4. Entity Relationship Model

a) **ER-Model:** Used to construct conceptual data model, representing the structure and constraints of a database, which is not dependent on a software (like DBMS) or any data model to be used to implement a database. Its basic elements are Entities, Attributes and Relationships.

b) **Entity:** An object of the realworld which can store data and can be defined clearly. Eg Customer, Furniture etc.

c) **Attribute:** Description or characteristics of an entity (what differentiates one entity from another). There are several types of attributes:
   i. **Simple Attributes:** Simple attributes has only one component, is independent and cannot be broken up.
   ii. **Composite Attributes:** Composite Attributes has many components each one existing independently.
   iii. **Solitary Value Attributes:** Contains one value
   iv. **Multiple Value Attributes**: Contains many values. Eg Phone Numbers
   v. **Derived Attribute:** Where its value is derived from another attribute or set of attributes, Eg age.

d) **Attribute Domain:** A set of values for an attribute. Eg Attribute Domain for Staff_Number is integer (1-30). Integer types are; Character, Numeric, Date

e) **Null Value:** An attribute value which does not exist, unknown at times or not related. (It is not zero)

f) **Key:** is one or several attributes which can differentiate the entities they describe. Its value must be unique and must not be null. Eg ID_Number

g) **Relationship:** The link between entities. There are 3 types of relationships:
   i. **Unary Relationship** (**Recursive**): Is a relationship involving only one entity. Eg Staff manages Staff
   ii. **Binary Relationship:** Relationship between two entities. Eg Student registered_to Course
   iii. **Ternary Relationship:** Simultaneous relationship between three entities. Eg Sponsor offer Scholarship to Student

h) **Relationship Attribute:** Like entity attributes, relations also can have attributes which describes the relationship. Eg Treatment relationship between Patient and Doctors can contain Type_of_treatment, Type_of_ailment and Medicine as attributes.

i) **Relationship Cardinality:** describes the number of relationships between one entity to other entities. There are 3 types of relationship cardinality:
   i. **One to One** Relationships (1:1)
   ii. **One to Many** Relationships (1:M)
   iii. **Many to Many** Relationships (M:N)

j) **Relationship Participation:** Entity participation in a relationship can be either compulsory or optional.

k) **Guidelines and Steps in Constructing an ER Model:**
   i. Do not insert the **System Environment** as an entity or attribute etc
   ii. **Attribute** and **key** for a particular environment are not necessarily same for others. Eg In a library, Burrower may be student or lecturer.
   iii. An entity must contain **description**. Objects with only one characteristic are attributes, not entities.
   iv. **Convert** multiple value attributes to entities.
   v. Two entities can have more than one **relationship**
   vi. Apply **the top down approach** in modelling entity and main relationship with sets of limited attributes.

Follow the steps below for top down approach:

1. Determine **entities** and **relationships** between them. Start with the **main entity** followed by others
2. Determine the **attributes** related to the **entities**
3. Determine the **attributes** related to the **relationships** (if any)
4. Choose the **keys** for the **entities**
5. Determine the **domain** for each **attribute**
6. **Combine** the diagrams of entity, relationship and attribute to develop a complete ER model. (No hanging entities)
7. Thoroughly **check** and **refine** the ER model (If necessary, discuss with users)

5. Enhanced Entity Relationship Model

a) **EER-Model:** The ER model is enhanced so that data in a complex business environment can be represented more accurately. It has additional concepts such as :

i. **Weak Entity:** Has no significance and its existance depends on another strong entity, without which it doesn't have any meaning. It doesn't even have its own key.

ii. **Composite Entity:** is formed when a M:N relationship is transformed to an entity

iii. **Super Class and Sub Class Entity:** Entity type used to represent a group of entities having same characteristics.

1. **Super Class Entity**: Type of entity which is more general and has relationship with one or more sub class.
2. **Sub Class Entity**: is one or several entities with different attributes from one another but share the same attributes as its super class

iv. **Generalization:** Process of creating a type of entity that is more general than a set of special entities. Eg. Making a general entity called Vehicle to generalise special entities; Cars, Lorries and Motorcycles.

v. **Specialization:** (Opposite to Generalisation) Process of determining one or several subclasses from an Entity (this entity later becomes a superclass). Eg Furniture entity divided into sub class entities of Made_furniture and Purchased_furniture.

vi. **Aggregation**

vii. **Disjoint Rule:** States that for the same superclass, entity occurance of a subclass may not become a member of another subclass at the same time.

viii. **Overlap rule:** (Opposite of Disjoint Rule) States that in the same super class, an entity occurance of one subclass can be a member of one or more than one of the other subclasses at the same time.

6. The Relationship Model

a) **The relationship model:** represents data in the form two dimensional tables with rows as records and columns as attributes.

b) **Logical Design:** Process of interpreting the conceptual data model to the logical data model.

c) **The Structure of Relationship Data:**

i. **Data** (same as Entity in the ER model) is represented as relationships.

ii. **Relationship** is a two dimensional **table** (rows and columns) with data and names. (Alternative term: **File**.)

iii. **Columns** in the table represent relationship **Attribute** or **Fields**

iv. **Rows** contain **record** or attribute values, also known as **Tuple**.

v. **Relationship structure** is portrayed in two formats, they are:

1. **Brief Text Statement:** RELATIONSHIP NAME (Attribute1, Attribute 2, … , Attribute n)
2. **Graphic Representation:**

RELATIONSHIP NAME

Attribute1	Attribute2	Attribute3	Attribute n

d) Relationship Key: Similar to ER model, key is used to identify each Tuple (Row/Record) uniquely.

i. **Primary Key:** a must have key for each relationship. Primary key is underlined to differentiate it from other attributes which are not keys.
ii. **Composite Key**: Primary keys consisting of more than one attributes.
iii. **Foreign Key:** Attribute(s) in a relationship that becomes a primary key in other relationships, to represent the connection between the two relationships.

e) Relationship Characteristics:

Characteristics that differentiate relationship table from a non-relationship table:

i. Each relationship (table) has a **unique name**
ii. Each data entry to the cell is **atomic**, doesn't have multiple value attributes
iii. Each **row** is unique
iv. Each **column** has a unique name
v. **Arrangement** of rows is not important
vi. **Arrangement** of columns is not important

f) Integrity Constraints: Several constrains the relationship model has to ensure accuracy and data integrity, they are:

i. **Domain Constraint**: In a relationship, each value in the same column MUST have the same domain. (Domains are values of attributes, same as in ER model, domain has Domain name, Description, Data type and Data size).
ii. **Entity Integrity:** Each relationship MUST have a primary key and a data value that is legal. (Legal: Primary key must be unique, not a null value)
iii. **Reference Integrity:** To ensure data rows consistency between related tables.
If a foreign key exists in a relationship, then:
   1. The value of each foreign key must match the value of each primary key in the relationship it represents. OR
   2. The value of the foreign key must be NULL. (Eg when new workers are not yet assigned to a specific department)
iv. **Organizational Constraints:** Statements in the form of Rules, which show the characteristics or constrains of a business or organization. Eg. "*If a pre-degree student intends to fulfil the conditions of becoming a degree holder he must accumulate 120 credit hours*" This can be done by making use of data manipulation facilities and other aspects of DMBS. (Eg. SQL statement can define that if the GPA is less than 2.00, the status is fail)

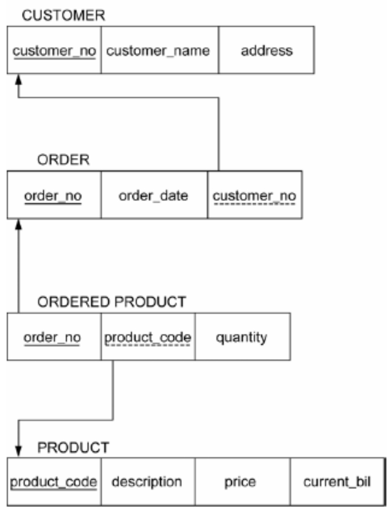

Figure 6.6: Reference- representation integrity table

7. Conceptual Design Methodology

a) Conceptual Data Model: Refers to the ER Model
b) Logical Data Model: Refers to the Relationship model or table.

c) Conceptual Design Phase:

i. Begins with the production of a conceptual data model on information requirements of an organization.
ii. Is not dependent on implementation aspects of the DBMS such as program application, programming language, hardware and performance issues.

d) Six Steps in the conceptual Design Phase:

i. **Determination of Entity:** Nouns, Eg Worker
   1. Examine the requirement specifications

ii. **Determination of Relationship:** Verbs, Eg Work (Worker –Work—Company)
   1. Model relationships that are needed. Try to limit to binary relationships.
   2. Determine relationship cardinality and relationship participation (1:M, M:N, 1:1)

iii. **Determination of Attribute for Entity and Relationship:**
   1. Determine attributes by refering to nouns in the user requirement specifications
   2. Model the attributes with the attribute types (simple or composite/solitary or multivalued/derived)
   3. Document the attributes into a data dictionary

iv. **Determination of Domain for Attribute:**
   1. Fill the domain information of data dictionary with:
      a. Allowable set of values for the attribute
      b. Size and format of each attribute

v. **Determination of Primary Key:**
   1. Guidelines for choosing a primary key from several possible key candidates:
      a. Key candidate with the minimum attribute set
      b. Key candidate least likely to have its values changed
      c. Key candidate with the shortest characters
      d. Key candidate with the lowest maximum value
      e. Key candidate that is the easiest to be used
   2. Fill the primary key information of the data dictionary

vi. **Re-evaluation of Data model with users**

8. Logical Design Methodology

a) Logical Design:

i. Process of interpreting the conceptual data model to the logical data model.
ii. Necessary as ER model cannot be directly implemented into any DBMS.
iii. Logical design is also not dependent on the DBMS (Similar to conceptual design)

b) Seven Steps in logical design:

i. **Removal of M:N Relationships:**
By introducing an intermediate Entity, the M:N relationship is then replaced by two 1:M Relationships.

ii. **Removal of complex Relationships:** (Three or more entities in a relationships, eg Ternary)
Needs to be changed into binary relationships that can represent the complex Relationship.

iii. **Removal of recursive Relationships:**
Changing the M:N recursive Relationships into two 1:M Relationships by introducing an intermediate Entity. (As in Removal of M:N Relationships)

iv. **Removal of Relationships containing Attribues:** There are two cases:
   1. **M:N Relationships:** Introduce an intermediate Entity (As in Removal of M:N Relationships) so that the Relationship Attributes automatically become the Attributes of the new intermediate Entity.
   2. **1:M Relationships:** Shift the Relationship Attributes to one of the two Entities linked by the Relationship.

v. **Removal of multiple value attributes:**
A new Entity, with a 1:M relationship can be defined to remove multiple value attributes.

vi. **Re-examination of 1:1 Relationships:**
Re-evaluate and check existing 1:1 relationships between similar entities, check if they can be combined to form a single Entity.

vii. **Removal of repetitive Relationships:**
   1. **Two Relationships having the same meaning:** Keep only on of the two relationships.
   2. **Derived Relationship:** Relationships that can be abstracted via other relationships can be removed.

c) Abstraction of the Relationship Scheme from the Logical Data Model:

i. Considerations during the process of Abstracting the Relationship:

1. **Strong (Common) Entity:**
   - For each strong Entity, construct a relationship by inserting all the simple attributes (or simple attributes of the composite attributes) of the strong Entity into the relationship.
2. **Weak Entity:**
   - For each strong Entity, construct a relationship by inserting all the simple attributes (or simple attributes of the composite attributes) of the strong Entity into the relationship.
   - Then we have to include an Attribute to represent the foreign Key, which is a copy of the primary Key of the string Entity which is related to this weak Entity.
3. **1:M Relationships:**
   - For each 1:M Relatonship between Entity1 (Parent Entity in position 1) and Entity2 (Child Entity in position M), a copy of the primary Key of Entity1 will be placed in Entity2 as a foreign key.
4. **1:1 Relationships:**
   - For each 1:M Relatonship between Entity1 (Parent Entity with optional participation) and Entity2 (Child Entity with mandatory participation), a copy of the primary Key of Entity1 will be placed in Entity2 as a foreign key. (If both Entities have same participation, we can choose any as parent and the other as child)
5. **Superclass and Subclass Relationships:**
   - For each Superclass/Subclass Relationship, we categorize superclass Entity as the parent Entity and the subclass Entity as the child Entity.
   - Three approaches in mapping to a Relationship scheme are:
     - **Approach 1:** Produce a table for each Entity. Chosen when the superclass Entity has Relationship with other Entities. Suitable for Disjoint super/sub class Relationships.
     - **Approach 2:** Produce a table for each subclass Entity. Chosen when the superclass Entity does not have a Relationship with other Entities. Suitable for Disjoint super/sub class Relationships.
     - **Approach 3:** Stack up all the related attributes into a single table. Suitable for Ovelapping super/sub class Relationships.

9. Normalization

a) Normalization: Method of determining Relationships based on the primary Key or candidate Key and of functional dependency between attributes.

b) Purpose of Normalization: To minimize Excess Data and solve problems of Anomalies of Updating.

c) Anomalies of Updating:

i. **Insertion Anomaly:**

1. The insertion anomaly occurs when a new record is inserted in the relation. When we cannot insert a fact about one entity until we have an additional fact about another entity. Suppose we want to store the information that the cost of car is Rs. 14,00,000, but we cannot enter this data into the relation until the data about the car is entered into the relation. This problem can be solved by dividing the relation into two relations, each one is used to store different facts.
2. It is difficult to insert details of a new branch that has no members of staff into the 'Staff-Branch' relation. The only way to do this is to place null values in the attributes for staff, such as Staff-No. It creates integrity problem because Staff-No is the primary key of Staff-Branch relation. Therefore, you cannot enter a row for a new branch into the Staff-Branch relation with a null value for Staff-No attribute. When Staff-Branch relation is divided into Staff and Branch relation, the above-mentioned problem will not occur.

ii. **Deletion Anomaly:**
Deletion anomaly is the inability to delete unwanted data, without deleting data that you need to retain on the database.

iii. **Modification Anomaly:**
The modification anomaly occurs when the record is updated in the relation. In this anomaly, the modification in the value of specific attribute requires modification in all records in which that value occurs to retain database consistancy.

d) **Functional Dependency:**

- Explains the relations between attributes in a relationship. Defined as follows:
- If A and B are attributes of the H relationship, B is said to be functionally dependent on A (represented as A→B) if each value of A is connected to only 1 value of B.
- In the Relationship H(A,B,C,D,E),
  - If A→B, A→C and A→D, then A→B,C,D
  - But if A,B→D, then it is not true that A→E or B→E

e) **Partial Dependency:** An Attribute being functionally dependent with Part of the primary Key of a relationship.

f) **Transitive Dependency:** If A, B and C are attributes of a relationship to the extent that A→B and B→C, then C also depends transitively on A (A→C) if A is not functionally dependent on B or C.

g) **Normalization Process:** Relationships that contain excess data and anomalies will be divided into several tables which contain the minimum number of anomalies. Steps in normalization are in the below forms:

i. **UNF (Unusual Normalization Form):**
- Table relationship which contains one or more repetitive groups.

ii. **1NF (First Normalization Form):**
- A Relationship which crosses between each row and column containing only one value (fulfilling Relationship characteristics).
- All key Attributes are defined
- All Attributed are dependent on primary Key

ii. **2NF (Second Normalization Form):**
- It is in 1NF
- No Partial Dependencies

iii. **3NF (Third Normalization Form):**
- It is in 2NF
- No Transitive Dependencies

iv. **BCNF (Boyce Codd Normalization Form):**
- It is in 3NF
- Only one cadidate key can determine the table