Solution 1 (a) Mapping cardinalities express the number of entities in one entity set that can be associated with entities in another set through a relationship. There are four main types of cardinalities in an ER model:

1. One-to-One (1:1): Each entity in both sets is related to at most one entity in the other set. Example: A person can have only one passport, and each passport is issued to only one person.
2. One-to-Many (1:M): An entity in one set can be related to multiple entities in the other set, but each entity in the second set is related to only one entity in the first. Example: A department can have many employees, but each employee works in only one department.
3. Many-to-One (M:1): Multiple entities in one set are related to one entity in the other set. It is similar to one-to-many but reversed in direction. Example: Many students can be advised by the same instructor, but each student has only one advisor.
4. Many-to-Many (M:N): Entities from both sets can be related to multiple entities in the other set. Example: Students can enroll in multiple courses, and each course can have multiple students.

The importance of cardinalities lies in their ability to define the relationships between entities, ensuring accurate database design

(b) A weak entity is an entity type that does not have a key attribute of its own. Instead, weak entities are identified by being related to a specific strong entity (also called the owner entity) and a partial key. Weak entities cannot exist without their owner entities, and this dependence is expressed by the existence dependency constraint. Weak entities usually participate in a one-to-many identifying relationship with the strong entity. The weak entity is identified using the primary key of the strong entity along with the weak entity's partial key. The participation of weak entities in the relationship is always total, meaning every weak entity must be related to a strong entity.

(c) To avoid redundancy, we often normalize the database, splitting larger tables into smaller ones, ensuring that each table stores only related data. We also use ER modeling techniques like removing unnecessary many-to-many relationships or redundant entities by redesigning schemas to avoid duplicate information.

(d) A composite key occurs when more than one attribute is needed to uniquely identify an entity. The combination of these attributes forms a unique identifier, but no proper subset of them would suffice. For example, in the Course_Registration entity, which records which students are enrolled in which courses, neither Student_ID nor Course_ID is sufficient alone to identify a row uniquely because a student can take multiple courses, and a course can have multiple students. However, the combination of Student_ID and Course_ID forms a composite key that uniquely identifies each entry in the Course_Registration entity. Composite keys are essential when no single attribute

can guarantee uniqueness on its own. This concept often appears in many-to-many relationships

Solution 2: The correct ER diagram is D:

- Entities: Instructor, Course, Student, and Classroom.

There's a relationship b/w:

- Instructors (Teaches) courses
- Students (Registration) courses
- Courses (Allocation) classrooms
- Instructors (Guides) students

When we link them and we get a figure like (iv)

Solution 3: The correct relations are:

1. Person (NID, Name) – This will hold because NID is the primary key, and Name is an attribute of the entity "Person."

2. Qualification (NID, ExamID, QualifiedDate) – This will hold because it's a many-to-many relationship between Person and Exam.

3. Exam (ExamID, NID, ExamName) – This will NOT hold because NID is not an attribute of the Exam entity. The correct relation should be Exam (ExamID, ExamName).

4. Exam (ExamID, ExamName) – This will hold because ExamID is the primary key, and ExamName is an attribute of the Exam entity.

We get two tables One for Entity Person - Person(NID,Name) One for Entity Exam - Exam(ExamID,ExamName) Now for relationship Entity Qualification, we can not add *QualifiedDate* in any of the two tables formed so we need to create a new table using Primary Key from both Entity Person and Entity Exam. We obtain Qualification(NID, ExamID, QualifiedDate) Hence *Option (C)* is incorrect.

Solution 4 & 5:
First strong entity types are made to tables. So, we get two tables M and P.

Assume R1 is 1:1 or 1:n as that would minimize the number of tables as asked in question. Now participation of M in R1 is total (indicated by double arrow) meaning every entity of M participate in R1. Since R1 is not having an attribute, we can simple add the primary key of P to the table M and add a foreign key reference to M. This handles R1 and we don't need an extra table. So, M becomes {M1, M2, M3, P1}.

N here is a weak entity weakly related to P. So, we form a new table N, and includes the primary key of P(P1) as foreign key reference. Now (P1,N1) becomes the primary key of N.

Thus we get 3 tables.

M: M1,M2,M3,P1 - M1 primary key, P1 references P

P: P1,P2 - P1 primary key

N: P1,N1,N2 - (P1,N1) primary key, P1 references P.

Solution 6: Correct Answer: B

We need just two tables for 1NF.

T1: {A11, A12, A13}

T2: {A21, A22, A23, A11}

A23 being multi-valued, A21, A23 becomes the key for T2 as we need to repeat multiple values corresponding to the multi-valued attribute to make it 1NF. But, this causes partial FD A21→A22 and makes the table not in 2NF. In order to make the table in 2NF, we have to create a separate table for multi-valued attribute. Then we get

T1:{A11,A12,A13}– key is A11
T2:{A21,A22,A11}– key is A21
T3:{A21,A23}– key is {A21,A23}

Here, all determinants of all FDs are keys and hence the relation is in BCNF and so 3NF also. So, we need minimum 3 tables.
Note: Even if “3NF” isn't explicitly mentioned in question we should assume it and we cannot add NULL entries during ER to Relational conversion.

Solution 7:

Applying the rules, minimum 6 tables will be required-

- Account (Ac_no , Balance , b_name)
- Branch (b_name , b_city , Assets)
- Loan (L_no , Amt , b_name)
- Borrower (C_name , L_no)
- Customer (C_name , C_street , C_city)
- Depositor (C_name , Ac_no)

Solution 8: Minimum 4 tables are required to convert the ER diagram into RDBMS table and Total 5 foreign keys are required

For a 1:m relationship, the foreign key goes in the table representing the "many" side of the relationship. For an m:n relationship, an additional table is created to store the foreign keys for both sides of the relationship.

The total number of foreign keys will depend on the number of relationships between entities. Each relationship (except 1:1 relationships) introduces at least one foreign key.