NCERT Solutions for Class 11 Accountancy

Financial Accounting Part-2 Chapter 7

Structuring Database for Accounting

Short answers : Solutions of Questions on Page Number : **549**

Q1 :

State main categories of data models.

Answer :

The following are the various categories of data models.

(i) Relational Data Model

(ii) Hierarchical Data Model

(iii) Network Data Model

1. Relational Data Model- This data model is based on the relationship of collected data values. In this data model, data is organised into rows and columns. A row is regarded as a tuple, a column header is known as an attribute and the collective set of rows and columns, i.e. a table is called a relation. The table is known as a relation as it expresses the relationship between the rows and columns. This model provides the storage and retrieval functions and defines the data structure.

The relational data model was first introduced by Ted Codd in 1970 in a classic paper Codd, 1970. Prior to this, there were other data models which were proposed in sixties such as, hierarchical data model and network data model. These models are also known as legacy-models due to their large existing user-base.

2. Hierarchical Data Model- This data model mainly consists of records and parent-child relationships. While, a record is regarded as a collection of values that provides information regarding an entity or a relationship instance, on the other hand, a parent-child relationship explains the relationship between the parent record and children record type. In this data model, the records are organised in a tree structure rather than as an arbitrary graph. The data is represented by a collection of records and relationship among data is represented by links.

3. Network Data Model- This type of data model is sometimes also known as DBTG model, as the original network model was presented in CODASYL Data Base Task Group's 1971, i.e. (DBTG). This data model basically consists of records and sets. While, data is stored in records, which consists of a group of related data values, on the other hand, sets describes the relationship between two records types. In this data model, data is also represented by collection of records and the relationship among data is represented by sets. This model provides many-to-many relationships in data.

Q2 :

How are computers useful in processing the accounting data?

Answer :

Data processing is a process of collecting, storing, summarising, analysing and interpreting the data and facts so as to represent reliable information for efficient and effective decision making. Data processing, in short, is a transformation of raw data into useful information. Computer system plays a very important role in processing the accounting data. Data processing requires a mechanism to store data content in a manner that allows easy and convenient retrieval of data as and when required. This can be easily done with the help of computers by designing suitable database for accounting. Moreover, computers also help in eliminating the duplication of data. Unlike the manual accounting system, computers are not subject to tiredness, boredom or fatigue, therefore; the reliability of data processed by computers is very high. Besides reliability, computers can process data at comparatively higher speed and with great degree of precision and accuracy.

Q3 :

What do you understand by accounting data? Discuss the stages through which it is finally transformed for being presented as information in financial statements.

Answer :

Accounting data refers to the facts and financial data contained in journals, ledgers, financial statements and other books of accounts. That is, accounting data pertains to the financial data recorded in the books of accounts.

The procedure through which the accounting data is transformed for being presented as information in financial statements is known as data processing. Data processing is a process of collecting, storing, summarising, analysing and interpreting the data and facts in such a manner so as to fetch reliable information for efficient and effective decision making. The following are the various stages that are involved in transformation of data into information so as to present in the financial statements.

1. Collecting Data from the Source Documents- First of all, data is collected from the source documents such as payment slips, receipt slips, etc. for preparing vouchers. The preparation of vouchers is the basis for recording the accounting data in a systematic and chronological (date-wise) manner.

2. Input Data- A suitable database is designed to enter the accounting data contained in the vouchers. This is done with the help of pre-designed Data Entry Form. This form is similar to the physical voucher form.

3. Data Storage- Data storage is created for storing the input of data. It involves structuring the database which is used for recording the accounting data. Following is the format of blank data record.

Code	Name	Type

4. Manipulation of Data- This stage involves transformation of the stored data to generate final reports.

5. Output of Data- It means representing the accounting reports such as Journals, Ledger, Trial Balance, Financial Accounts, etc. as useful information. The accounting users can obtain these reports just by accessing the transformed data. This stage basically implies generation of final reports in a pre-designed format.

Q4 :
What do you understand by database? How does it differ from DBMS?

Answer :

Database is a collection of inter-related data, events and transactions. It is organised in a particular manner and provide access to the various users simultaneously. In this system, data is gathered and stored for a particular operation.

The following are the two important properties of a database.

1. Shared Property- The database is a combination of related data that can be accessed by the people who have authority to access it and to meet their different information needs.

2. Integrated Property- The database is arranged in such a manner so as to avoid and eliminate duplication of data.

On one hand, database are used for storing accounting data, whereas, on the other hand, DBMS is that software which helps in creating, developing and maintaining the database. It is a system that provides easier access to data recorded in the database. It handles a huge amount of data and defines, organises and transforms the database for different functions as per the need. Hence, it can be said that DBMS facilitates access to the database.

Q5 :
What is meant by entity type? How it is different from entity set? Illustrate by giving suitable example from accounting reality.

Answer :

An Entity Type means a common definition that is shared by a collection of entities in terms of their attributes is known as Entity Type. A separate name is given to each entity for their further identification. The entity type is described in the database through its various attributes. The values of attributes of an entity belonging to particular entity type are known as Entity Instance.

On the other hand, an Entity Set is defined as a collection of all entity instances of a particular entity type. A set of attributes that is used to describe an entity type is referred to as Schema. The same set of attributes is shared by the set of entities belonging to a particular entity type. The entity set that consists of a collection of entities of a particular entity type is referred as extension of the entity type.

Example

350967 Sunita and Company 7, is an entity instance of an account whose

Code is 350967

Name is Sunita and

Company *Type* is 7.

In this example, entity refers to Accounts, on the other hand, Entity Set is the collection of entity instances of the entity type 'Accounts'.

Code	Name	Type
350967	Sunita and Company	7
350823	Jayanti and Company	7
360592	Sunil and Bros.	7

Q6 :
What do you understand by relationship type? How is it different from relationship instance and relationship set?

Answer :

The relationship between various entities that belong to different entity types in a specific manner is known as a relationship type.

A collection of various relationships that belongs to the same relationship type is called a relationship set. The result that is derived from the relationship type is known as instance and is shown in an entity set. The relationship prepared between the two entity types: Salary Slips and Employees, associates each salary slip with the employee who prepared it. Similarly, another example of relationship type of STUDY_IN associates one student entity and one class entity. In this example, Student and Class are entities and the relationship between i.e. STUDY_IN is the relationship set.

In the following ER diagram, the relationship types are shown in a diamond-shaped box. The participating entities are shown in the rectangular boxes, which are connected to the diamond-shaped box through straight lines.

Q7 :
What do you understand by multi-valued attribute? How is it different from complex and composite attribute? Illustrate by giving suitable example.

Answer :

Attributes may be defined as characteristics that reflect the features of the entity. In case of a person these characteristics may be height, weight, name, date of birth, etc. and in case of accounts it may be code, type, name, etc. These attributes own a value which is stored in database as data.

Multi-valued attribute is an attribute that has multiple values. For example, flavour of ice-creams. It is a multi-valued attribute as the flavour of an ice-cream can be strawberry, vanilla, butter-scotch, etc. In other words, multi-valued attributes are the different characteristics of a same entity.

Composite attributes are the combination or aggregate of related attributes. These attributes can be further divided into small portions to represent the independent meaning of some basic attributes. For example, contact details which is generally divided into telephone numbers (with STD code), mobile number and e-mail Ids.

Complex attributes are formed by grouping together the attributes of composite and multi-valued attributes. To show the grouping of components of composite attributes, the parenthesis ( ) are used and to show the grouping of components of complex attributes, the brackets { } are used.

Q8 :
What do you understand by the concept of weak entity used in data modelling? Explain the relevance of owner entity type, partial key and identifying relationship in the context of such modelling.

Answer :

The weak entities refer to those entities which do not have their own key attributes or identities. The weak entities are identified on the basis of their relation with the specific entities from another entity type in combination with some of their attribute values. These other entity types are regarded as identifying or owner entity type. The entity type, whose entities are in relationship with the weak entities are referred to as identifying or owner entity type. This can be described in other words, as the entities which are identified because of their relationship with their parent entity or dominant entity is referred to as weak entities or child entity type (or subordinate entity type), while the parent entity is referred to as identifying entity type. Accordingly, the relationship through which the weak entity type relates to its owner entity is known as identifying relationship of weak entity. For a unique identification of the weak entity that is related to the same owner entity, a set of attributes is used which is known as partial key. Sometimes, the partial key is also known as the discriminator. This is because it is used to uniquely identify the weak entities that are related to the same parent entity. For example, if we assume that no two children of the same employee have the same first name, then the attribute Name of Children is the partial key that can be used to identify the children (weak entities) that are related to the same employee (owner entity).

Q9 :
What is a participation role? State the circumstances under which the use of role names becomes necessary in description of relationship types.

Answer :

A particular role is played in the relationship by each entity that participates in a relationship type. The Participation role specifies the existence of an entity depending on being related to another entity via a relationship or not. The two types of such constraints are total or partial participation.

Total Participation- If it is required that every entity of an entity type must relate to another entity type, then such an entity can exist only if it participates in that specific relationship. This kind of participation is called total participation. This type of participation constraints is also known as existence dependency. For example, if it is required that every account must be related to at least one of the accounts type, then an account entity can only exist if it participates in CLASSIFY relationship instance.

Partial Participation- If in case, it is not required that every entity of an entity type must relate to another entity type, then the participation of the entity in the relationship type is partially constrained. Such an entity can exist even if a part of it is related to the relationship type. For example, if every employee is not expected to prepare at least one of the vouchers, then the participation of employee in PREPARED_BY relationship is partial.

In case, there exist two relationships between any two entities, then in order to map both the relationships, the primary key of the common entity is to be included twice. But a relation cannot have the same name, hence, we need to use different role names as attributes as foreign keys to indicate the relations.

Q10 :
Define foreign key. How is this concept useful in relational data model? Illustrate with suitable example.

Answer :

A foreign key is defined as a key that refers to a primary key column of another table. In other words, it is a field in a relational table that matches the primary key of another table. These keys relate different tables together in order to form an integrated database. For example, let us consider the two following tables- STUDENT table and BOOKS ISSUED table. The STUDENT table includes all the student data and the BOOKS ISSUED table includes all the books issued by the students from the school library. The basic objective here is that all the books issued must corresponds to the students that are listed in the STUDENT table. In order to do this, we need to place a foreign key in the BOOKS ISSUED table and need to relate it to the primary key of the STUDENT table.

STUDENT Table	
S_Id	**Name**
142	Noor
135	Jagat
153	Mohini

BOOKS ISSUED Table	
Name_of_Books	**S_Id**
Economics	135
Mathematics	135
English	142

The Join of STUDENT table and BOOKS ISSUED table appears as:

S_Id	Name	Name_of_Books
135	Jagat	Economics
135	Jagat	Mathematics
142	Noor	English

Thus, we can see that in the STUDENT table, student ID (i.e. S_Id) is the primary key, whereas, in the BOOKS ISSUED table, S_Id is a foreign key, which will be used to relate the data in the STUDENT table to that of the books issued by them in the BOOKS ISSUED table.

Q11 :
What is meant by NULL value? What are the reasons that lead to their occurrence in database relations?

Answer :

The absence of a data item which is represented by a special value is known as NULL value.

The following are the reasons that lead to its occurrence in the database relations.

(i) When a particular attribute does not apply to an entity

(ii) When the existing value of an attribute is unknown

(iii) When the value is unknown because it does not exist

Q12 :
Why are duplicate tuples not allowed in a relation?

Answer :

Tuple refers to a row in a table that represents a set or collection of related data values. It corresponds to the relationship which represents entity type as a set of tuples, where each tuple is an ordered list of values corresponding to attributes of relation. As each tuple represents a distinct data record entity, so no two tuples in a relation (i.e. a table) can have the same combination of values for all their data items. Moreover, if a new tuple is added in the relation, then it should not reflect the existing data-values, otherwise, it will violate the uniqueness constraint and the relation as a whole will not have a minimal super-key.

Q13 :
What do you understand by union compatibility of relations? For which operations such compatibility is required and why?

Answer :

The union compatibility of relations implies that the participating relations must fulfil the following conditions.

1. Same degree, i.e. The two relations must have the same number (set) of attributes.
2. Same domain of each corresponding pair of attributes of relation A and relation B, that is

Dom(A) = Dom (B)

That is, the domain (stands for data type) for the corresponding attributes must be identical.

Hence, we can say any two relations say, relation A and relation B are union compatible, *iff* (if and only if), both the relations have the same number of attributes and the domains of their corresponding attributes are the identical (column by column).

The following are the various operations for which union compatibility of relations is required.

1. Union (A ∪ B)- It contains all tuples from each of the relations.
2. Intersection (A ∩ B)- It contains all the tuples that are contained in both the relations A and B.
3. Difference (A - B)- It contains all the tuples that are contained in the relation A but are not present in the relation B.
4. Cartesian Product (A x B)- It contains the set of all concatenated tuples (*x*, *y*), where *x* is a tuple in the relation A and*y* is the a tuple in the relation B. The tuples in the Cartesian product contains the product of each tuple of the relation A and each tuple of the relation B. (It must be noted that for performing Cartesian product it is not necessary that the two relations must be union compatible).

In order to run these operations, it is important that the two relations must be union compatible. It is because of the fact that the relations without being union compatible that is not of same degree of attributes and having same domain, may lead to difficulty in performing such operations.

Q14 :
What is the need for database normalisation?

Answer :

Database Normalisation refers to the process of organising and maintaining the data into the database in an efficient manner. The basic rationales behind normalising the database are:

1. to eliminate the data redundancy i.e. to avoid/eliminate the storage of the same data at more than one places.

2. to ensure data dependency i.e. to store the related data in the database.

Database Normalisation is a very important process as it makes the database free from storage of irrelevant data and removes the duplicate data items from the database. As a result, normalisation ensures more free space available in the database. In addition, normalisation also enhances the reliability and consistency of the database by protecting the database against the data anomalies such as logical and structural problems.

Long answers : Solutions of Questions on Page Number : **549**

Q1 :

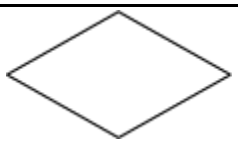
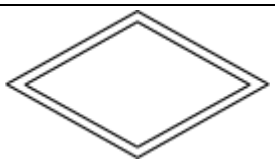
Discuss the basic concepts of Entity Relationship (ER) Model. Illustrate as to how an ER model is diagrammed.

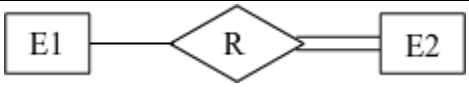
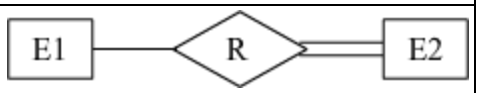
Answer :

Entity Relationship (ER) data model is proposed by Perter Pin-Shan Chen in 1976. This model is developed using DBMS to frame an application database. The ER model is a generalisation of old hierarchal and network approach models. This model describes data in terms of the following elements.

1. Entities
2. Attributes
3. Relationship between entities

These major elements of ER model are used to express a real world situation (reality) for which a database is to be designed. The database structure using the ER model can be shown diagrammatically with the help of ER diagrams. There are different symbols that are used to represent different types of entities, attributes, identifiers and relationships.

Major Elements of ER Model	Represented as (Symbols)
Entity Type is represented by a rectangular box	
Weak Entity Type is represented by a double-lined rectangular box	
Relationship Type is represented by a diamond-shaped box	
Identifying Relationship Type is represented by a double-lined diamond-shaped box	
Attribute Name is enclosed in ovals and are attached to their entity type with the help of straight lines	
Key Attribute Name enclosed in ovals and attached to their entity	

type by straight lines.	
Multi-valued Attributes are represented by double ovals.	
Derived Attributes are represented by dashed line Ovals	
Total Participation of E2 in R is represented as	
Cardinality Ratio 1 : N for E1 : E2 in R is represented as	

Q2 :
What integrity constraints are specified on database schema? Why each is considered important?

Answer :

A database schema implies the structure and design of database. Database schema is regarded as a collective term for various properties of database such as, tables, constraints, relations, etc. A relational database schema consists of a set of relational database schemas and set of integrity constraints.

There are mainly four different integrity constraints that are imposed on a relational database. These are mentioned as:

1. Domain Constraints- These constraints in the database schema state the conditions that each relational instance must satisfy. The value of each attribute of a relation must be an indivisible value and must be drawn from the domain associated with that attribute. Hence, the value of an attribute must confirm to the data type associated with the domain.

2. Key Constraints and NULL Value- These constraints imply the existence of candidate keys. As per the Key Constraints, in every instance of a relational database schema, the tuples can be uniquely identified by their values for certain attributes. This implies that each data record corresponding to a tuple of relation in a table must be distinct. That is, in other words, no two tuples in a relation can have the same combination of values for all their attributes. Every relation has at least one key, which is the combination of all its attributes and is known as Super-Key. The Super-key specifies the uniqueness constraints.

However; a Super-key may contain some additional attributes that are not necessary for unique identification. Minimal super key is that part of super key from which removal of any attribute from the Super-key leaves a set of attributes that is not a Super-key. That is, if any attribute is removed, then the uniqueness constraint is disturbed. The Minimal Super-key is also known as Candidate key.

In case a relation has many Candidate keys, then out of them one is arbitrarily chosen and regarded as Primary key. The attributes of Primary key are represented as underlined in a schema. The other Candidate keys are regarded as Alternate or Unique keys.

3. Entity Integrity Constraints- A Primary key is used to identify individual tuple in a relation. As per the Entity Integrity Constraints, the value of Primary key cannot be null. If in case, the Primary key has null value, then it implies that we cannot identify such tuples as they are same. This implies that tuples are duplicated and the uniqueness constraints are violated.

4. Referential Integrity Constraints- Referential integrity constraint is specified between two or more relations to maintain consistency among the tuples of such relations. These are the constraints implies by the existence of foreign keys. Hence, a tuple in one relation that refers to another relation must refer to an existing tuple in that other relation.

The above mentioned integrity constraints are certain conditions that are imposed on a database schema. The importance of integrity constraints is that by imposing certain specifications, the integrity constraints restrict the data that is stored in a database instance. In case, the database instance satisfies all the integrity constraints, then it is regarded as a legal database instance and

hence, can be stored in the database. In case, these constraints are violated, then it implies that database is not maintained in an efficient manner and the database contains illegal database instances.

Q3 :
Discuss the different types of update operations in relation to the integrity constraints which must be satisfied in a relational database model.

Answer :

The following are the different types of update operations in relation to the integrity constraints which must be satisfied in a relational database model.

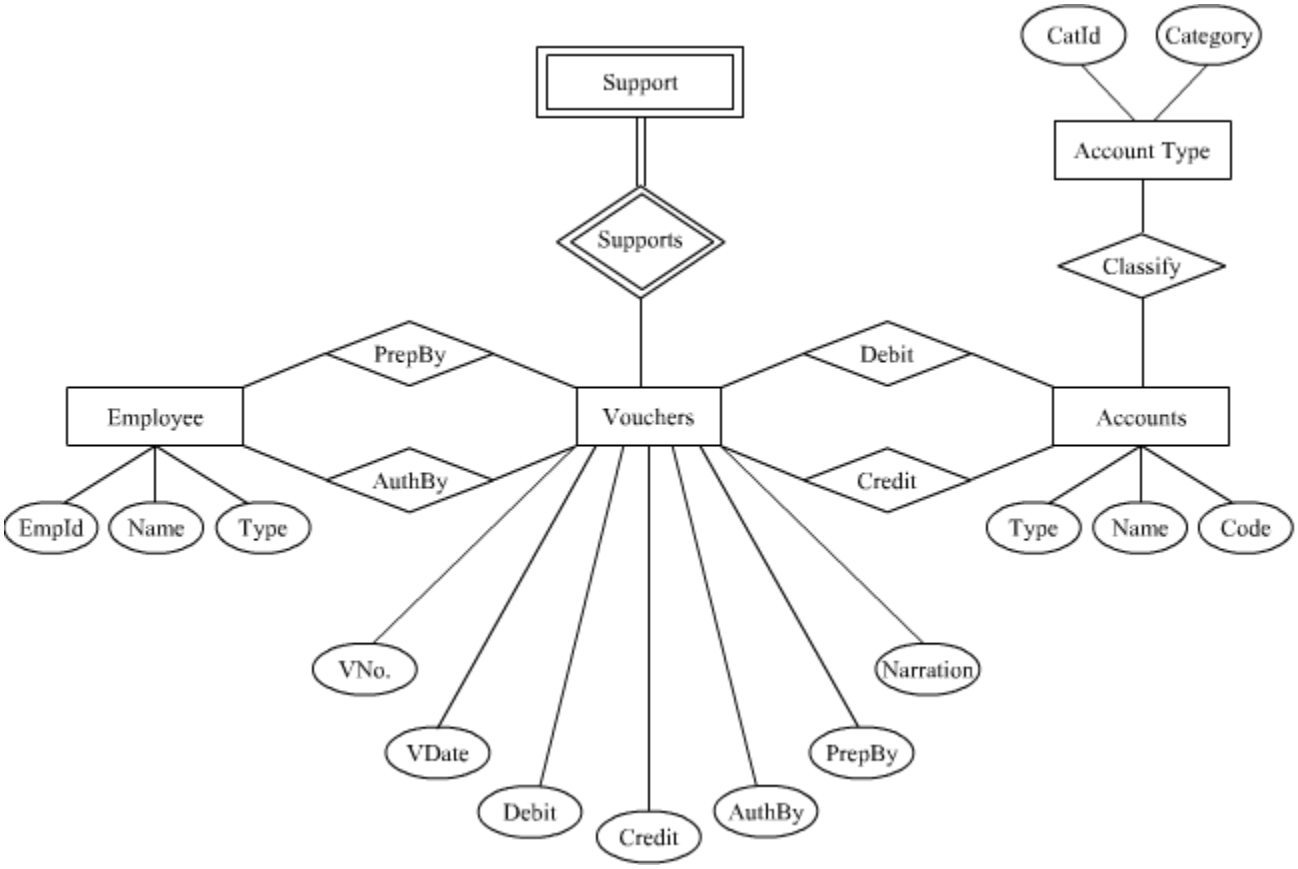
1. **Insert-** This implies addition of a new tuple in the existing relation. This operation can violate any of the four integrity constraints.
2. **Delete-** This operation is used to remove an existing tuple from the relation. This operation can violate Referential Integrity Constraints, if the removed tuple is reference by foreign key from other tuples in the database.
3. **Modify-** This operation is used to make changes in the existing values of the records in a data table. Generally, this operation does not lead to any violation problems, if the modification is done except on Primary key and on Foreign keys.

Hence, we can conclude that after applying these update operations, integrity constraints are specified on the relational database schema.

Q4 :
Discuss the steps you would take to transform an ER Model into various relations of Relational Data Model. Give suitable examples.

Answer :

The requisite to transform on ER model into various relations of the Relational Data Model is to have an ER design. Let following be the example of our ER model that we need to transform into Relational Data Model.



In the above diagram, the rectangular boxes are used to represent entities. The diamond shaped boxes are used to describe the relation type between the two entities. The double lined rectangular boxes are used to show the weak entities. The double-lined diamond shaped box is used to describe the identifying relationship type. In ovals, attribute names are enclosed and attached to their respective entity type through the straight lines.

The following section describes the procedure to transform ER model into Relational Data Model.

1. Creation of Relation for Every Strong Entity

A separate relation that includes all the simple attributes is to be created for each strong entity type in the ER schema. A primary key out of these attributes is chosen arbitrarily for easy and unique identification of the strong entity. For example, in the above ER design, Employee, Vouchers, Accounts and Account Type are the strong entities as they have primary key which is one of their unique attributes. These attributes are shown in the ovals and are attached to their respective entity type by the straight lines.

Separate relation is created for each strong entity. This is represented as:

Employee (EmpId, Name, Type)

Vouchers (VNo, SNo. Narration)

Accounts (Type, Name, Code)

Accounts Type (CatId, Category)

2. Creation of Relation for each Weak Entity Type

Weak entities do not have their own identities and are identified through the identifying relationship. So, we can say that every weak entity has its own entity that helps in its identification. A separate relation that includes the attributes is to be created for each weak entity. The primary key of this new relation is the combination of its unique attributes, along with the primary key attribute of the owner relation. For example, Support entity is the weak entity as it does have its own primary key; Voucher is the owner entity of Support entity.

Support entity has a partial key which is SNo Assigned to each document. Therefore, VNo, the primary key of the vouchers along with SNo is designed as composite key for the Support entity. The relation so formed can be represented as:

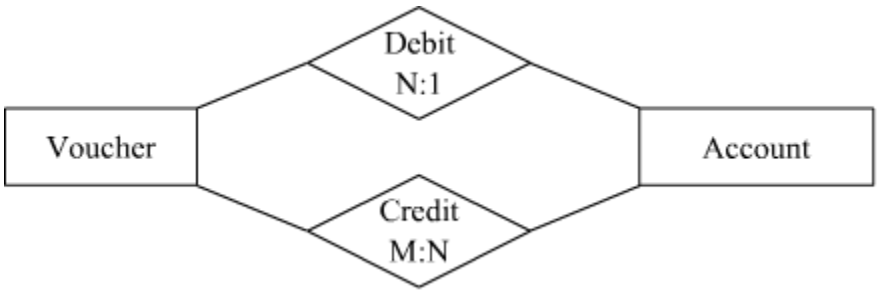
Support (VNo, SNo, dName sDate)

3. Identity Entity Types Participating in Binary 1 : N Relationship Type

First of all, the first relation on n-side of the relationship and second on the 1-side of such relationships is to be identified. The primary key of the second relation should be included in the first relation as its foreign key. For example, in the example, an employee can authorize a number of vouchers. It means that Voucher entity participates in AuthBy relationship on the n-side, on the other hand, the Employee entity participates in the same relationship on the 1-side. Similarly, Prep. by relationship between employees and vouchers participates in binary 1 : N relationship.

4. Identify entity types participating in binary M : N relationship type:

A new relation is to be created for each binary M : N relationship type. This new relation should include foreign keys to represent primary key of the new relation. For example, consider two entities namely, Voucher entity and Account entity. These entities have two relation debit and credit. The debit relationship has cordially ratio of N : 1 i.e. many debit voucher relates to one accounts. On the other hand, the credit relationship has cardinality ratio of M : N. For example, many credit vouchers are related to many accounts.



So, based on the diagram, the following two relations can be formed as:

(i) **Credit** (VNo, SNo, Code, Amount, Narration)

(ii) **Debit** (VNo. SNo, Code, Amount, Narration)

In the Credit relation, Credit Code is included as foreign key to represent the primary key of the Accounts relation. VNo is included as foreign key to represent the primary key of the Vouchers relation. Both VNo. and the Code together constitute the primary key of the new relation Credit.

At the end, the following relations have been formed to formulate the relational data model for the above example of ER design.

Employee (EmpId, Name, Type)

Vouchers (VNo, SNo, Naration)

Accounts (Type, Name, Code)

Accounts Type (CatId, Category)