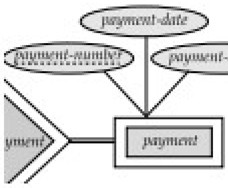
	Part-A
1	State how the weak entity set is mapped in to a table.

2. Tabular Representation of Weak Entity Sets

Let A be a weak entity set with attributes a1, a2, . . . , am. Let B be the strong entity set on which A depends. Let the primary key of B consist of attributes b1, b2, . . . , bn. The entity set A can be represented by a table called A with one column for each attribute of the set:

$$\{a1,a2,...,am\} \cup \{b1,b2,...,bn\}$$



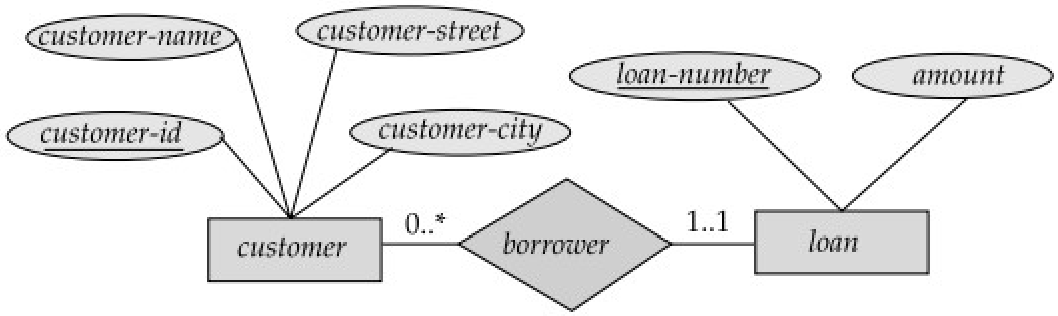
This entity set has three attributes: payment-number, payment-date, and payment-amount. The primary key of the loan entity set, on which payment depends, is loan-number. Thus, we represent payment by a table with four columns labeled loan-number, payment- number, payment-date, and payment-amount, as in Figure

loan-number	*payment-number*	*payment-date*	*payment-amount*
L-11	53	7 June 2001	125
L-14	69	28 May 2001	500
L-15	22	23 May 2001	300
L-16	58	18 June 2001	135
L-17	5	10 May 2001	50
L-17	6	7 June 2001	50
L-17	7	17 June 2001	100
L-23	11	17 May 2001	75
L-93	103	3 June 2001	900
L-93	104	13 June 2001	200

2	State the tabular representation of relationship

3. Tabular Representation of Relationship Sets

Let R be a relationship set, let a1, a2, . . . , am be the set of attributes formed by the union of the primary keys of each of the entity sets participating in R,



customer-id	loan-number
019-28-3746	L-11
019-28-3746	L-23
244-66-8800	L-93
321-12-3123	L-17
335-57-7991	L-16
555-55-5555	L-14
677-89-9011	L-15
963-96-3963	L-17

3	State the tabular representation of Generalization/Specialization.

6. Tabular Representation of Generalization

Create a table for the higher-level entity set. For each lower-level entity set, create a table that includes a column for each of the attributes of that entity set plus a column for each attribute of the primary key of the higher-level entity set. Thus, for the E-R diagram of Figure 2.17, we have three tables:

- account (account-number, balance)
- savings-account (account-number, interest-rate)
- checking-account (account-number, overdraft-amount)

4 What are the pitfalls in the relational database design

* Pitfalls on the Relational Database Design.

i. Red~~emand~~eddange information in tuples.

ii. Anomation — Insert Anomalies
- Delete Anomalies.
- Update Anomalies

5 What is meant by normalization?

Normalization

Normalization is the process of organizing the data in the database with two goal:

- To minimize the redundancy from a relation or set of relations.
- To ensure the data dependencies

It is also used to eliminate the undesirable characteristics like Insertion, Update and Deletion Anomalies.

Normalization divides the larger table into the smaller table and links them using relationship.

6	What is meant by functional dependencies?	

Functional dependency in DBMS, as the name suggests is a relationship between attributes of a table dependent on each other. It is a constraint between set of attributes. let us consider P is a relation with attributes A and B. Functional Dependency is represented by -> (arrow sign)

A → B

The following is an example that would make it easier to understand functional dependency:

We have a **<Department>** table with two attributes: **DeptId** and **DeptName**.

DeptId = Department ID
DeptName = Department Name

The **DeptId** is the primary key. Here, **DeptId** uniquely identifies the **DeptName** attribute.

DeptId	DeptName
001	Finance
002	Marketing
003	HR

Therefore, the above functional dependency between **DeptId** and **DeptName** can be determined as **DeptId** is functionally determines **DeptName**:

DeptId -> DeptName

7 Define decomposition.

5. Decomposition Rule:

This rule is the reverse of Union rule and also known as **project rule**.In the **decomposition rule**, if X determines Y and Z together, then X determines Y and Z separately.

Proof of this Rule:

$X \rightarrow YZ$ (given)…. (1)
$YZ \rightarrow Z$ and $YZ \rightarrow Y$ (using reflexive method) …. (2)
$X \rightarrow Y$ and $X \rightarrow Z$ (using transitive rule) …. (3)

Example:

Rollno → Firstname, Lastname
then, **Rollno → Firstname** and **Rollno → Lastname**

8 List the difference between first normal form and second normal form

First Normal Form (INF):

- A relation will be 1NF if it contains an atomic value.
- It states that an attribute of a table cannot hold multiple values. It must hold only single-valued attribute.
- First normal form disallows the multi-valued attribute, composite attribute, and their combinations.

Second Normal Form (2NF):

- In the 2NF, relational must be in 1NF.
- In the second normal form, every attributes are fully functional dependent on all prime attribute.

Prime attribute – Member of the candidate key
Non-Prime attribute – Non Member of the candidate key

9 When the functional dependency is said to be Trivial Functional dependency?

Trivial Functional Dependency

It occurs when B is a subset of A in:

A ->B

10 What is meant by full functional dependency?

iv. Fully fuctional Dependony

A non keying attribute depends on the entire keying attribute is called fully functional Dependency.

P P
Rg no, Roll no, name, dep

11 Explain reflexivity rule?

1. Reflexive Rule:

In the **reflexive rule**, if X is a set of attributes and Y is the subset of X, then X functionally determines Y. X -→ Y

Example:
Lastname ⊆ Firstname, Lastname
then, **Firstname, Lastname → Lastname**

12 Explain augmentation rule?

2. Augmentation Rule:

In the **augmentation rule**, if X determines Y and Z is any attribute set, then XZ determines YZ. It is also called as a partial dependency.

Example:
Regno → Firstname, Lastname
then, **Regno, address → Firstname, Lastname, address**

13 Explain transitivity rule?

3. Transitive Rule:

In the **transitive rule**, if X determines Y and Y determines Z, then X also determines Z.
Example:
Rollno → address and **address → Pincode**
then **Rollno → Pincode**

14 What is meant by dependency preservation?

ii. Dependency preservation
↳ If any constrain set in the original relation the same constrain will be ensure in any one of the decomposed relation,

15 Compare 3NF and BCNF

Third Normal Form (3NF):

A relation will be in 3NF if it is in 2NF and not contain any transitive dependency.. If there is no transitive dependency for non-prime attributes, then the relation must be in third normal form. 3NF states that all column reference in referenced data that are not dependent on the primary key should be removed.

3NF is used to reduce the data duplication. It is also used to achieve the data integrity

Boyce codd Normal Form (BCNF):

- BCNF is the advance version of 3NF. It is stricter than 3NF.
- A table is in BCNF if every functional dependency X → Y, X is the super key of the table.
- For BCNF, the table should be in 3NF, and for every FD, LHS is super key.

Example: Let's assume there is a company where employees work in more than one department.

16 What is transaction?

A **transaction** is a **unit** of program execution that accesses and possibly updates various data items. Usually, a transaction is initiated by a user program written in a high-level data-manipulation language (typically SQL), or programming language (for example, C++, or Java), with embedded database accesses in JDBC or ODBC. A transaction is delimited by statements (or function calls) of the form **begin transaction** and **end transaction**. The transaction consists of all operations executed between the **begin transaction** and **end transaction**.

17 Illustrate the properties of transaction?

Transaction Properties:
The concept of transactions can be described with the following four key properties described as **ACID – Properties**

Atomicity. This is "all-or-none" property Either all operations of the transaction are reflected properly in the database, or none are.

Consistency. Execution of a transaction in isolation (that is, with no other transaction executing concurrently) preserves the consistency of the data- base.

Isolation. Even though multiple transactions may execute concurrently, the system guarantees that, for every pair of transactions T_i and T_j , it appears to T_i that either T_j finished execution before T_i started or T_j started execution after T_i finished. Thus, each transaction is unaware of other transactions executing concurrently in the system.

Durability. After a transaction completes successfully, the changes it has made to the database persist, even if there are system failures.

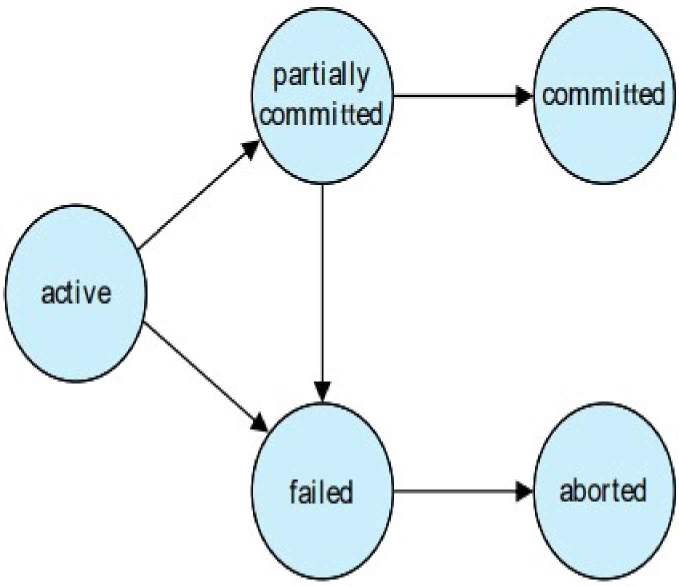
18 When is a transaction rolled back?

*correct a theriyala

When a detection algorithm determines that a deadlock exists, the system must **recover** from the deadlock. The most common solution is to roll back one or more trans- actions to break the deadlock. Three actions need to be taken:

19 What are the states of transaction?

A transaction in a database can be in one of the following states :



20	Outline the reasons for allowing concurrency. Need for Concurrency control: • To apply Isolation through mutual exclusion between conflicting transactions • To resolve read-write and write-write conflict issues • To preserve database consistency through constantly preserving execution obstructions • The system needs to control the interaction among the concurrent transactions. This control is achieved using concurrent-control schemes. • Concurrency control helps to ensure serializability
21	When the schedule is considered to be serial? **Serializable Schedule:** The non-serial schedule is said to be in a serializable schedule only when it is equivalent to the serial schedules, for an n number of transactions. A serializable schedule helps in improving both resource utilization and CPU throughput. These are of two types: • Conflict Serializable schedule • View Serializable schedule
22	What are the two types of serializability? Types of Serializability: • Conflict Serializability • View Serializability **Conflict Serializability:** **Conflict Serializable:** A schedule is called conflict serializable if it can be transformed into a serial schedule by swapping non-conflicting operations.

	View Serializability A Schedule is called view serializable if it is view equal to a serial schedule (no overlapping transactions). Two schedules S1 and S2 are said to be view equal if below conditions are satisfied : • Initial Read • Updated Read • Final Write
23	Explain lock **1) Lock based protocols** Lock based protocols allow a transaction to access a data item only if it is currently holding a lock on that item. The data item can be locked in two modes: • **Shared Mode (S)** • **Exclusive Mode (X)**
24	Explain the phases of two phase locking protocol. **Two Phase Locking** This protocol requires that each transaction issue lock and unlock requests in two phases: 1. **Growing phase: A transaction may obtain locks, but may not release any lock.** 2. **Shrinking phase. A transaction may release locks, but may not obtain any new locks.** Initially, a transaction is in the growing phase. The transaction acquires locks as needed. Once the transaction releases a lock, it enters the shrinking phase, and it can issue no more lock requests. The point in the schedule where the transaction has obtained its final lock (the end of its growing phase) is called the lock point of the transaction.

1	Explain in detail about Functional Dependencies with example.
2	Explain in detail about first, second and third normalization form with example.
3	Explain in detail about Boyce codd normal form, fourth normalization and fifth normalization form with example.
4	i) What is Closure of an Attribute and Outline the steps involved in finding the Closure of an Attribute? ii) Consider the relation R(A, B, C, D, E) having the Functional dependencies : FD1 : A → BC FD2 : C → B FD3 : D → E FD4 : E → D And calculate the closure of attributes of the relation R
5	Explain the types of serializability with examples.
6	State and explain the lock based concurrency control with suitable example
7	Explain how dead lock is handled with example?

8	What is Concurrency? Explain it in terms of locking mechanism and two-phase Commit Protocol.

1	Make use of the following two transactions: T1: read(A); read(B); if A = 0, then B := B + 1; write(B). T2: read(B); read(A); if B = 0, then A := A + 1; write(A). Add lock and unlock instructions to transactions T1 and T2, so that they observe the two-phase locking protocol. Can the execution of these transactions result in a deadlock? Generalize your view.
2	Experiment with the following schedules and state whether it is conflictserializable and/or view-serializable. If you cannot decide whether a schedule belongs to either class, explain briefly. The actions are listed in the order they are scheduled, and prefixed with the transaction name. (i) T1: R(X) T2: R(X) T1: W(X) T2: W(X) (2) (ii) T1: W(X) T2: R(Y) T1: R(Y) T2: R(X) (2) (iii) T1: R(X) T2: R(Y) T3: W(X) T2: R(X) T1: R(Y) (2) (iv) T1: R(X) T1: R(Y) T1: W(X) T2: R(Y) T3: W(Y) T1: W(X) T2: R(Y) (2)

3 Given the following two transactions, and assuming that initially x = 3, and y=2,

Time	T_1	T_2
1	read_item(y)	read_item(y);
2	y:=y+3;	y:=y*1.2;
3	write_item(y);	write_item(y);
4	read_item(x);	read_item(x);
5	x:=x+y;	x:=x+y;
6	write_item(x);	write_item(x);

1. create all possible serial schedules and examine the values of x and y;

4 Experiment with the DBMS and justify your answer with dynamically maintaining a wait-for graph rather than create it each time the deadlock detection algorithm runs? Explain your answer.