very important as it contains information such as what is in the database, who is allowed to access it and where the datas are actually stored.

→ It is handled by the database administrators.

☒ Database Schema :→ A database schema gives the description of the database. This can be considered as a blueprint of the database and gives a list of fields in the database with their data types. It is designed during the database design and not expected to change it frequently.

☒ Database state :→ A database state provides the present state of the database and its data that means the data in a database at a particular moment of time is know its current database state. It is changed via some kind of insertion, deletion and modification in the database.

Q Difference between three level of data abstraction by using a two dimentional array of size m×n :-

A → The grid is of 2 dimentional array of size m×n.

Now,

(i) The Physical level would simply be m×n (Probably consecutive.) storage locations of whatever size is specified by the implementation.

(ii) The conceptual level is the grid of boxes, each possibly containing an integer, which is n boxes high by m boxes wide.

(iii) There are $2^{m \times n}$ possible views, that is,

→ A view might be the entire array
→ A view might be a particular row of the array.
→ or it may be all n rows but only columns 1 through i.

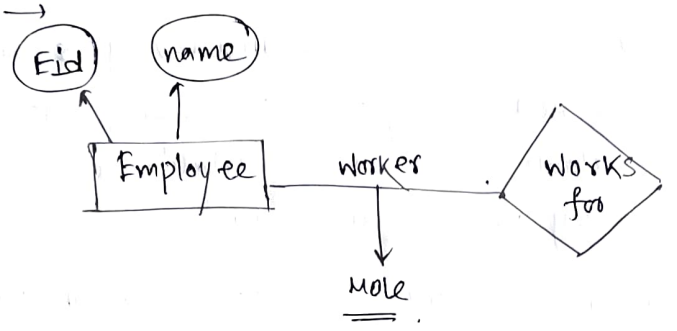
(2.) R

Need of Role name in E-R diagram :→

Role in the E-R diagrams is the function that an entity plays in an relationship. Roles are generally explicit and not specified.

In E-R diagram roles are very useful when the meaning of a relationship set needs a clarification.

Example :→



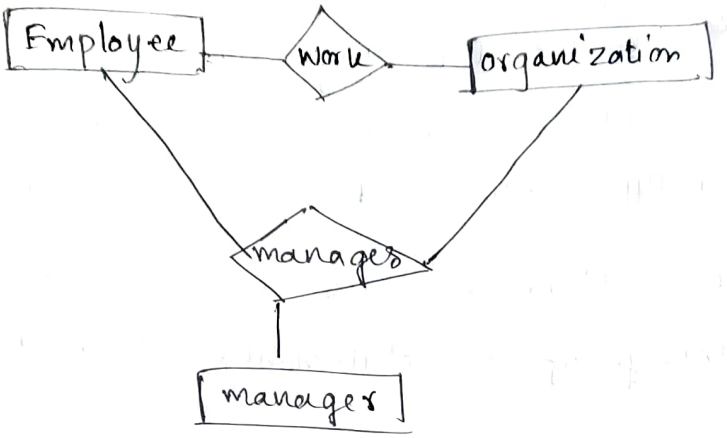
Aggregation :→

In EER (Enhanced ER Model) the Aggregation is an abstraction through which relationships are treated as higher level entities.

~~Let's consider a binary relationship as given below between Employee between~~

Let us consider a binary relationship work, between a Employee and a organization. Now suppose there is a manger also which manages both the employee as well as the organization. In order to make an relationship between them we need to create

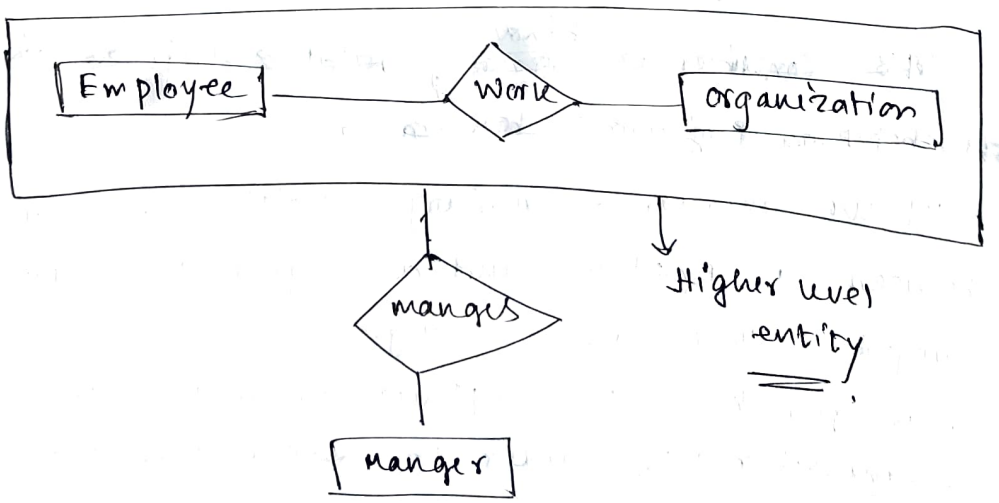
a 3 ary relationship in ER diagram.



But, the problem here is the redundency. ~~& So we~~
To avoid this redundency, we can see if we could somehow relate the two relationships then the redundency can be avoided.

To do so, we use the concept of Aggregation.
We consider the binary relationship work, Employee, organization as a higher level entity and describe a relationship of ~~a~~ relationship between the manages and ~~the~~ ~~output~~ the entity.

Now the EER diagram looks like,



☐ Duplicate tuples are not considered in a relation because they create redundency of data inside a database which slows down the data processing, querying, inserting, deleting and other database related operations.

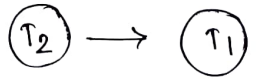
(5.)

Transactions:→

T_1:	T_2:
read(X);	read(X).
X = X + 10;	
write(X)	

→ (i) There is only one conflict between read(X) in T_2 and write(X) in T_1, or an edge from T_2 towards T_1 will be there.

The corresponding Precedence directed graph is,



As we can see, there is no cycle in the graph, so Transaction T_1 and T_2 are conflict equivalent. and the order of the conflict serializability is,

S : T_2, T_1

So, the concurrent execution of T_1 and T_2 produces a serializability with the order, T_2, T_1.

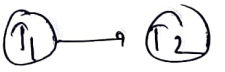
☒ Strict schedule :→ A strict schedule is a one in which if transaction T_1 reads from transaction T_2, then T_1 must write or read or modify only if the T_2 commits.

Example,

T_1	T_2
r(x)	
	r(x)
	x = x + 5
x = 20;	
w(x)	w(x)
	commit;

The above schedule is serializable as there is a conflict from r(x) of T_1 and w(x) in T_2

The precedence graph,



so, order, T_1, T_2

But it is not strict as T_1 modifies x before T_2 commits that will result in a loss of data and inconsistency.

☒ Two phase locking decreases the degree of multiprogramming:

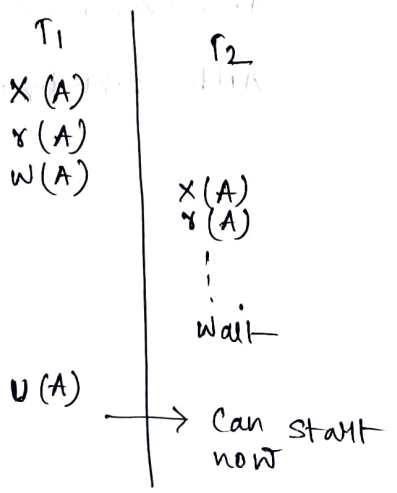
(i) It limits the amount of concurrency

(ii) A transaction cannot release a lock on item X after using it, if it lock another item Y.

(iii) A transaction must lock Y before its needs so that X can be released.

(iv) The other transactions seeking to access to X have to wait.

(v) If Y is locked earlier than it is needed, then other transactions seeking access to it have to wait



This is how it reduces the degree of multiprogram.

(4.)

A spurious tuple is, mainly, a record in a database that gets created while two table are joined badly. In database-ese, spurious tuples are formed while two tables are joined on attributes which are neither primary keys nor foreign keys. ~~To prevent spurious tuples, avoid joining etc~~

☒ To prevent spurious tuples, avoid joining relations that consist of matching attributes that are not Primary key or foreign key combinations as joining on such attributes may generate spurious tuples.

☒ Lets Consider a relations,

R(A, B, C, D, E)

~~Let the~~,

the functional dependencies are,

$AB\,C \longrightarrow D$

$B \longrightarrow E$

$C \longrightarrow D$.

The Candidate Key $\longrightarrow$ ABC

As we can see from the relation there exists partial dependency, ~~between~~ $B \longrightarrow E$, $C \longrightarrow D$.

Now the problems here is,

According to the relation as ABC is the Primary key by using ABC we can generate any attribute means by ~~or~~ knowing the value of ABC any other attributes can be generated.

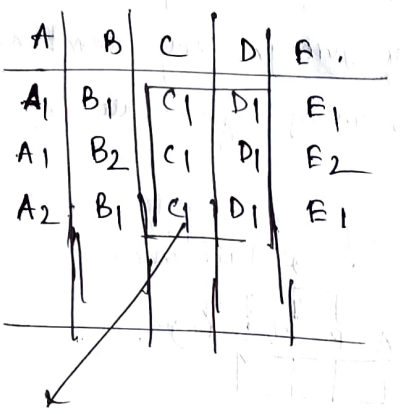
but, in the functional dependey,

We can see that, $C \rightarrow D$

that means C can alone generate D and C is the prime attributes, so for same values of C the values of D will also be same for all time. and this leads to redundeney in database,

like,

A	B	C	D	E
A_1	B_1	C_1	D_1	E_1
A_1	B_2	C_1	D_1	E_2
A_2	B_1	C_1	D_1	E_1



We need to repeat the values all the times.

→ otherwise there are other problems like —

(i) Insert anomalies.

(ii) Delete anomalies.

(iii) update anomalies.

(10)

☒ **Proof :⟶**

Let a, b be the two attributes in a relation R.

The possible functional dependencies are :

Case 1 :⟶ LHS contains both attributes

a, b → ...

Clearly this is a trivial functional dependency because RHS attributes form subset of LHS attributes

Case 2 :⟶ LHS contains only one attributes.

a ⟶ ...

b ⟶ ...

Clearly in this case, the LHS attribute will be Candidate key.

Hence the possible functional dependencies are either trivial or the LHS attributes for a superkey.

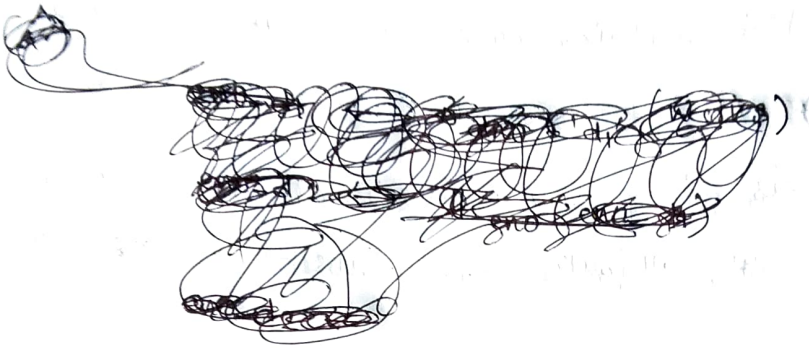
So, the relation having only two attributes will be in BCNF.

(3.)

Employee (eno, ename, age, city, salary)

Works (eno, dno, hr)

Dept (dno, budget, city, manger_eno).



(a.)

$E_1 \leftarrow \sigma_{dno = 'd_1'}(Works)$

$E_1 \leftarrow \pi_{eno}(E_1)$

$E_2 \leftarrow \sigma_{dno = 'd_2'}(Works)$

$E_2 \leftarrow \pi_{eno}(E_2)$

$E \leftarrow E_1 \cap E_2$

$Res \leftarrow \pi_{ename}(Employee \bowtie_{Employee.eno = E.eno} E)$

(b.)

```
Select  ename
from    emp e, dept d
where   e.eno = d.manager_eno
and     emp.salary > ( select AVG (salary)
                       from   emp e2, works w
                       where  e2.eno = w.eno
                       and    d.dno = w.dno
                       group by works.dno);
```

(c.)

$$\{ e.ename \mid Employee(e) \text{ AND } ((\exists d)\ Dept(d) \text{ AND } (d.manager\_eno = e.eno \text{ AND } d.budget > 50{,}000))\}$$