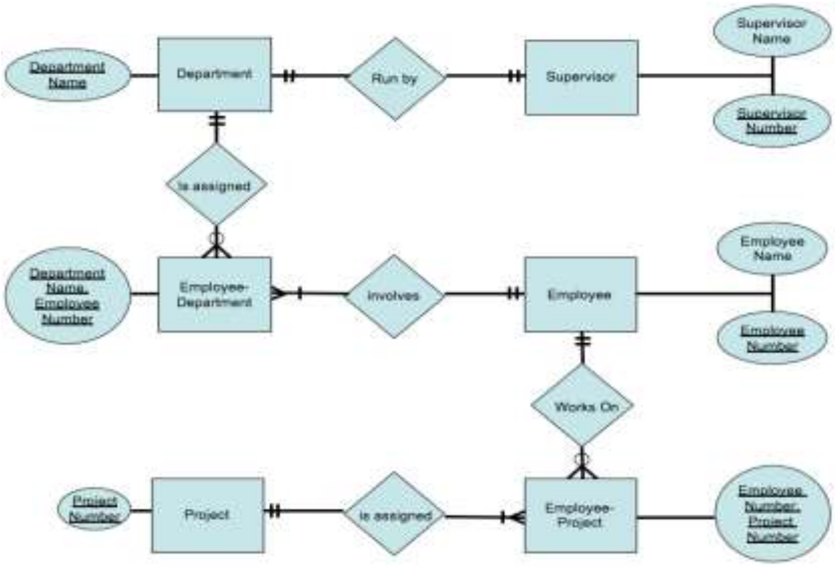
የሥራና ክህሎት ሚኒስቴር
Ministry of Labor and Skills

Web Development and Database Administration Level III

Based on November, 2023 Version-IICurriculum



Module Title: Model Data Object

Module code: EIS WDDBA3 M02 1123

Nominal duration: 50Hours

Prepared by: Ministry of Labor and Skill

November, 2023

Addis Ababa, Ethiopia

Acknowledgment

Ministry of Labor and Skills wish to extend thanks and appreciation to the many representatives of TVET instructors and respective industry experts who donated their time and expertise to the development of this Teaching, Training and Learning Materials (TTLM).

Acronym

DBMS	Database Management System
EDM	Entity Data Model
ER	Entity Relation
ERD	Entity Relation Diagram
ISP	Internet Service Provider
SQL	Structured Query Language

Introduction to the Module

Through this module, you will gain the knowledge and skills necessary to create robust and efficient data models that accurately represent real-world entities and their relationships. The module covers how to analyze business requirements, identify data objects, define attributes, and establish relationships between entities. By the end of this module, you will be equipped with the expertise to design and implement data models that meet the needs of organizations.

This module covers the units:

- Conceptual data model
- Normalization
- Data model validation

This module covers the objectives:

- Identify entities, attributes, data types and relationships
- Develop the ability to design an ER diagram
- Apply normalization rules to tables
- Validate data model

Module Instruction

For effective use of this module trainees are expected to follow the following module instruction:

- Read the specific objectives of this Learning Guide.
- Read the information that this module contain.
- Complete the Self-check.
- Submit your accomplished Self-check.
- Do the operationin the module.
- Do the LAP test(if you are ready) and show your output to your teacher.

Unit One: Conceptual data model

This unit is developed to provide you the necessary information regarding the following content coverage and topics

- Analysis of business data operations
- Scope of the system
- Entities, attributes, data types and relationships of data
- Review business rules
- Documentation of entity relationship diagram

This unit will also assist you to attain the learning outcomes stated in the cover page.

Specifically, upon completion of this learning guide, you will be able to:

- Understand and analyze business operations
- Understand scope of the system
- Identify and define entities, attributes, data types and relationships
- Review business rules to determine impact
- Document relationships in an entity relationship diagram

1.1. Analysis ofbusiness dataoperations

I. Understanding Business operations

Business operations are those ongoing cyclic activities involved in the running of a business for the purpose of producing value.

The outcome of business operations is the *harvesting* of value from assets owned by a business. Assets can be either tangible (*physical)* or *intangible*. An example of value derived from a physical asset like a **building is rent**. An example of value derived from an *intangible asset* like an idea is a royalty.

Business operations encompasses three fundamental management imperatives

1. Generate recurring income
2. Increase the value of the business assets
3. Secure the income and value of the business

II. AnalyzingBusiness operations

- Operational analysis is a business approach that is used to understand and develop operational processes.
- It is a technique of probing into the present and past performance of an operational investment.
- It measures the performance against particular standard costs, operations and services. It also considers how goals can be achieved in a better way, how cost effectively they can be achieved.
- It answers the questions related to the areas of:
  - ➢ **Customer Results**: tries to determine whether the investment is delivering planned goods and services or not.
  - ➢ **Strategic and Business Results**: It analyses whether the current investment level is sufficient to get the job done. It looks at how other organizations are doing this work in a better and more cost-efficient way.

- **Financial Performance**: analyses whether the cost is compatible with the performance.
- **Innovation**: It finds solutions to the questions: How can the customers needs be satisfied in a better way and at a lower cost?; How can technology be best used to provide better services at lower cost? etc.

III. Major Data Analysis methods

a) Text Analysis

Text Analysis is also referred to as Data Mining. It is one of the methods of data analysis to discover a pattern in large data sets using databases or data mining tools. It used to transform raw data into business information. Business Intelligence tools are present in the market which is used to take strategic business decisions. Overall it offers a way to extract and examine data and deriving patterns and finally interpretation of the data.

b) Statistical Analysis

Statistical Analysis includes collection, Analysis, interpretation, presentation, and modeling of data. It analyses a set of data or a sample of data.

c) Diagnostic Analysis

Diagnostic Analysis shows "Why did it happen?" by finding the cause from the insight found in Statistical Analysis. This Analysis is useful to identify behavior patterns of data. If a new problem arrives in into the business process, then you can look into this Analysis to find similar patterns of that problem. And it may have chances to use similar prescriptions for the new problems.

d) Predictive Analysis

Predictive Analysis shows "what is likely to happen" by using previous data. The simplest data analysis example is like if last year I bought two dresses based on my savings and if this year my salary is increasing double then I can buy four dresses.

This Analysis makes predictions about future outcomes based on current or past data. Forecasting is just an estimate. Its accuracy is based on how much detailed information you have and how much you dig in it.

e) Prescriptive Analysis

Prescriptive Analysis combines the insight from all previous Analysis to determine which action to take in a current problem or decision. Most data driven companies are utilizing Prescriptive Analysis because predictive analysis are not enough to improve data performance. Based on current situations and problems, they analyze the data and make decisions.

1.2. Scope of the system

1. Define the System Scope

The scope statement/identification defines what the system/project will and will not include, in enough detail to clearly communicate to all participants. The scope must be a complete definition encompassing all types of requirements:

- functional requirement
- non-functional requirement
- pseudo requirement

The conceptual or scoping model defines the boundaries of the system (i.e., what is in scope and what is out of scope). It identifies:

- Events outside the system that cause the system to react,
- Actors outside the system that interact with the system,
- Information that flows between the system and the actors outside the system,
- Major functions included in the system,
- User population.

I. Clarify System Boundaries

In addition to the scope, it is important that the system boundaries are clearly understood. The boundaries identify where the system to be sized starts and ends. The sizing should include everything for which the team is responsible.

A scope of a system is identified based on the following

- Databases
- Applications
- Servers

- Operating systems
- Gateways
- Application service provider and
- ISP (Internet service provider)

1.3. Entities, attributes, data types and relationships of data

A. Entity

- An entity is an existing or real thing. The fact that something exists also seems to indicate separateness from other existences or entities.

In relation to adatabase , an entity is a single person, place, or thing about which data can be stored. ex. school, student, course, department, employee, university.

- In data modeling (a first step in the creation of a database), an entity is some unit of data that can be classified and have stated relationships to other entities.
- Are **abstract** concepts, each representing one/more instances of a concept
- Considered as container that holds all instances of a particular thing in a system.
- Entities are equivalent to database tables in a relational database, with each row of the table representing an instance of that entity**.**
- The diagram below has an entity for “student” and “school.” This indicates that the system being modeled may contain one or more students and one or more schools.

STUDENT
SCHOOL

Figure 1.1 Entities

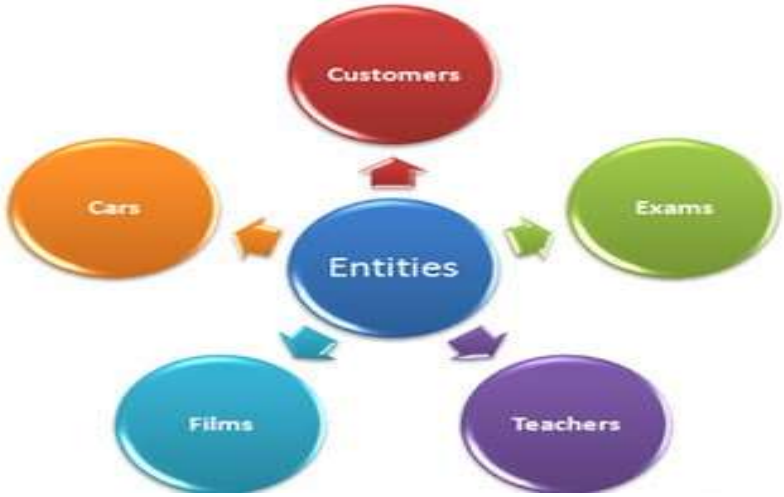


Figure 1.2 Entities

Entity type

- An entity type allows for distinction between the way records are viewed and linked.
- An entity type is a collection of entity instances sharing similar properties;
- Two entity type instances are considered equal only if they are of the same type and the values of their entity keys are the same.
- The *entity type* is the fundamental building block for describing the structure of data with the Entity Data Model (EDM).

Example: The diagram below shows a conceptual model with three entity types: Book, Publisher, andAuthor:

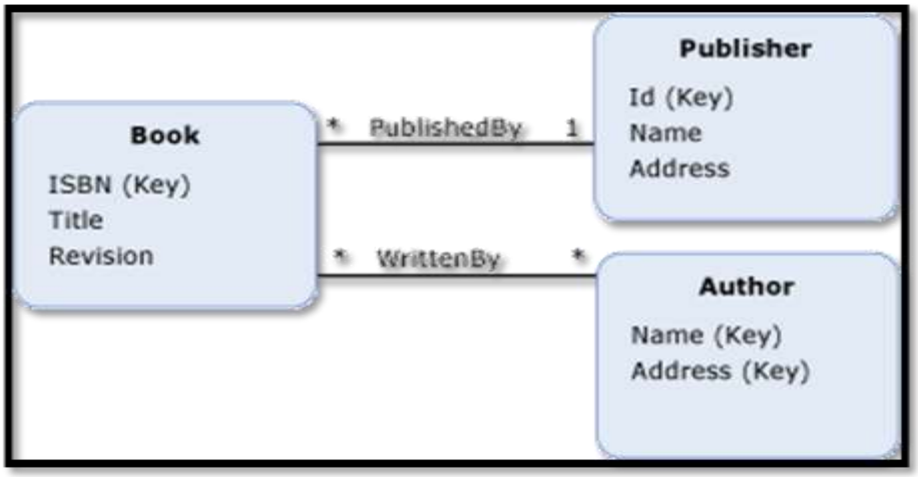


Figure 1.3 Conceptual model

Entity Set

- It is set of entities of the same type (e.g., all persons having an account at a bank)
- An *entity set* is a logical container for instances of an entity type and instances of any type derived from that entity type.
- The relationship between an entity type and an entity set is analogous to the relationship between a row and a table in a relational database:
  - ✓ Like a row, an entity type describes data structure, and,
  - ✓ Like a table, an entity set contains instances of a given structure.
- An entity set provides a construct for a hosting or storage environment (such as the common language runtime or an SQL Server database) to group entity type instances so that they can be mapped to a datastore.

Example

- ***Branch***: the set of all branches of a particular bank. Each branch is described by the attributes w

  Those are*: Branch-name, branch-city and assets.*

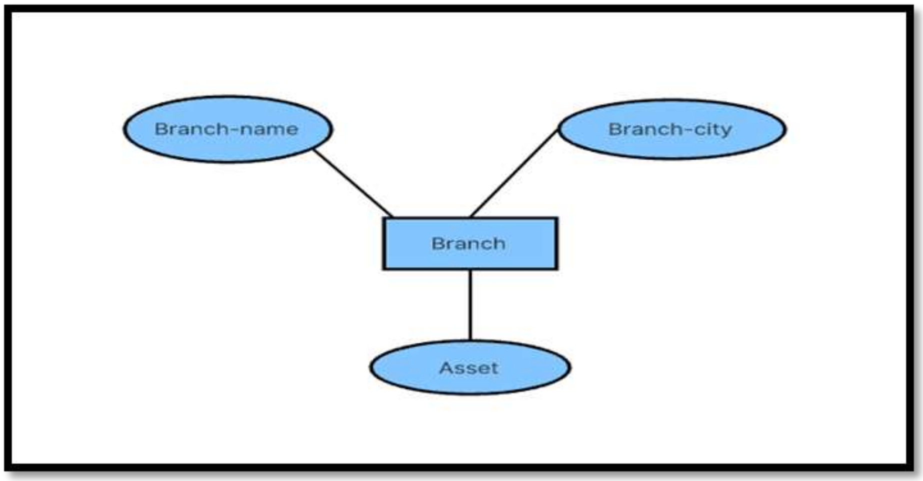


Figure 1.4 Entity examples 1

- ***Customer***: the set of all people having an account at the bank. Attributes are *ID*, *name*, *Gender*, *Gender* and *Phone-number*.

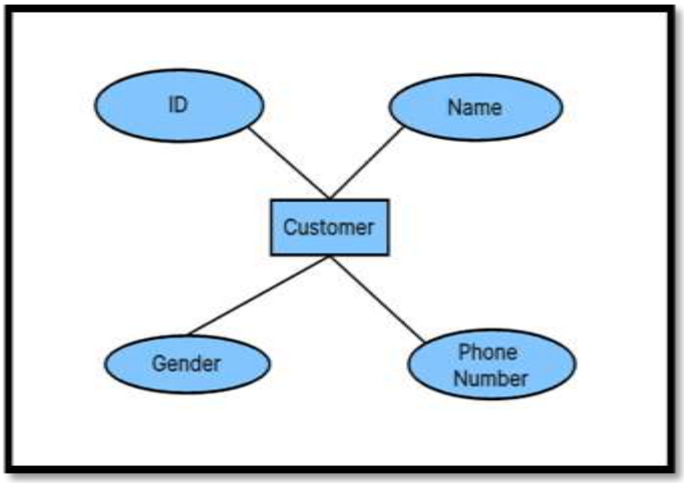


Figure 1.5Entity examples 2

- ***Employee***: with attributes *emp-id*, *name*, *phone-number, gender and age*.

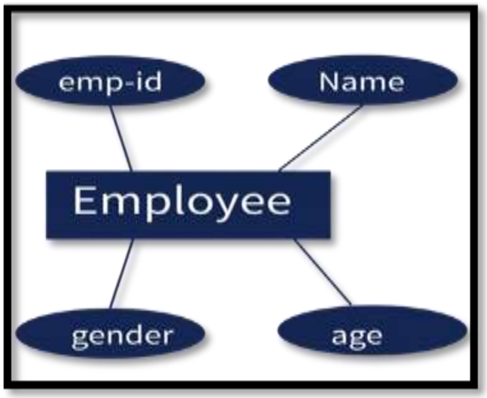


Figure 1.6Entity examples 3

- ***Account***: the set of all accounts created and maintained in the bank. Attributes are *account-number* and *balance*.

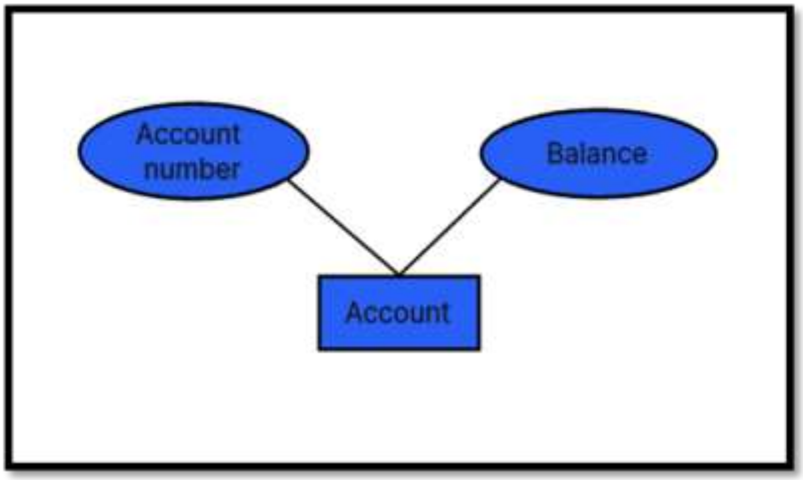


Figure 1.7 Entity examples 4

Classification of Entity:

a. **Strong Entity:** An entity set that has a primary key is termed as strong entity.

b. **Weak Entity**: an entity set that does not have sufficient attributes to form a primary key. The existence of a weak entity depends on the existence of stored entity. The discriminator (partial key) is used to identify other attributes of a weak entity set.

c. **Recursive Entity:** is one in which a relation can exist between occurrences of the same entity set. This occurs in a unary relationship.

d. **Composite Entities:** If a Many to Many relationship exist we must create a bridge entity to convert into 1 to many. Bridge entity composed of the primary keys of each of the entities to be connected. The bridge entity is known as a **composite entity**.

B. Attribute

- A factor/property/characteristic that describes an entity
- In a database management system (DBMS), an attribute may describe a component of the database, such as a table or a field

  Example: (Colour: attribute of your hair/skin/cloth) **Employee's name**, **age**, **address**, **salary** and **job:** attribute of Employee, etc.

Types of Attributes

a. Simple and Composite Attribute

- **Simple attribute**: consist of a single atomic value that can't be subdivided. (For example, age, sex etc.).
- **Composite attribute**: can be further subdivided. (E.g.,**ADDRESS** can be subdivided into city, Sub-city, Woreda, region, House No., etc.)

b. Single Valued and Multi Valued attribute

- **Single valued**: can have only one or a single value. For example, a person can have only one 'date of birth', 'age', etc. But it can be simple or composite attribute. **Example:** 'date of birth' is a composite attribute; 'age' is a simple attribute. But both are single valued attributes.
- **Multi-valued:** can have multiple values. For instance, a person may have multiple phone numbers, multiple degrees etc. Multi-valued attributes are shown by a double line connecting to the entity in the ER diagram.

c. Stored and Derived Attributes

- **Stored attribute**: supplies a value to the related attribute. (e.g., 'Date of birth')
- **Derived attribute**: the value is derived from the stored attribute. (e.g., the value of 'AGE' can be derived by subtracting the 'Date of Birth'(DOB) from the current date.

d. Complex Attribute: attribute that is both composite and multi valued. (e.g., Phone no)

2. Selecting Attributes for Entities: choose ones that have the following qualities:

- **Significant:** Include only attributes that are useful to the database users.
- **Direct**: not derived. Derived data complicates the maintenance of a database.
- **Non-decomposable:** An attribute can contain only single values, never lists or repeating groups. Composite values must be separated into individual attributes.

- **Contain data of the same type:** For example, you would want to enter only date values in a birthday attribute, not names or telephone numbers.

Entity Key

- ✓ A property or a set of properties of an entity type that are used to determine identity.
- ✓ Value of entity key must uniquely identify an entity type instance within an entity set.
- ✓ The properties that make up an entity key should be chosen to guarantee uniqueness of instances in an entity set.

Requirements of entity key:

- No two entity keys within an entity set can be identical. That is, for any two entities within an entity set, the values for all of the properties that constitute a key can't be the same.
- An entity key must consist of a set of non-null, immutable, primitive type properties.
- The properties that make up an entity key can't change. You cannot allow more than one possible entity key for a given entity type; surrogate keys aren't supported.

Types of keys

a. **Super/Candidate Key**: is a field or combination of fields, can act as a primary key for a table to uniquely identify each record. Every entity in relational database must have at least one candidate key but it is possible to have two or more. (**Example**: social security number, employee number or driver license number may identify an employee. All of them are considered candidate keys)

b. **Primary Key:** is an attribute or set of attributes that uniquely identifies one entity from the other. Every entity must have a primary key. It is a candidate key chosen as the main method of uniquely identifying a row.

c. **Alternate key** - is any candidate key which is not selected to be the primary key

d. **Foreign Key**: references a particular attribute of an entity containing the corresponding primary key. These keys are used to create relationships between tables. (For example, an employee entity with employee number as its primary key and department entity with department number as its primary key can be related to each other through employee number. Therefore, employee number will be a foreign key for department and primary key for employee).

e. **Compound/Composite Key**: A Combination of more than one column identifying records of a table uniquely.

3 Key Terms

1. **Conceptual Entity Relationship Diagram:** The highest-level view of the entity relationship diagram, which contains little detail and is solution agnostic. Showing the overall scope of the ERD model from the business perspective. This level of modeling establishes the entities, and their relationships, and defines consistent terminology of the business information.
2. **Logical Entity Relationship Diagram:** Contains mid-level detail. Attributes are introduced and operational, transactional, and business rules are defined in this model. This entity relationship diagram level defines the structure of the data elements and the relationships between them. Logical data models are associated with the solution design.
3. **Physical Entity Relationship Diagram:** Provides the most detail. It can be developed for each logical model. Shows enough detail for subject matter experts to build the physical organization of a database. Physical entity relationship diagrams describe the database-specific implementation of the model and illustrate non-functional requirements such as performance, concurrency, and security.
4. **Database:** A structured collection of information. Usually organized so that data can be easily stored to allow for prompt research, retrieval, and updating.
5. **Adjective:** Attributes that describe or provide details about the entity. For example, a student (noun) might have attributes such as name, age, and address. Note, the term "adjective" is used loosely with the concept of ERDs as many attributes are formally nouns.
6. **Noun (common or proper):** Entity type of person, object, concept, or event. For example, a person entity relevant to school enrollment would be a "student".
7. **Verb:** Relationship types between entities such as enroll. For example, a student (entity) would "enroll" in a course (entity).

C. Data Types

Database data types refer to the format of data storage that can hold a distinct type or range of values. When computer programs store data in variables, each variable must be designated a distinct data type. Some common data types are as follows:

- **Integer** – is a whole number that can have a positive, negative or zero value. It cannot be a fraction nor can have decimal places. It is commonly used in programming especially for increasing values. Addition, subtraction and multiplication of two integers results to an integer. But division of two integers may result to an integer or a decimal. The resulting decimal can be rounded off or truncated to produce an integer.
- **Character** – refers to any number, letter, space or symbol that can be entered in a computer. Each character occupies one byte of space.
- **String** – is used to represent text. It is composed of a set of characters that can have spaces and numbers. Strings are enclosed in quotation marks to identify the data as string and not a variable name nor a number.
- **Floating Point Number** – is a number that contains decimals. Numbers that contain fractions are also considered as floating-point numbers.
- **Array** – contains a group of elements which can be of the same data type like an integer or string. It is used to organize data for easier sorting and searching of related set of values.
- **Varchar** – as the name implies is variable character as the memory storage has variable length. Each character occupies one byte of space plus 2 bytes for length information. *Note: Use Character for data entries with fixed length, like phone number. Use Varchar for data entries with variable length, like address.*
- **Boolean** – is used for creating true or false statements. To compare values the following operators are being used: AND, OR, XOR, and NOT.

D. Relationship

- It is an association between entities**,** captures how entities are related to one another. Relationships can be thought of as verbs, linking two or more nouns.
- A relationship is how the data is shared between entities.
- Are represented by lines between entities, lines indicate that each instance of an entity may have a relationship with instances of the connected entity, and vice versa.

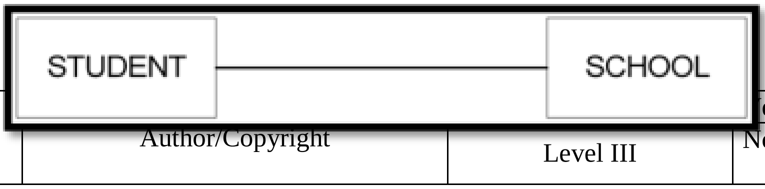


Figure 1.8 Entity relation

The diagram above indicates that students may have some relationship with schools. More specifically, there may be a relationship between a particular student (an instance of the student entity) and a particular school (an instance of the school entity).

If necessary, a relationship line may be labeled to define the relationship. In this case, one can infer that a student may attend a school, or that a school may enroll students.

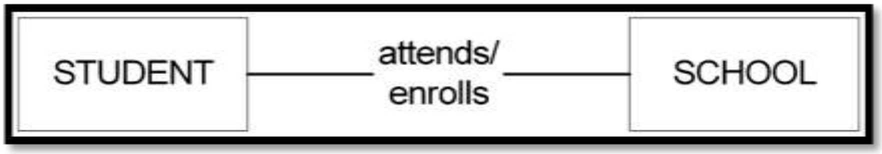
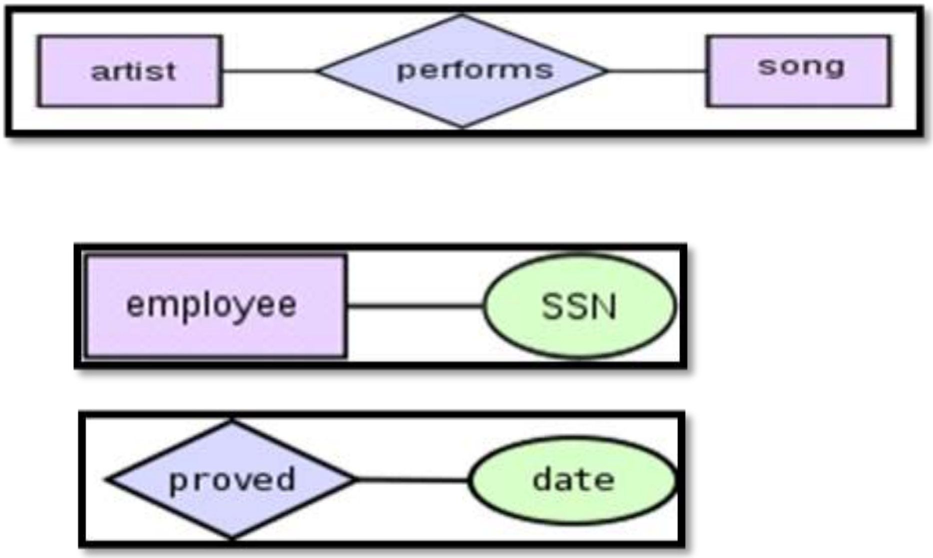


Figure 1.8 Entity relations with verb

Relationship and Entity: can both have attributes. **Examples**: an *employee* entity might have a *Social Security Number* (SSN) attribute; the *proved* relationship may have a *date* attribute.



Two related entities

An entity with an attribute

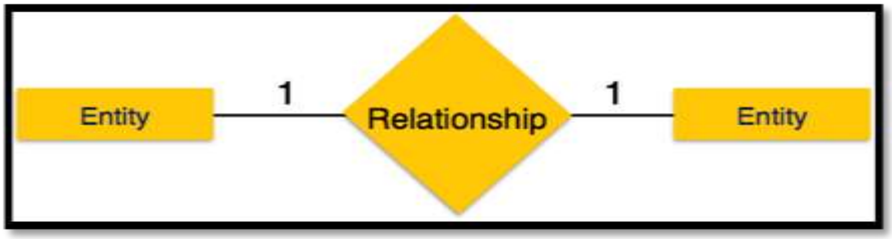
A relationship with an attribute

Figure 1.9 Relation with entity

There are four types of relationships between entities:

Cardinality: Defines the numerical attributes of the relationship between two entities or entity sets

- **One-to-one (1:1)**: one instance of an entity (A) is associated with one other instance of another entity (B). For example, in a database of employees, each employee name (A) is associated with only one social security number (B).



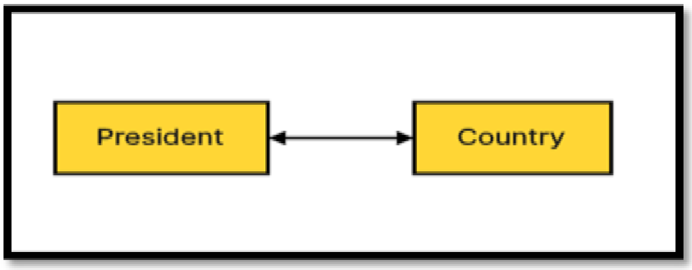
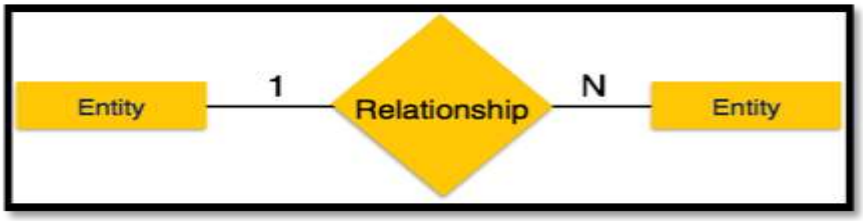


Figure 1.10 one to one relations

- **One-to-many (1: N):** is a hierarchical relationship created or viewed from the primary entity. Any one entity instance from the primary entity can be referenced by many entity instances from the related entity. One instance of an entity (A) is associated with zero, one or many instances of another entity (B), but for one instance of entity B there is only one instance of entity (A). Example, for a company with all employees working in one building, the building name (A) is associated with many different employees (B), but those employees all share the same singular association with entity



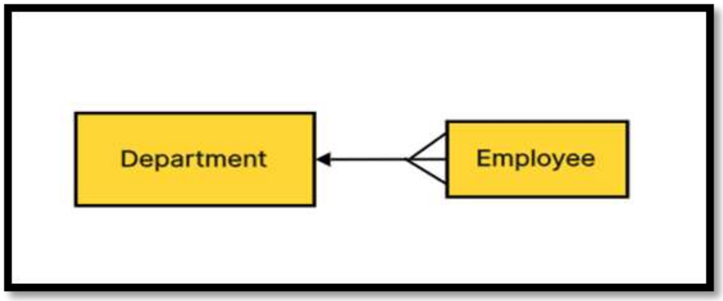
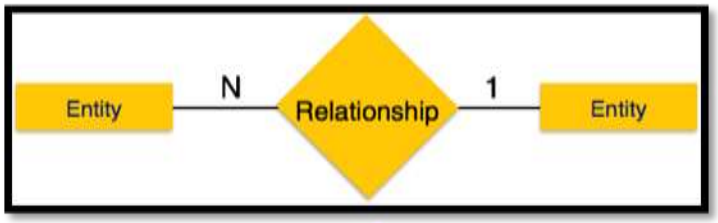


Figure 1.11 one to many relations

- **Many-to-one** (**N: 1):** is a hierarchical relationship created or viewed from the related entity. Many entity instances from the related entity can reference any one entity instance from the primary entity. Remember that the same relationship can be viewed from either of the two entities that participate in the relationship.



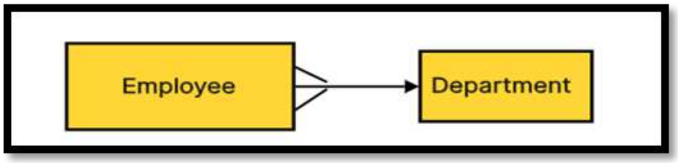
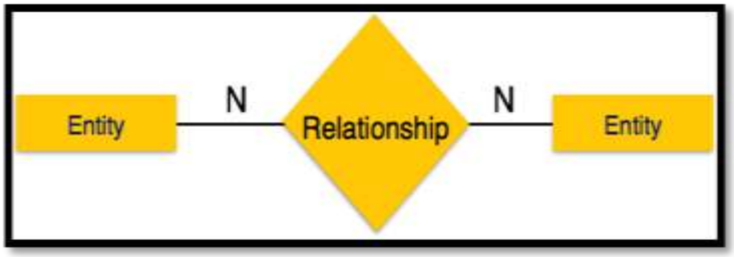


Figure 1.13 many to one relation

- **Many-to-many (N: N)**: A many-to-many relationship lets users relate one or more entity instances from another entity to an entity instance of the current entity. A many-to-many relationship is **reciprocal**. Therefore, entity instances can be related from either entity. One instance of an entity (A) is associated with one, zero or many instances of another entity (B), and one instance of entity B is associated with one, zero or many instances of entity A. For example, for a company in which all of its employees work on multiple projects, each instance of an employee (A) is associated with many instances of a project (B), and at the same time, each instance of a project (B) has multiple employees (A) associated with it.



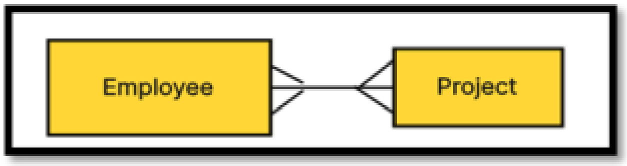


Figure 1.13 many to many relations

1.4. Review business rules

I. Business rule

- It is a rule of a business, company, or corporation that defines or constrains some aspect of business and always resolves to either true or false.
- It is a statement that defines or constrains some aspect of the business, intended to assert business structure, to control/influence behavior of the business.
- Describes the operations, definitions and constraints that apply to an organization.
- Tells an organization *what* it can do in detail, provides detailed guidance about how a strategy can be translated to action.
- Can apply to people, processes, corporate behavior and computing systems in an organization, and are put in place to help the organization achieve its goals.
- While a business rule may be informal or even unwritten, writing the rules down clearly and making sure that they don't conflict, is a valuable activity.
- When carefully managed, rules can be used to help the organization to better achieve goals, remove obstacles, reduce costly mistakes, improve communication, comply with legal requirements, and increase customer loyalty.

  Example: rent rules, payment rules, service rules, attendance rules, product rules, etc.

II. Categories of business rules

- **Definitions of business terms:** The most basic element of a business rule is the language used to express it. The very definition of a term is itself a business rule that describes how people think and talk about things.
- **Facts relating terms to each other:** The nature or operating structure of an organization can be described in terms of the facts that relate terms to each other. To say that a customer can place an order is a business rule. Facts can be documented as natural

language sentences or as relationships, attributes, and generalization structures in a graphical model.

- **Constraints ("action assertions"):** Every enterprise constrains behavior in some way, and this is closely related to constraints on what data may or may not be updated. To prevent a record from being made is, to prevent an action from taking place.
- **Derivations:** Business rules (including laws of nature) define how knowledge in one form may be transformed into other knowledge, possibly in a different form.

1.5. Documentation of entity relationship diagram

1. Overview of data modeling:

A data model provides the details of information to be stored, and is of primary use when the final product is the generation of computer software for an application. It is an abstract model that documents and organizes the business data for communication between team members and is used as a plan for developing applications, specifically how data are stored and accessed.

A data model is a way finding tool for both business and IT professionals, which uses a set of symbols and text to precisely explain a subset of real information to improve communication within the organization and thereby lead to a more flexible and stable application environment. It determines the structure of data or *structured data*. Typical applications of data models include database models, design of information systems and enabling exchange of data

2. Entity – Relationship Diagram (ERD)

An ERD, or entity relationship diagram, is a type of flowchart that helps you clearly visualize your database design by showing how the "entities" in the system relate to one another. It is an abstract way to describe adatabase. It usually starts with a relational database, which stores data in tables.

It is a visual representation of different data using conventions that describe how these data are related to each other. **For example**, the elements writer, novel, and consumer may be described using ERD this way:

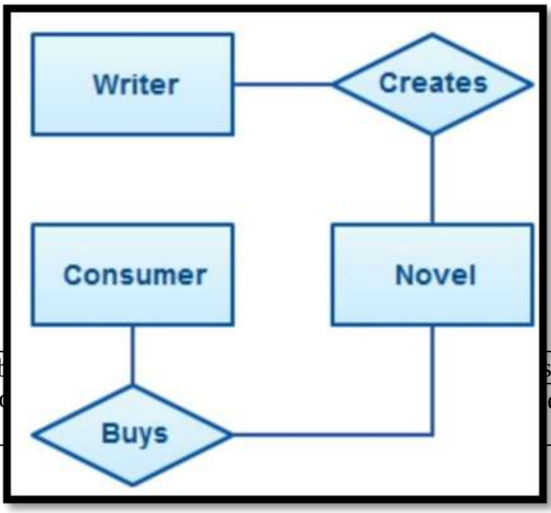
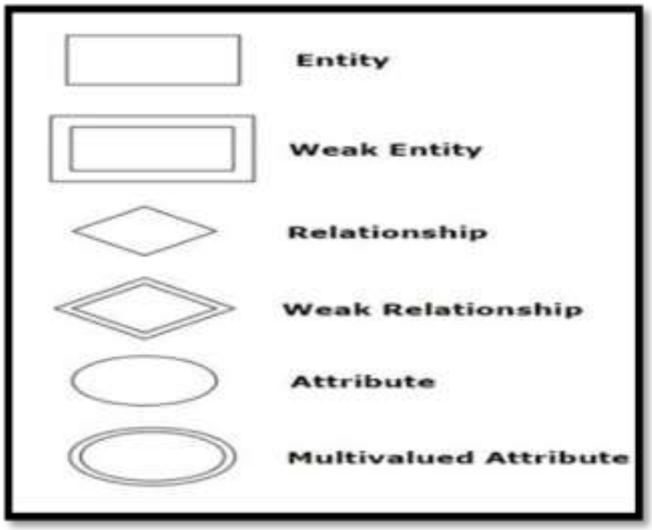


Figure 1.14 ERD

In the diagram, the elements inside rectangles are called entities while the items inside diamonds denote the relationships between entities.

3. Entity relation diagram symbols



4. ER Diagrams Usage

ER is able to describe just about any system, ER diagrams are most often associated with complex databases that are used in software engineering and IT networks.

In particular, ER diagrams are frequently used during the design stage of a development process in order to identify different system elements and their relationships with each other. **For example**: an inventory software used in a retail shop will have a database that monitors elements such as purchases, item, item type, item source and item price. Rendering this information through an ER diagram would be something like this:

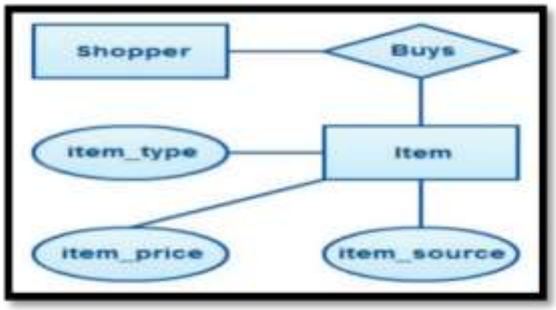


Figure 1.14 ERD

In the diagram, the information inside the oval shapes is attributes of a particular entity.

There are three basic elements in an ER Diagram: entity, attribute, relationship. There are more elements which are based on the main elements. They are weak entity, multi-valued attribute, derived attribute, weak relationship and recursive relationship.

- **Entity:** An entity can be a person, place, event, or object that is relevant to a given system. For example, a school system may include students, teachers, major courses, subjects, fees, and other items. Represented in ERD by a rectangle and named using singular nouns.
- **Weak Entity:** is an entity that depends on the existence of another entity. In more technical terms it can defined as an entity that can't be identified by its own attributes. It uses a foreign key combined with its attributed to form the primary key.

Example: The order item will be meaningless without an order so it depends on the existence of order.

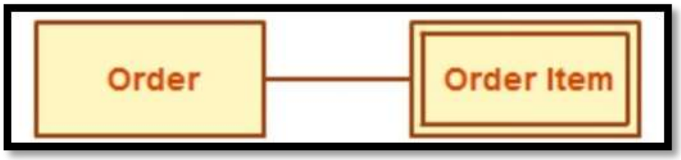
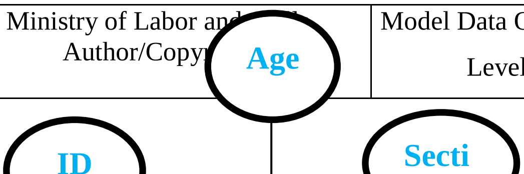


Figure 1.15 Weak entity

- **Attribute**: An attribute is a property, trait, or characteristic of an entity, relationship, or another attribute. An entity can have as many attributes as necessary. Attributes are



represented by oval shapes. (For example: a student entity may have attributes such as Name, Roll no and Age)

Figure 1.16 Attribute

- **Composite attributes:** attributes having their own specific attributes. **For example:** the attribute “customer address” can have the attributes number, street, city, and state.

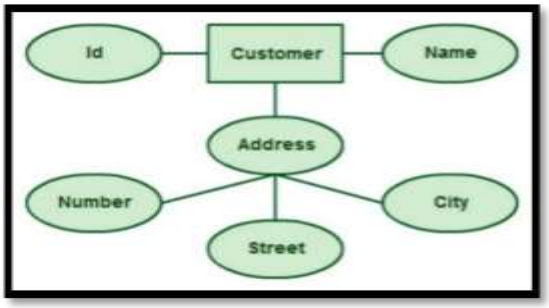


Figure 1.17 Composite attribute

- **Multi-valued Attribute:** If an attribute can have more than one value. It is important to note that this is different to an attribute having its own attributes. For example, a teacher entity can have multiple subject values.

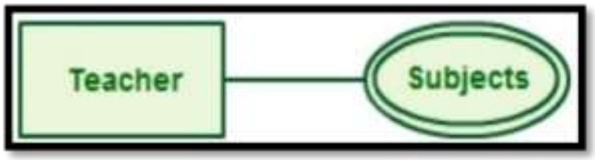


Figure 1.18 Multi-valued attribute

- **Derived Attribute:** An attribute based on another attribute, found rarely in ER diagrams. For example, for a circle the area can be derived from the radius.

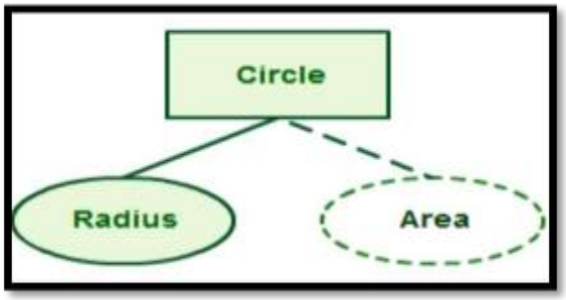


Figure 1.19 Derived attribute

- **Relationship:** A relationship describes how entities interact. For example, the entity “carpenter” may be related to the entity “table” by the relationship “builds” or “makes”. Relationships are represented by diamond shapes and are labeled using verbs.

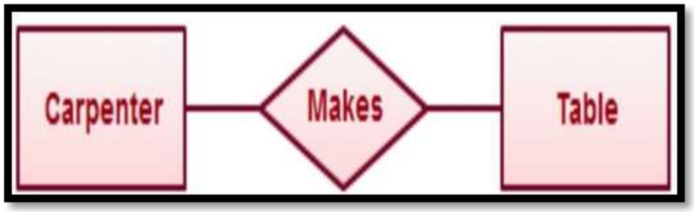


Figure 1.20 Relationships

- **Recursive Relationship**: If the same entity participates more than once in a relationship it is known as recursive relationship. In the below example an employee can be a supervisor and be supervised, so there is a recursive relationship.

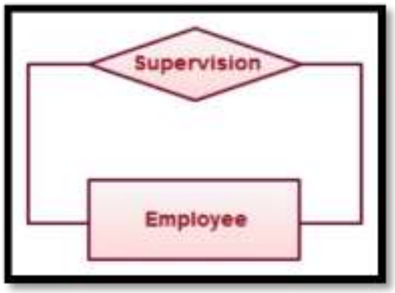
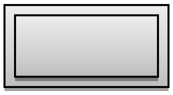
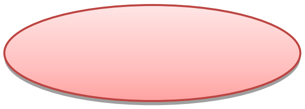


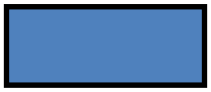
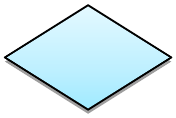
Figure 1.21 Recursive relationships

Self -Check 1

Part I: Choose the best answer

1. Select one of the following which is not true about Entity.
   a) An existing or real thing
   b) A single person, place, or thing about which data can be stored.
   c) Are abstract concepts, each representing one/more instances of a concept
   d) Are equivalent to database tables in a relational database
   e) None
2. Which one of the following is an example of entity?
   a) Address b) Salary c) Computer d) Name
3. Which one of the following is an example of attribute?
   a) Book b) Customer c) IDNO d) Student
4. One of the following is an example of derived attribute
   a) Date of birth b) Age c) Name d) ID
5. One of the following symbols represents an attribute diagrammatically in ERD

a)  b) 

c)  d) 

6. The following can be a one-to-one relation ship
   a) Relation between Department and employees
   b) Relation between a president and a country
   c) Relation between Employee and project
   d) Relation between Employees and department
7. Which one of the following is an example of One-to-Many relationship?
   a) Student – to – course c) Student – to – Instructor
   b) Department – to – Student d) Husband – to – Wife
8. In ERD, entities are represented by

a) Nouns & Diamond c) Verbs & Diamond

b) Verbs & Rectangle d) Nouns & Rectangle

9. In ERD, relationships are represented by

a) Nouns & Diamond c) Verbs & Diamond

b) Verbs & Rectangle d) Nouns & Rectangle

10. A method of data analysis which analyses a set of data or a sample of data is:

a) Diagnostic Analysis c) Prescriptive Analysis

b) Predictive Analysis d) Statistical Analysis

Part-II: Fill the blank space

1) __________defines what the system/project will and will not include.
2) A statement that defines or constrains some aspect of the business is known as__________
3) A property that determines identity is__________
4) Ongoing cyclic activities involved in the running of a business is called _____________

_____________is a type of entity in which a relation can exist between occurrences of the same entity set.

Part-III: Answer the following questions briefly

1. Why is defining the scope important before implementing a model data object?
2. What information does an ERD convey about the entities and relationships within a model data object?
3. How are attributes and data types related to the entities in a model data object?
4. What is the purpose of documenting an entity relationship diagram (ERD) in the context of a model data object?

Operation Sheet-1.1 Install Microsoft Visio 2010

Operation title: InstallMicrosoft Visio 2010

Purpose: To install Microsoft Visio 2010

Equipment Tools and Materials: Microsoft Visio 2010

Steps in designing the diagram:

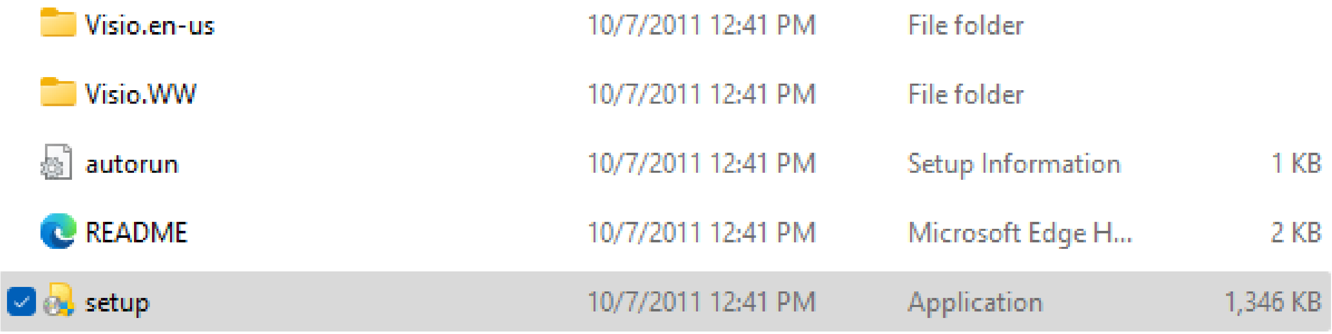
Step 1: Before installing Microsoft Visio 2010 make sure your PC meets minimum system requirements.

- Operating System: Windows 7or above.
- Memory (RAM): 256MB of RAM or above.

- Hard Disk Space: 2GB of RAM or above.
- Processor: 500MHz processor or faster.

Microsoft Visio 2010 software installation

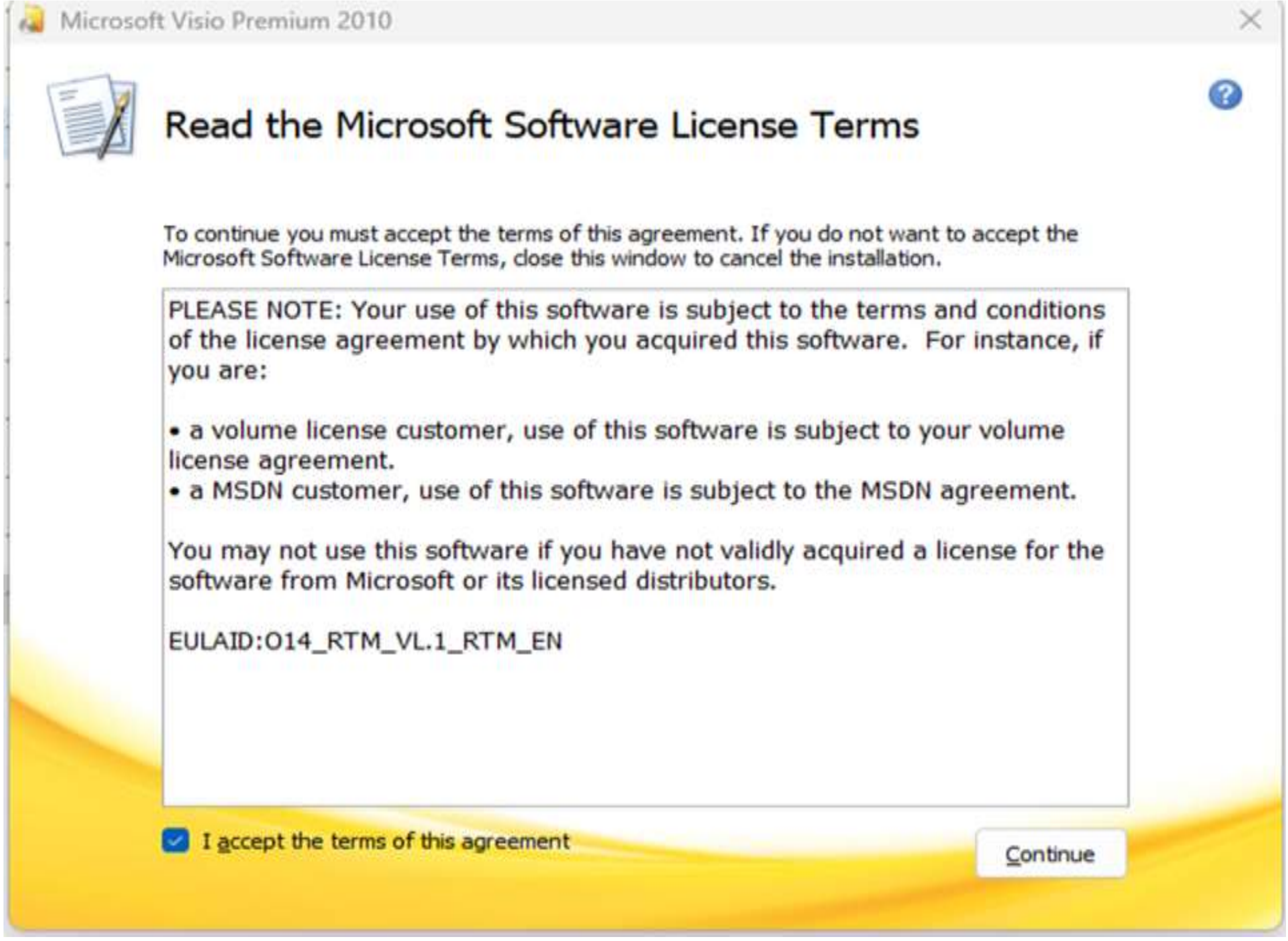
Step 2: Open installation media in new window and right click on setup file to run it “As Administrator”.



Visio.en-us	10/7/2011 12:41 PM	File folder	
Visio.WW	10/7/2011 12:41 PM	File folder	
autorun	10/7/2011 12:41 PM	Setup Information	1 KB
README	10/7/2011 12:41 PM	Microsoft Edge H...	2 KB
☑ setup	10/7/2011 12:41 PM	Application	1,346 KB

Step 3: Accept terms

Accept the terms of the agreement and continue



Read the Microsoft Software License Terms

To continue you must accept the terms of this agreement. If you do not want to accept the Microsoft Software License Terms, close this window to cancel the installation.

PLEASE NOTE: Your use of this software is subject to the terms and conditions of the license agreement by which you acquired this software. For instance, if you are:

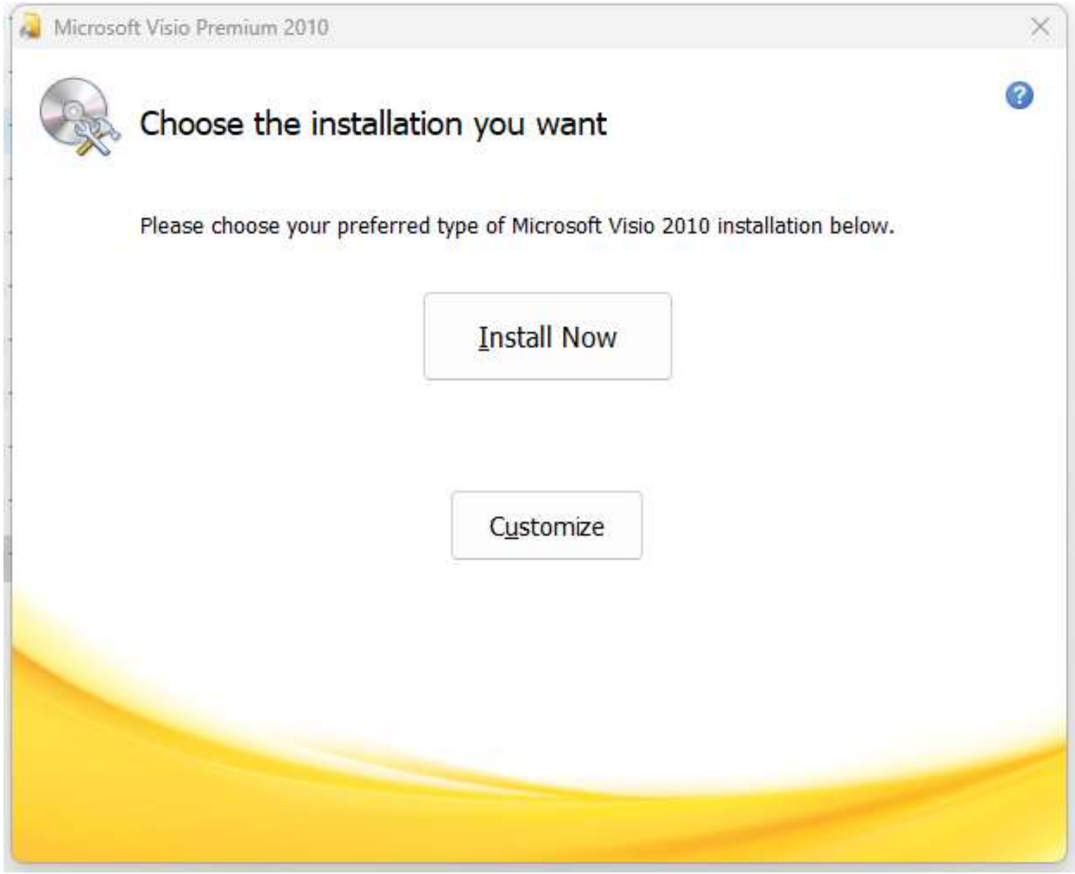
- a volume license customer, use of this software is subject to your volume license agreement.
- a MSDN customer, use of this software is subject to the MSDN agreement.

You may not use this software if you have not validly acquired a license for the software from Microsoft or its licensed distributors.

EULAID:O14_RTM_VL.1_RTM_EN

☑ I accept the terms of this agreement

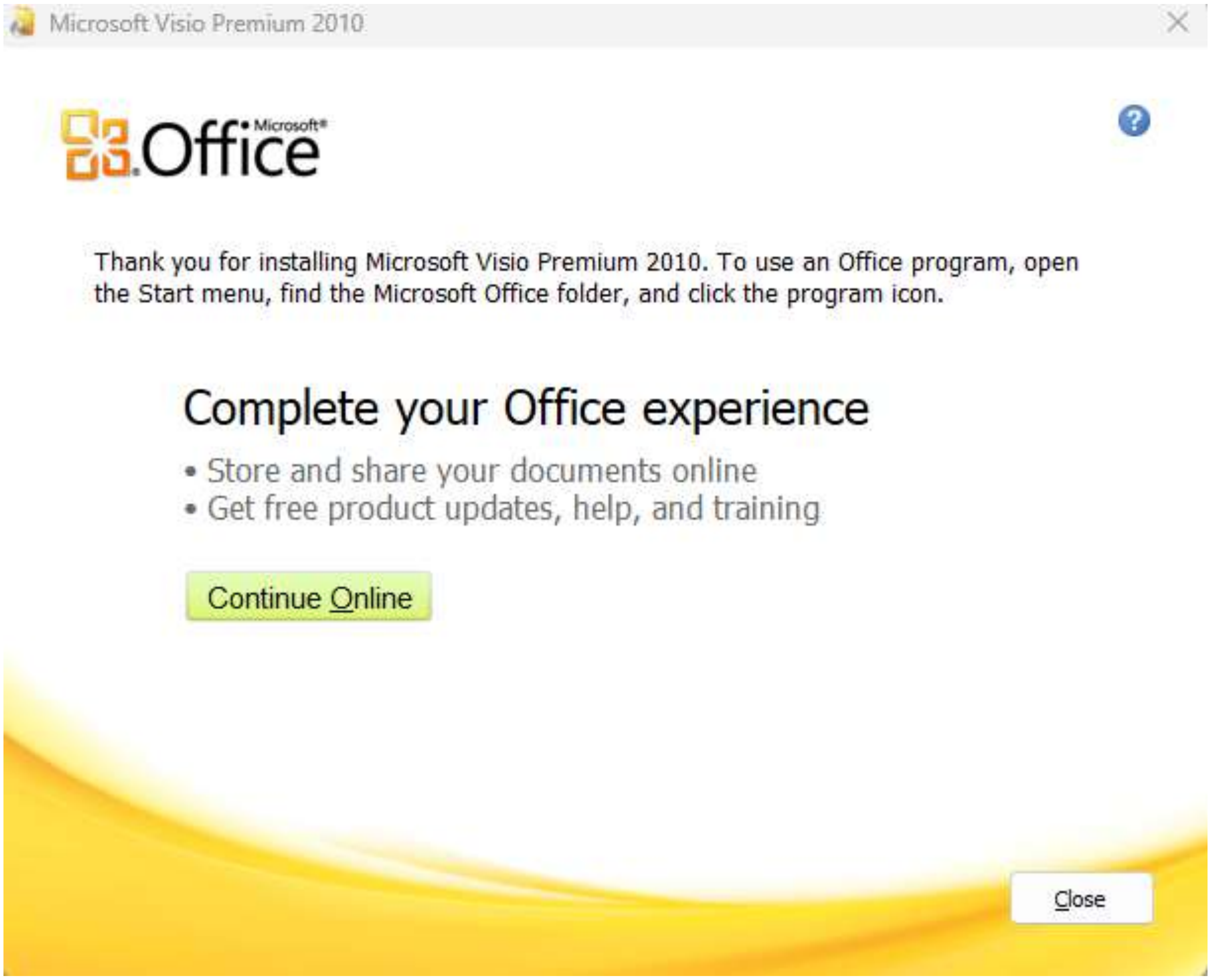
Step 4: choose install now option if you have no prior version of Microsoft visio or customize if you have previous version.



Choose the installation you want

Please choose your preferred type of Microsoft Visio 2010 installation below.

Step 5: After the installation progress is finished you can start designing your ER diagram



Microsoft Visio Premium 2010

Thank you for installing Microsoft Visio Premium 2010. To use an Office program, open the Start menu, find the Microsoft Office folder, and click the program icon.

Complete your Office experience

- Store and share your documents online
- Get free product updates, help, and training

Operation Sheet-1.2Data Modeling

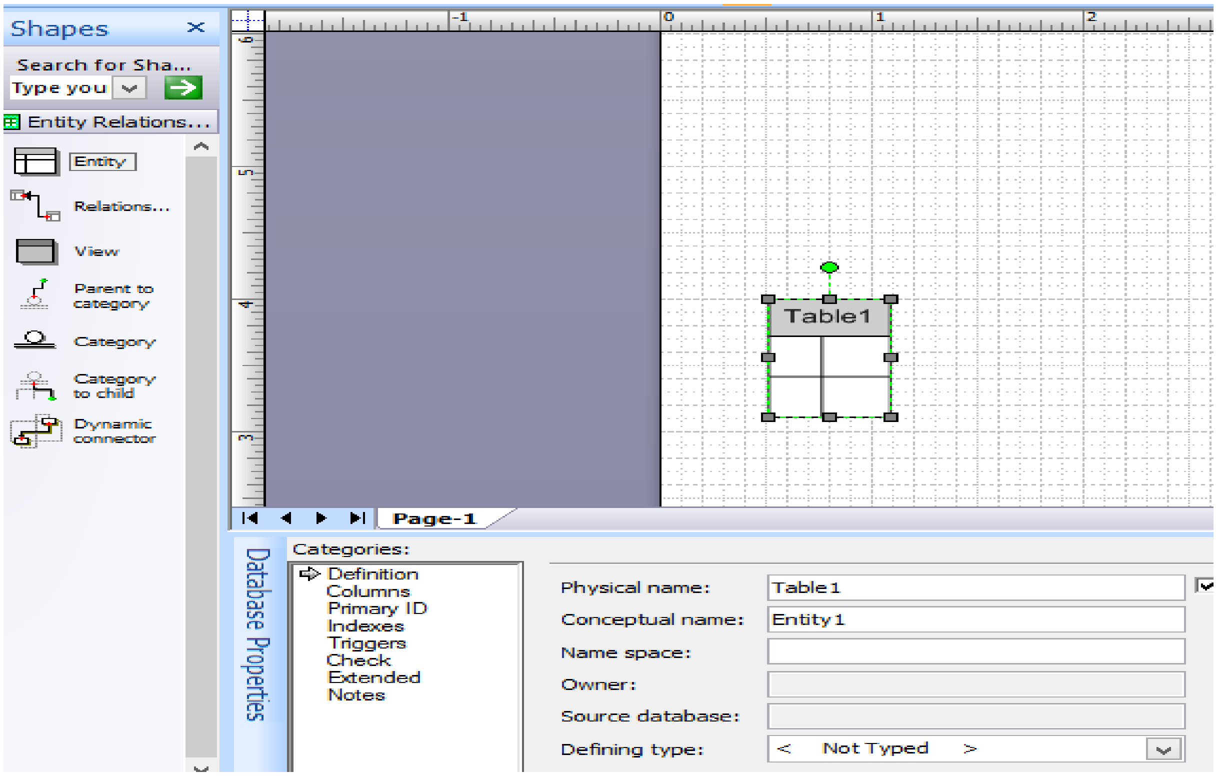
Operation title: Data Modeling

Purpose: To identify entities, attributes and relationships and perform data modeling properly.

Equipment Tools and Materials: Microsoft Visio 2010.

Step: - 1.

1. Click on start button → All program→Microsoft office→Microsoft Visio→File→New→Software Database→Database Model Diagram (US units)→ select Entity under shape



Physical name: Table1

Conceptual name: Entity1

Name space:

Owner:

Source database:

Defining type: < Not Typed >

Step - 2. Add table and columns

Click on Entity shape “Table1” →click on Physical Name and rename to table name Eg. “Employee”

To add column click on “columns” under categories and assign physical name and data types and other constraints such as (primary key and foreign keys)

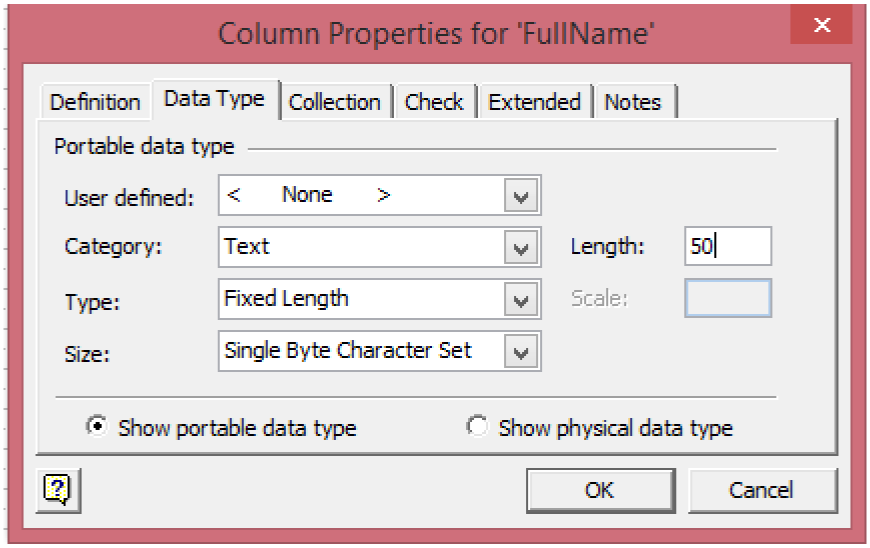
- Database schema

Employee Table

Column name	Data type	Size	Constraints	Key
Empid	Char	10	Not null	Primary key
FullName	Char	50		
Sex	Char	10		
Address	Char	5		

To change the size of data type of employee “FullName” from default one (10) to 50 please

Click on Data type of required data (Full Name)→Edit→select data type→ Length then write number of size→ok



Portable data type

User defined: < None >

Category: Text

Length: 50

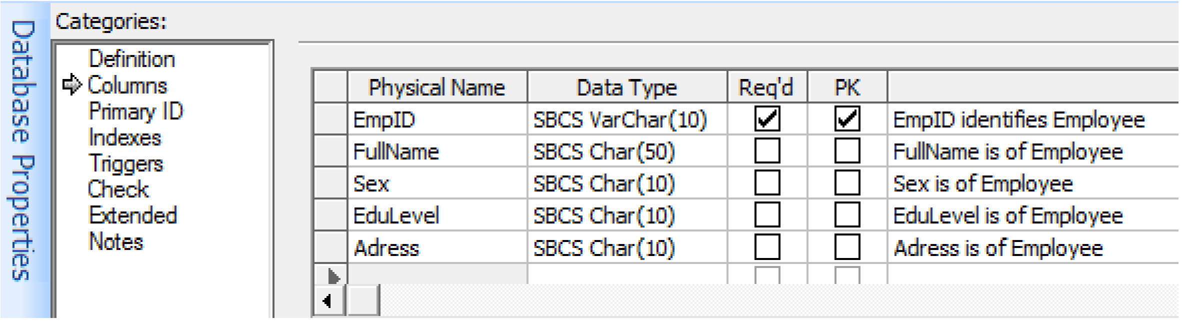
Type: Fixed Length

Scale:

Size: Single Byte Character Set

● Show portable data type ○ Show physical data type

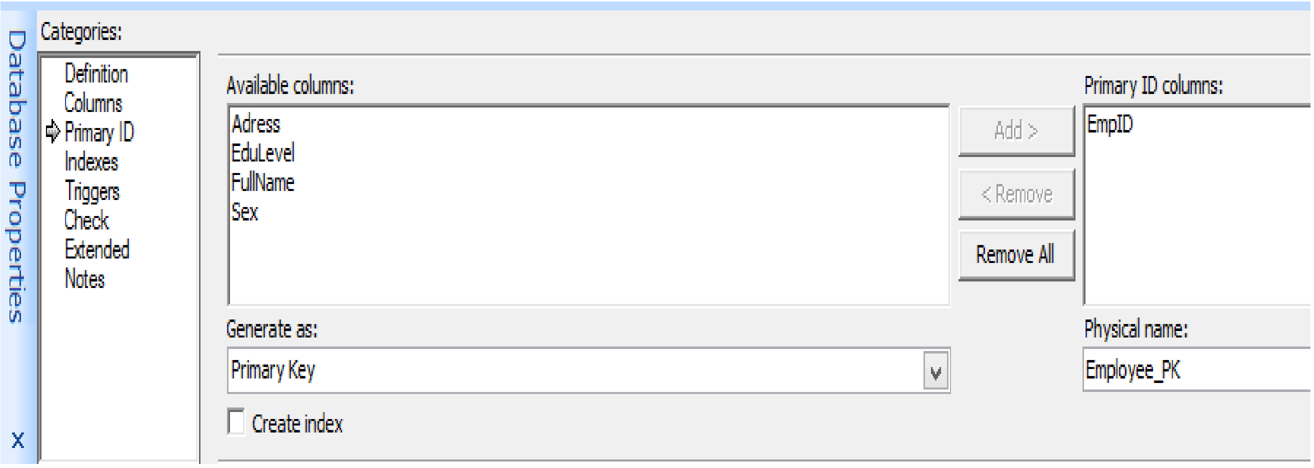
Also we can add columns as the below procedures



Physical Name	Data Type	Req'd	PK	
EmpID	SBCS VarChar(10)	☑	☑	EmpID identifies Employee
FullName	SBCS Char(50)	□	□	FullName is of Employee
Sex	SBCS Char(10)	□	□	Sex is of Employee
EduLevel	SBCS Char(10)	□	□	EduLevel is of Employee
Adress	SBCS Char(10)	□	□	Adress is of Employee

To set primary key

Click on primary ID under categories→



Available columns:
Adress
EduLevel
FullName
Sex

Primary ID columns:
EmpID

Generate as: Primary Key

□ Create index

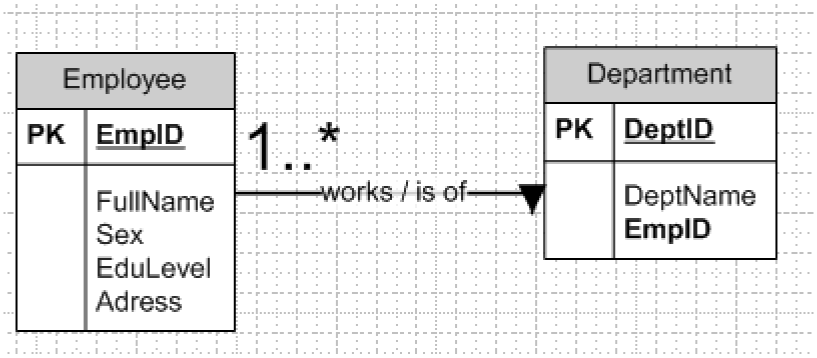
Physical name: Employee_PK

step- 3 Add related table and identify cardinality ratios

Column name	Data type	Size	Constraints	Key
DeptID	Char	10	Not null	Primary key
DeptName	Char	50		
EmpID	Char	10		Foreign key

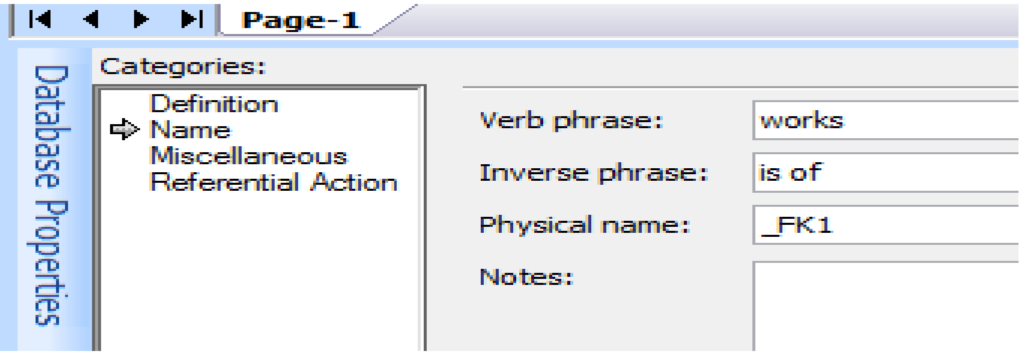
To create relationship between both Employee and Department

Select relationship under shape→



To check cardinality ratios

Click on Arrow between both tables→click on Name under categories to set relationship which is by (word)



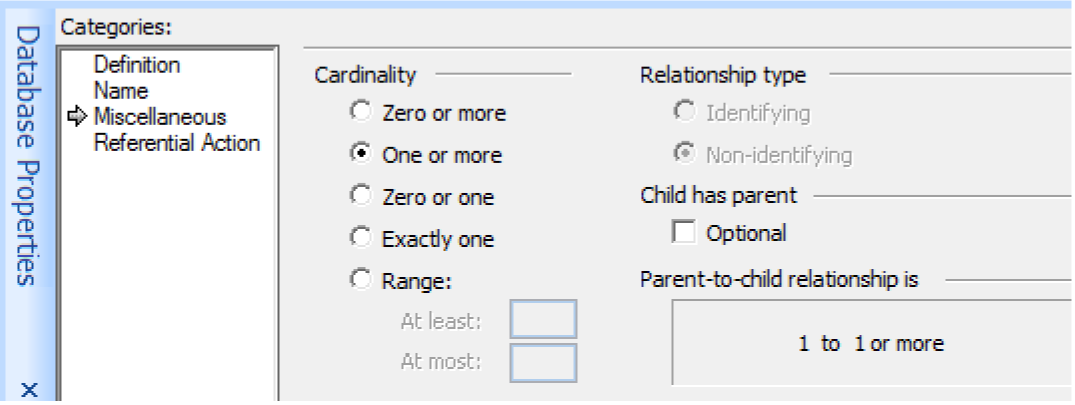
Verb phrase: works

Inverse phrase: is of

Physical name: _FK1

Notes:

To check cardinality ratios
Click on →Miscellaneous →



Database Properties

Categories:

Definition
Name
Miscellaneous
Referential Action

Cardinality
○ Zero or more
● One or more
○ Zero or one
○ Exactly one
○ Range:
At least:
At most:

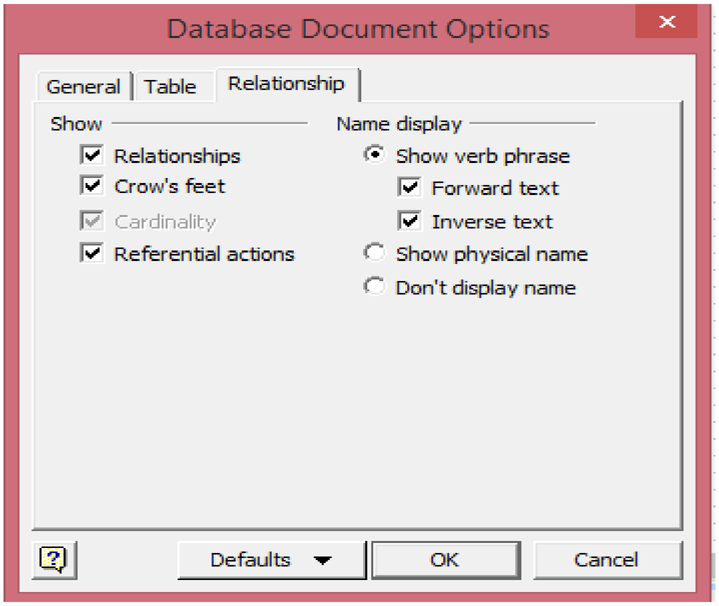
Relationship type
○ Identifying
● Non-identifying

Child has parent
□ Optional

Parent-to-child relationship is
1 to 1 or more

Step-4To add necessary information under relation such as (words, cardinality ratios) follow the following procedures.

Select Database (from menu bar)→Option→Document→Relationship→Ok



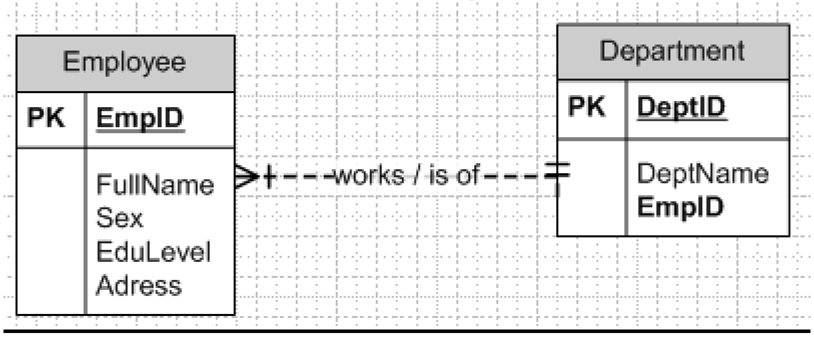
Database Document Options

General | Table | Relationship

Show
☑ Relationships
☑ Crow's feet
☑ Cardinality
☑ Referential actions

Name display
● Show verb phrase
☑ Forward text
☑ Inverse text
○ Show physical name
○ Don't display name

The output of the procedure is looks like the below figure.



Operation Sheet-1.3Design ER diagram

Operation title: Design ER diagram

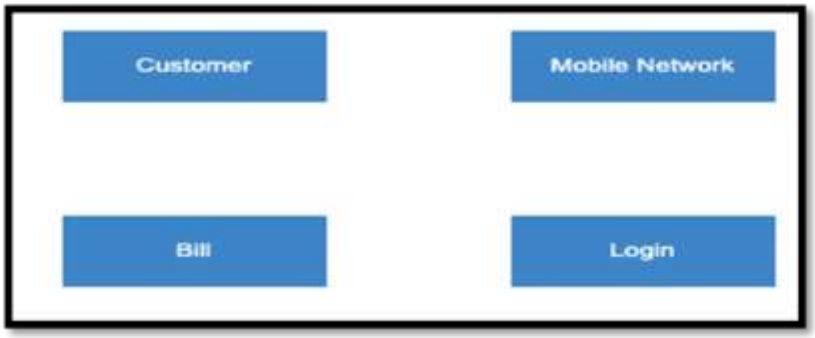
Purpose: To draw a simple ER diagram for Mobile network registration by identifying entities, attributes with their relationships

Equipment Tools and Materials: Microsoft Visio 2010

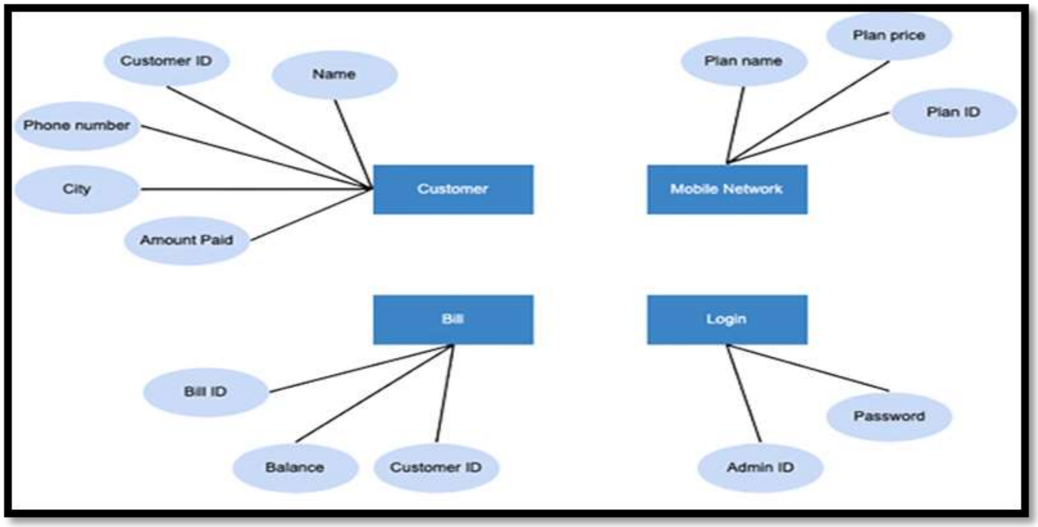
Steps in designing the diagram:

Step 1: The first step in designing ERD is to start by identifying what's in your system or architecture. (Define each entity) and you'll want to give them plenty of room so that you can add to your diagram in the next steps.

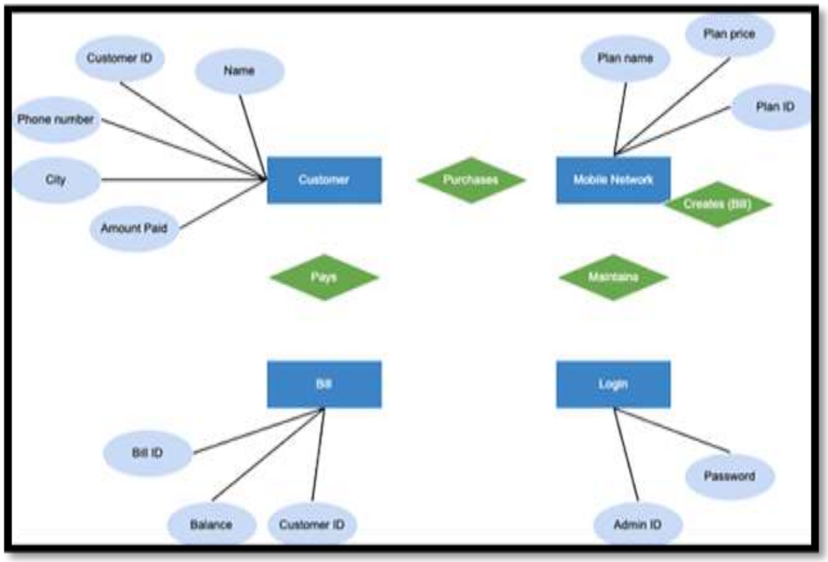
- The possible entities when in designing a mobile network registration is a customer, a mobile network, bill and login.



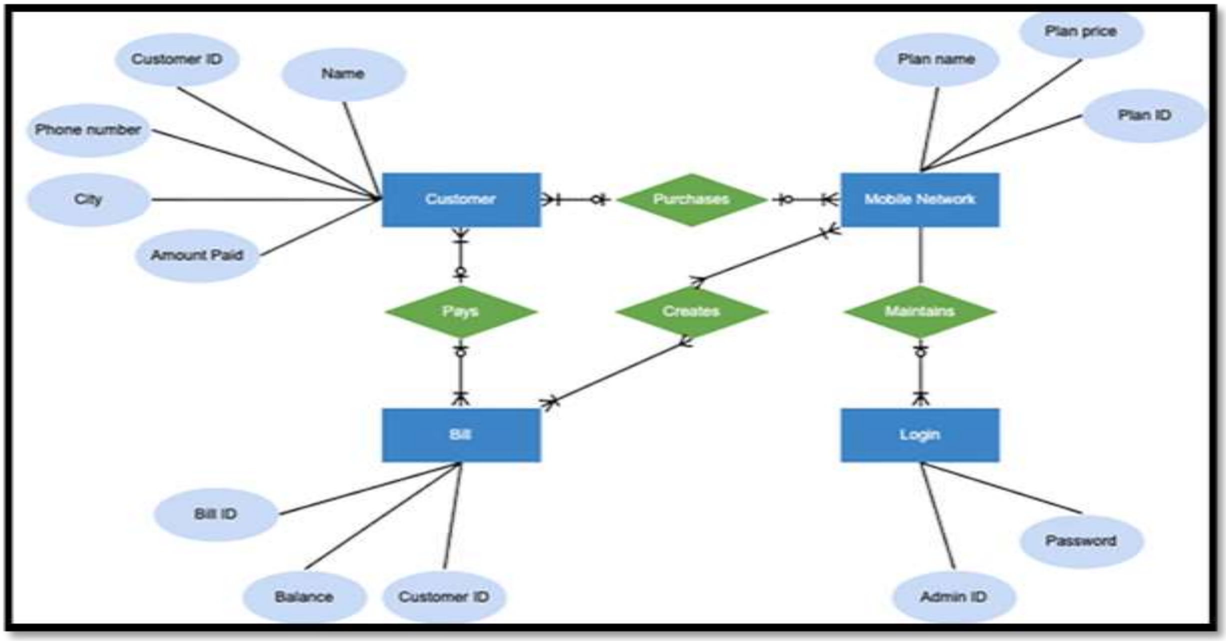
Step 2: Consider or identify the attributes that you need to describe each entity.draw them inside ovals. Connect these to the relevant entity and position your attributes to the outside of your diagram, which leaves room for relationships.



Step 3: Think through the relationships or verbs taking place within the system. customer *purchases* the phone. The cell service *maintains* the phone. The cell service *creates* a bill. The customer *pays* the bill.



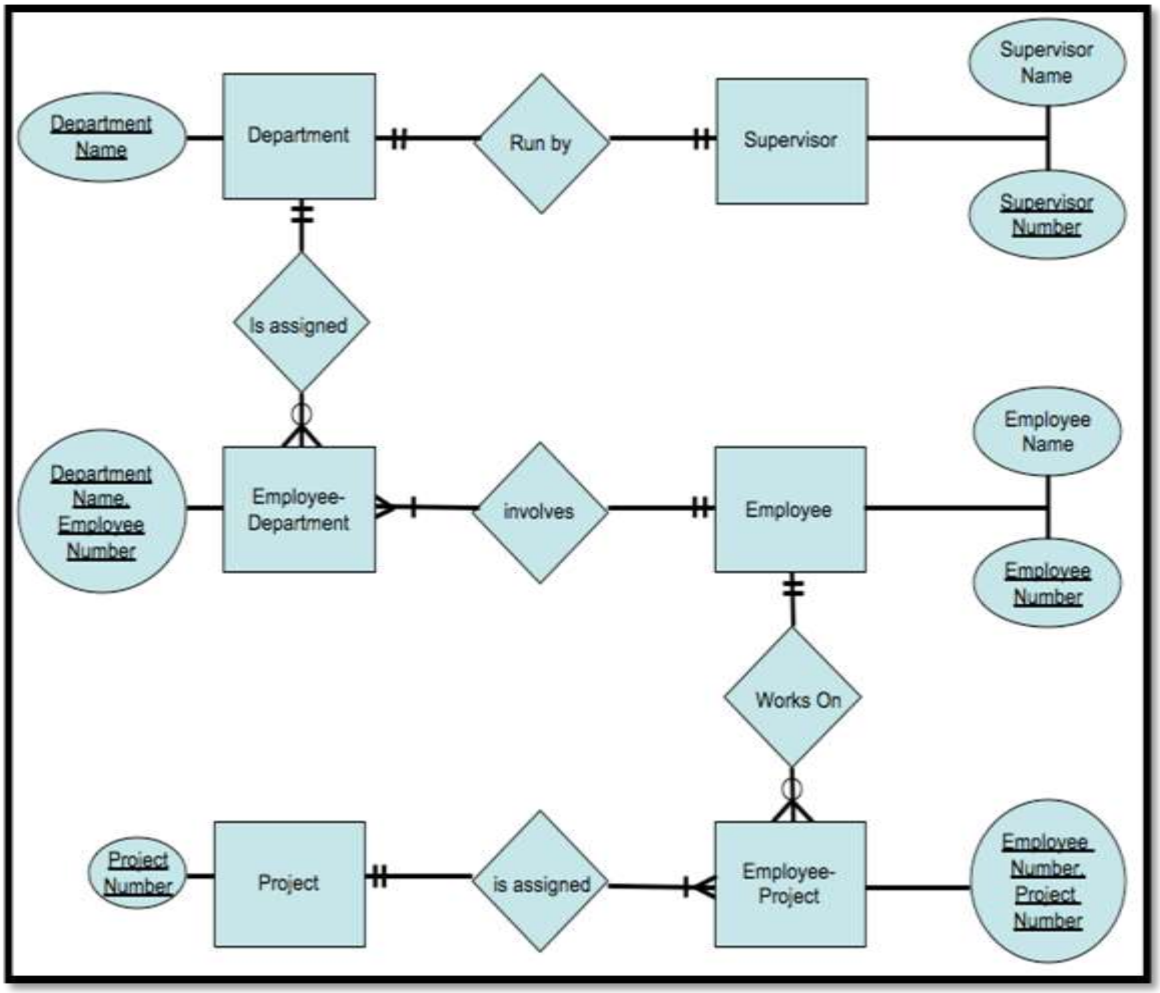
Step 4: Final step for this simple ER diagram is to define the amount of data that will come from each entity. Your customer can purchase *one or many* phones. The cell service maintains *many* phones. The customer pays *one* bill.



Lap Test

Instruction: Design the below ER diagram by using the required software material.

Task 1: Simple Employee database



Task 2: Hospital database with a set of patients and a set of medical doctors.

Task 3: Simple library management system.

Task 4: Simple car rental system.

Unit Two: Normalization

This unit is developed to provide you the necessary information regarding the following content coverage and topics

- Identification of suitable business data
- Rules of normalization
- Normalize business data and document results
- Compare normalization results with ER diagram
- Reconcile differences between data

This unit will also assist you to attain the learning outcomes stated in the cover page. Specifically, upon completion of this learning guide, you will be able to:

- Understand and analyze business data
- Understandrules of normalization
- Understand benefits of normalization
- Contrast normalization results with ER diagram
- Understand how to conform differences between data

2.1. Identify Suitable Business Data

When developing normalization for a database, it is important to identify suitable business data that accurately represents the entities and relationships within the system. Here are some steps to help identify suitable business data for normalization:

1. Understand the Business Processes: Gain a deep understanding of the business processes and operations that the database will support. This involves analyzing the workflow, data flows, and interactions between different entities within the organization. Identify the key entities, their attributes, and the relationships between them.
2. Conduct Stakeholder Interviews: Engage with stakeholders, such as business owners, managers, and end-users, to gather their input on the data requirements. Conduct interviews to understand their needs, pain points, and how they interact with the data. This will help identify the critical data elements and relationships that need to be captured in the database.
3. Review Existing Documentation: Examine any existing documentation, such as business requirements documents, process flowcharts, or data dictionaries. This can provide valuable insights into the data elements that are already identified or documented. Reviewing existing documentation will help ensure that no important data elements are overlooked during the normalization process.
4. Analyze Sample Data: Analyze sample data sets to identify the different types of data that need to be captured. This can be done by reviewing existing spreadsheets, forms, reports, or any other sources of data. Identify the unique values, common patterns, and potential data dependencies within the sample data. This analysis will help determine the appropriate entities, attributes, and relationships that should be included in the normalized database.
5. Identify Key Business Rules: Understand the business rules that govern the data and its relationships. These rules define the logic, constraints, and dependencies that shape the data. Identify any unique constraints, mandatory fields, or conditional relationships that need to be captured in the database design.
6. Consider Future Growth and Scalability: Anticipate future growth and scalability requirements of the database. Identify data elements that are likely to change or expand over time. This includes considering potential new features, additional data sources, or changes in business processes. By planning for future growth, you can ensure that the normalized

database can accommodate evolving business needs without significant redesign or restructuring.

7. Collaborate with Database Experts: Seek guidance from database experts or experienced database administrators. They can provide valuable insights into best practices for normalization and help identify suitable business data based on their expertise and knowledge.

Remember, the goal of normalization is to eliminate redundancy and improve data integrity. By carefully identifying suitable business data, you can create a well-structured and efficient database that accurately represents the business processes and supports the organization's goals.

2.2. Understand Rules of Normalization

2.2.1 Normalization

- It is a technique for producing a set of relations with desirable properties, given the data requirements of an enterprise.Database designed based on ER model may have some amount of inconsistency, ambiguity and redundancy. To resolve these issues, some amount of refinement is required. This refinement process is called as **Normalization**.
- The process of normalization is a formal method that identifies relations based on their primary key.Primarily it is a tool to validate and improve a logical design so that it satisfies certain constraints that ***avoid unnecessary redundancy of data.***
- It is the process of decomposing relations with anomalies to produce smaller, ***well-structured*** relations and helps eliminate data anomalies/problems

2.2.2 Benefits of database normalization

- Reduced usage of storage space by intelligently categorizing data.
- It enables better, faster, stronger searches as it entails fewer entities to scan in comparison with the earlier searches based on mixed entities.
- Improves data integrity: it splits all the data into individual entities yet building strong linkages with the related data.
- More efficient database structure.
- Better understanding of data.
- More flexible database structure.
- Easier to maintain database structure.

- Few (if any) costly surprises down the road.
- Validates your common sense and intuition.
- Avoids redundant fields.
- Ensures that distinct tables exist when necessary

2.2.3 Anomalies in Normalization

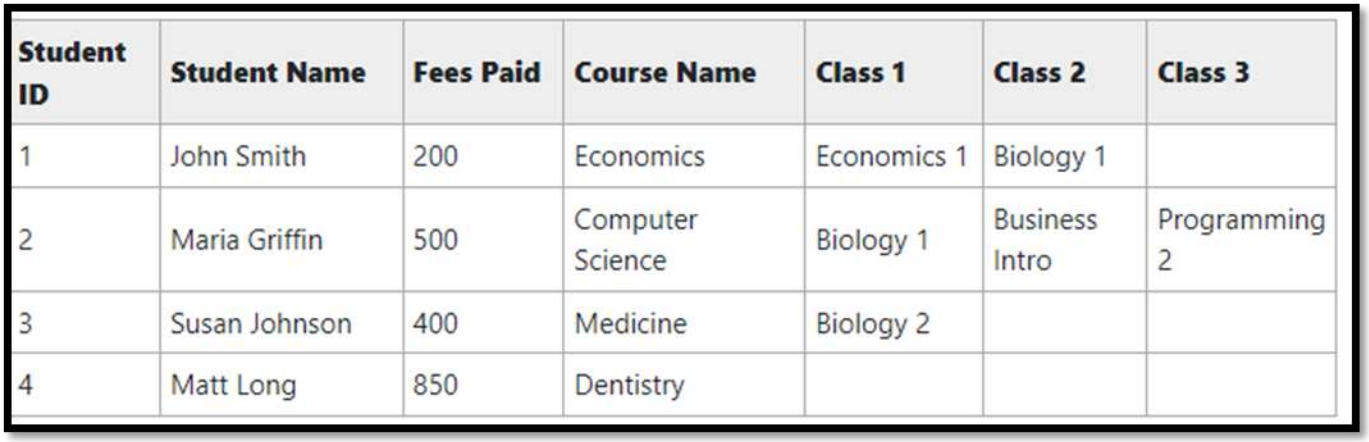
Anomalies in normalization refer to inconsistencies or issues that can occur in a database when it is not properly normalized. These anomalies can affect data integrity, accuracy, and the ability to perform efficient data operations. There are three main types of anomalies that can occur:

1. **Insertion Anomaly**: An insertion anomaly happens when it is not possible to insert a new record into a table without including additional, unrelated data. This occurs when a table has attributes that are functionally dependent on only a part of the primary key.
2. **Update Anomaly**: An update anomaly occurs when modifying data in a table leads to inconsistencies or redundant updates in other parts of the table. This happens when a table has redundant data or dependencies between non-key attributes.
3. **Deletion Anomaly**: A deletion anomaly occurs when removing data from a table unintentionally removes other related data that should have been preserved. This happenswhen a table has dependencies between attributes, and removing data leads to the loss of other necessary data.

Normalization helps to eliminate these anomalies by organizing data into well-structured tables, ensuring data dependencies are properly defined, and reducing redundancy. By achieving higher levels of normalization, such as 3NF or BCNF, the likelihood of anomalies occurring is minimized, and data integrity is improved

Example

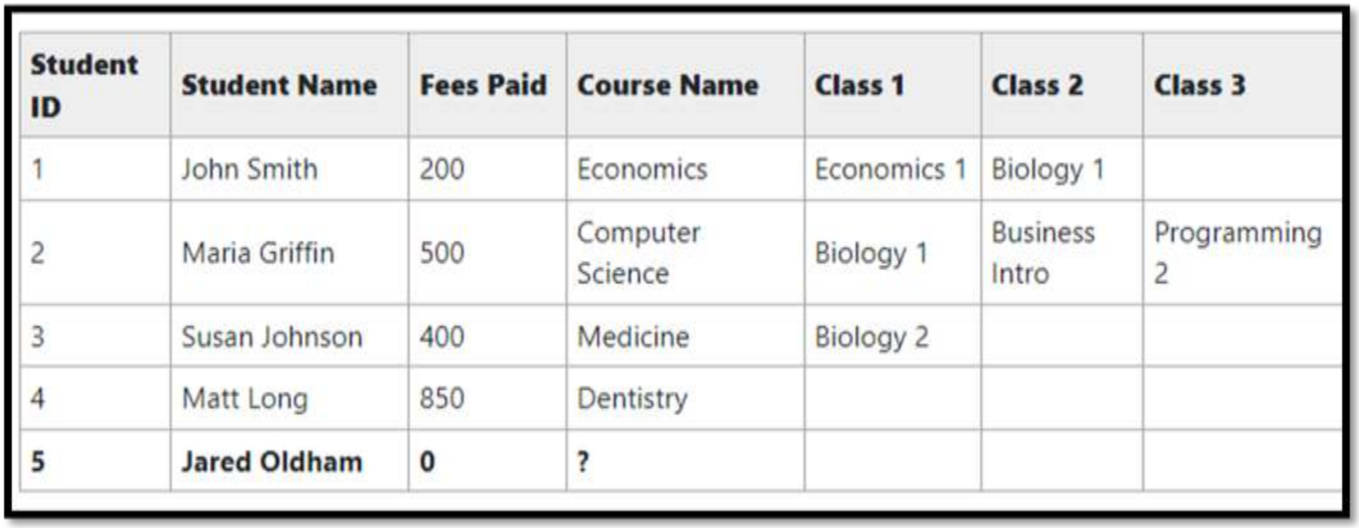
We’ll be using a student database as an example in this article, which records student, class, and teacher information.



Student ID	Student Name	Fees Paid	Course Name	Class 1	Class 2	Class 3
1	John Smith	200	Economics	Economics 1	Biology 1	
2	Maria Griffin	500	Computer Science	Biology 1	Business Intro	Programming 2
3	Susan Johnson	400	Medicine	Biology 2		
4	Matt Long	850	Dentistry			

Insert Anomaly

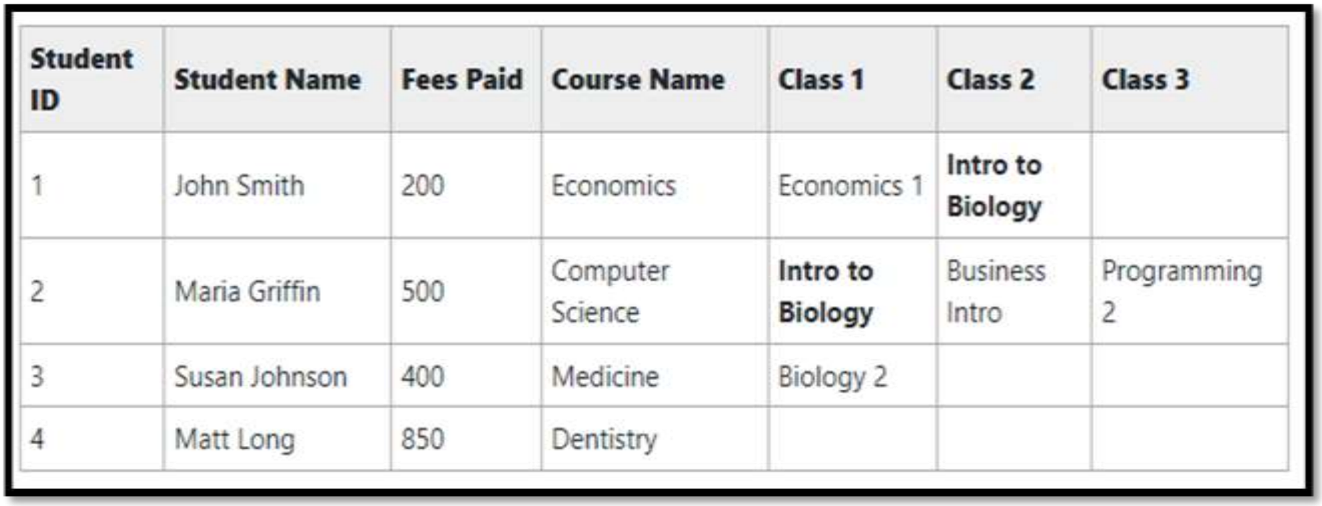
For example, if we wanted to add a new student but did not know their course name this will be how



Student ID	Student Name	Fees Paid	Course Name	Class 1	Class 2	Class 3
1	John Smith	200	Economics	Economics 1	Biology 1	
2	Maria Griffin	500	Computer Science	Biology 1	Business Intro	Programming 2
3	Susan Johnson	400	Medicine	Biology 2		
4	Matt Long	850	Dentistry			
5	**Jared Oldham**	**0**	**?**			

Update Anomaly

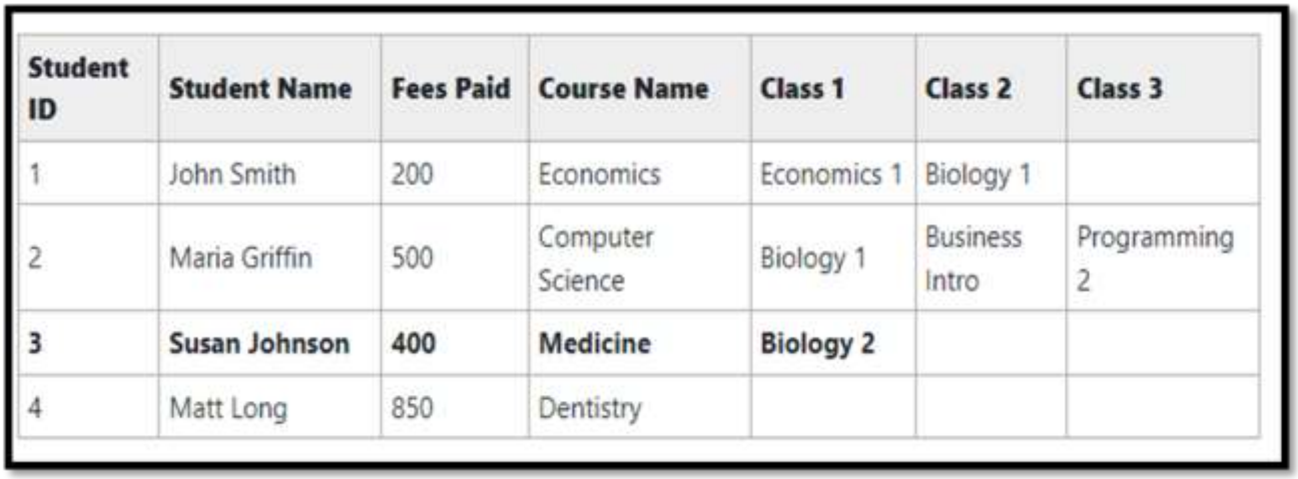
For example, let's say the class Biology 1 was changed to "Intro to Biology". We would have to query all of the columns that could have this Class field and rename each one that was found.



Student ID	Student Name	Fees Paid	Course Name	Class 1	Class 2	Class 3
1	John Smith	200	Economics	Economics 1	**Intro to Biology**	
2	Maria Griffin	500	Computer Science	**Intro to Biology**	Business Intro	Programming 2
3	Susan Johnson	400	Medicine	Biology 2		
4	Matt Long	850	Dentistry			

Delete Anomaly
For example, let’s say Susan Johnson quits and her record needs to be deleted from the system.

We could delete her row:



Student ID	Student Name	Fees Paid	Course Name	Class 1	Class 2	Class 3
1	John Smith	200	Economics	Economics 1	Biology 1	
2	Maria Griffin	500	Computer Science	Biology 1	Business Intro	Programming 2
3	**Susan Johnson**	**400**	**Medicine**	**Biology 2**		
4	Matt Long	850	Dentistry			

Normalization stages

- 1NF - First normal form
- 2NF - Second normal form
- 3NF - Third normal form
- 3.5NF - Boyce Codd Normal Form (BCNF)
- 4NF - Fourth normal form
- 5NF - Fifth normal form
-

The Normal Forms

The database community has developed a series of guidelines for ensuring that databases are normalized. These are referred to as normal forms and are numbered from one (the lowest form of normalization, referred to as First normal form or 1NF) through five (fifth normal form or 5NF). In practical applications, you'll often see 1NF, 2NF, and 3NF along with the occasional 4NF. Fifth normal form is very rarely seen.

I. First normal form (1NF):Eliminating Repeating Groups

- Is the "basic" level of normalization, corresponds to the definition of any database
- It contains two-dimensional tables with rows and columns.
- Each column corresponds to a sub-object or an attribute of the object represented by the entire table.
- Each row represents a unique instance of that sub-object or attribute and must be different in some way from any other row (no duplicate rows are possible).
- All entries in any column must be of the same kind. For example, in the column labeled "Customer," only customer names or numbers are permitted.

Requirements:

- There should be a primary key, and
- No column should contain more than one value (no multi-valued attributes)

In order to perform first normalizationrule, we have to consider the following concepts

- Eliminate duplicative columns from the same table.
- Create separate tables for each group of related data and identify each row with a unique column or set of columns (theprimary key).
- Eliminate composite attributes

Example 1: Develop 1NF of the following table

Student Details	**Course Details**	**Result details**
1001 Ram 11/09/1986	M4 Basic Math's 7	11/11/2004 89 A
1002 Shyam 12/08/1987	M4 Basic Math's 7	11/11/2004 78 B
1001 Ram 23/06/1987	H6 4	11/11/2004 87 A
1003 Sita 16/07/1985	C3 Basic Chemistry 11	11/11/2004 90 A
1004 Gita 24/09/1988	B3 8	11/11/2004 78 B

1002 Shyam 23/06/1988	P3 Basic Physics 13	11/11/2004 67 C
1005 Sunita 14/09/1987	P3 Basic Physics 13	11/11/2004 78 B
1003 Sita 23/10/1987	B4 5	11/11/2004 67 C
1005 Sunita 13/03/1990	H6 4	11/11/2004 56 D
1004 Gita 21/08/1987	M4 Basic Math’s 7	11/11/2004 78 B

The above table Student Details, Course Details and Result Details can be further divided.

- **Student Details** attribute is divided into Student#(Student Number), Student Name and date of birth.
- **Course Details** is divided into Course#, Course Name and duration.
- **Results** attribute is divided into Date ofexam, Marks and Grade.

II. Second normal form (2NF): **Eliminating Redundant Data**

- Second normal form (2NF) requires that all non-key columns are fully dependent on the entire primary key. If the table has only a single-column primary key, this requirement is easily met.

- At this level of normalization, each column in a table that is not a determiner of the contents of another column must itself be a function of the other columns in the table. For example, in a table with three columns containing customer ID, product sold, and price of the product when sold, the price would be a function of the customer ID (entitled to a discount) and the specific product.

In order to perform first normalizationrule, we have to consider the following concepts

- Meet all the requirements of the first normal form and remove subsets of data that apply to multiple rows of a table and place them in separate tables.
- Create relationships between these new tables and their predecessors through the use of foreign keys.

Example 2

Let us re-visit 1NF table structure.

- Student# is key attribute for Student,
- Course# is key attribute for Course
- Student# and Course# together form the composite key attributes for result relationship.

- Other attributes are non - key attributes.

To make this table 2NF compliant, we have to remove all the partial dependencies.

- Student Name and Date ofBirth depends only on student#.
- CourseName, Pre-Requisite and DurationInDays depends only on Course#
- Date ofExam depends only on Course#.

To remove this partial dependency, we need to split Student_Course_Result table into four separate tables, STUDENT, COURSE, RESULT and EXAM_DATE tables as shown in the following:

STUDENT TABLE

Student #	**Student Name**	**DateofBirth**
1001	Ram	Some value
1002	Shyam	Some value
1003	Sita	Some value
1004	Geeta	Some value
1005	Sunita	Some value

COURSE TABLE

Course#	**CourseName**	**Duration of days**
C3	Bio Chemistry	3
B3	Botany	8
P3	Nuclear Physics	1
M4	Applied Mathematics	4
H6	American History	5
B4	Zoology	9

RESULT TABLE

Student#	**Course#**	**Marks**	**Grade**
1001	M4	89	A

1002	M4	78	B
1001	H6	87	A
1003	C3	90	A
1004	B3	78	B
1002	P3	67	C
1005	P3	78	B
1003	B4	67	C
1005	H6	56	D
1004	M4	78	B

EXAM DATE Table

Course#	**DateOfExam**
M4	Some value
H6	Some value
C3	Some value
B3	Some value
P3	Some value
B4	Some value

- In STUDENT table, the key attribute is Student# and all other non-key attributes, Student name and Date ofBirth are fully functionally dependent on the key attribute.
- In COURSE table, the key attribute is Course# and all the non-key attributes, Course name, Duration in days are fully functional dependent on the key attribute.
- In RESULT table, the key attributes are #StudentCourse# together and all other non-key attributes, Marks and Grade are fully functional dependent on the key attributes.
- In EXAM DATE table, the key attribute is Course# and the non key attribute Date ofExam is fully functionally dependent on the key attribute.

At first look it appears like all our anomalies are taken away! Now we are storing Student 1003 and M4 record only once. We can insert prospective students and courses at our will.

We will update only once if we need to change any data in **STUDENT**, **COURSE** tables. We can get rid of any course or student details by deleting just one row.

Let us analyze the RESULT Table

We already concluded that:

- All attributes are atomic in nature
- No partial dependency exists between the key attributes and non-key attributes
- RESULT table is in 2NF

Assume, at present, as per the university evaluation policy,

- Students who score more than or equal to 80 marks are awarded with “A” grade
- Students who score more than or equal to 70 marks up till 79 are awarded with “B” grade
- Students who score more than or equal to 60 marks up till 69 are awarded with “C” grade
- Students who score more than or equal to 50 marks up till 59 are awarded with “D” grade

The University management which is committed to improve the quality of education wants to change the existing grading system to a new grading system. In the present RESULT table structure,

- We don’t have an option to introduce new grades like A+, B- and E
- We need to do multiple updates on the existing record to bring them to new grading definition
- We will not be able to take away “D” grade if we want to.
- 2NF does not take care of all the anomalies and inconsistencies.

III. Third normal form (3NF): Eliminating Columns Not Dependent on Keys

Requires that there are no transitive dependencies, where one column depends on another column which depends on the primary key.At the 2NF, modifications are still possible because a change to one row in a table may affect data that refers to this information from another table. For example, using the customer table just cited, removing a row describing a customer purchase (because of a return perhaps) will also remove the fact that the product has a certain price. In the third normal form, these tables would be divided into two tables so that product pricing would be tracked separately.

In order to perform first normalizationrule, we have to consider the following concepts

- Meet all the requirements of the second normal form.
- Remove columns that are not dependent upon the primary key.
- No transitive dependency exists between non-key attributes and key attributes.

Example 3: In the above RESULT table Student# and Course# are the key attributes. All other attributes, except grade are non-partially, non-transitively dependent on key attributes. The grade attribute is dependent on "Marks ", and in turn "Marks" is dependent on #Student#Course. To bring the table in 3NF, we need to take off this transitive dependency.

Student#	**Course#**	**Marks**
1001	M4	89
1002	M4	78
1001	H6	87
1003	C3	90
1004	B3	78
1002	P3	67
1005	P3	78
1003	B4	67
1005	H6	56
1004	M4	78

UpperBound	**LowerBound**	**Grade**
100	95	A+
94	90	A
89	85	B+
84	80	B
79	75	B-
74	70	C
69	65	C-

After normalizing tables to 3NF, we got rid of all the anomalies and inconsistencies. Now we can add new grade systems, update the existing one and delete the unwanted ones. Hence the Third Normal form is the most optimal normal form and 99% of the databases which require efficiency in

- INSERT, UPDATE and DELETE Operations are designed in this normal form

IV. Boyce-Codd Normal Form (BCNF or 3.5NF)

- The Boyce-Codd Normal form, also referred to as the "**third and half (3.5) normal form**", adds one more requirement:
  - ➢ Meet all the requirements of the third normal form.
  - ➢ Every determinant must be a candidate key.
- Boyce Codd Normal Form (BCNF) is a further refinement of 3NF. A row is in Boyce Codd normal form if and only if every determinant is a candidate key.
- Most entities in 3NF are already in BCNF.

V. Fourth Normal Form (4NF)

- Fourth normal form (4NF) has one additional requirement:
  - ➢ Meet all the requirements of the third normal form.
  - ➢ A relation is in 4NF if it has no multi-valued dependencies.
- An entity is in Fourth Normal Form (4NF) if and only if it is in 3NF and has no multiple sets of multi-valued dependencies. In other words, 4NF states that no entity can have more than a single one-to-many relationship within an entity if the one-to-many attributes are independent of each other.

VI. Fifth Normal Form (5NF)

- 5NF specifies that every join dependency for the entity must be a consequence of its candidate keys.

Summary of normal forms

- A row is in first normal form if and only if all underlying domains contain atomic values only. 1NF eliminates repeating groups by putting each into a separate table and connecting them with a one-to-many relationship.
- A row is in second normal form if and only if it is in first normal form and every non-key attribute is fully dependent on the key. 2NF eliminates functional dependencies on a partial key by putting the fields in a separate table from those that are dependent on the whole key. A row is in third normal form if and only if it is in second normal form and every non-key attribute is non-transitively dependent on the primary key. 3NF eliminates

functional dependencies on non-key fields by putting them in a separate table.non-key fields are dependent on the key.

- **Functional Dependency**

Functional dependency is a relationship between two sets of attributes in a database table. It describes the dependency of one attribute (or a set of attributes) on another attribute (or a set of attributes). In other words, if changing the value of one attribute determines the value of another attribute(s), then a functional dependency exists.

Example: In a table called "Employees," if the attribute "EmployeeID" determines the attribute "EmployeeName," it can be represented as: EmployeeID -> EmployeeName. This means that for every value of EmployeeID, there is a unique value of EmployeeName associated with it.

- **Full Functional Dependency**

Full functional dependency occurs when an attribute is functionally dependent on the entire primary key of a table, and not just a part of it. In other words, all non-key attributes depend on the entire primary key and not on any subset of it.

Example: In a table called "Students," if the primary key is composed of "StudentID" and "CourseID," and the attribute "Grade" depends on both of these attributes, it can be represented as: StudentID, CourseID -> Grade. This means that for every combination of StudentID and CourseID, there is a unique value of Grade associated with it.

- **Partial Dependency**

Partial dependency occurs when an attribute is functionally dependent on only a part of the primary key, rather than the entire primary key. It means that a non-key attribute depends on only a subset of the primary key, and not on the entire primary key.

Example: In a table called "Orders," if the primary key is "OrderID," and the attributes "CustomerName" and "CustomerAddress" depend on only the attribute "OrderID," it indicates a partial dependency. This can be represented as: Ordered ->CustomerName, CustomerAddress. To remove the partial dependency, the table can be split into two separate tables: "Orders" and "Customers," where the customer details are stored separately.

- **Transitive Dependency**

Transitive dependency occurs when an attribute depends on another non-key attribute, rather than directly depending on the primary key. It means that the dependency is indirectly established through another attribute.

Example: In a table called "Employees," if the primary key is "EmployeeID," and the attributes "Department" and "DepartmentLocation" depend on each other, it indicates a transitive dependency. This can be represented as: EmployeeID -> Department -> DepartmentLocation. To remove the transitive dependency, the table can be split into two separate tables: "Employees" and "Departments," where the department details are stored separately.

Identifying and eliminating partial and transitive dependencies are crucial in achieving higher levels of normalization (such as 3NF or BCNF) to ensure data integrity, reduce redundancy, and avoid anomalies in a database.

2.3. Normalize business data and document results

Undertaking normalization of business data involves the process of organizing and structuring the data in a database to eliminate redundancy, improve data integrity, and reduce anomalies. The goal is to ensure that each piece of data is stored in the most efficient and logical manner.

The steps involved in normalization typically include:

1. Analyzing the data: The first step is to analyze the existing data in the database. This involves identifying the various entities, attributes, and relationships between them.
2. Applying normalization rules: Next, the data is normalized by applying normalization rules, specifically the rules outlined in normal forms, such as First Normal Form (1NF), Second Normal Form (2NF), and so on. Each normalization form has specific criteria that need to be met.
3. Breaking down tables: In order to meet the criteria for normalization forms, it may be necessary to break down existing tables into multiple tables, with each table focusing on a specific entity or relationship.

4. Resolving dependencies: During the normalization process, dependencies between attributes are identified and resolved. This includes identifying and eliminating partial dependencies and transitive dependencies.
5. Documenting the results: Once the normalization process is complete, it is important to document the results. This documentation includes the structure of the normalized tables, the relationships between them, and any changes made to the original data model.

By undertaking normalization of business data and documenting the results, organizations can ensure that their databases are efficiently structured, leading to improved data quality, easier data maintenance, and more effective data operations.

2.4. Compare normalization results with ERdiagram

When comparing normalization results with an entity relationship diagram (ERD), it is important to understand the relationship between the two and how they complement each other in the process of designing a database.

An entity relationship diagram is a visual representation of the entities (tables), attributes, and relationships between them in a database. It helps in understanding the structure and organization of the data, as well as the dependencies between different entities. An ERD typically includes entity boxes, attribute labels, and lines representing the relationships between entities.

Normalization, on the other hand, is a set of rules and guidelines used to eliminate redundancy, improve data integrity, and reduce anomalies in a database. It involves breaking down tables, resolving dependencies, and organizing data into well-structured tables.

When comparing normalization results with an ERD, the main focus is on ensuring that the normalization process aligns with the relationships and dependencies depicted in the ERD.

Suppose we have an ER diagram representing a library database. The diagram includes entities such as "Books," "Authors," and "Publishers," with relationships like "Author writes Book" and "Publisher publishes Book."

Now, let's say we apply normalization to this database. During the normalization process, we break down the initial table into smaller, more atomic tables to eliminate redundancy and improve data integrity.

For instance, we might have initially had a single table with columns like "Book ID," "Book Title," "Author Name," and "Publisher Name." After normalization, we would have separate tables for "Books," "Authors," and "Publishers," each with their own unique identifiers and relevant attributes.

When comparing the normalization results with the ER diagram, we would examine if the relationships depicted in the ER diagram are accurately represented in the normalized tables. We would ensure that the foreign keys in the normalized tables correctly establish the relationships between entities.

In our example, we would check if the "Author ID" in the "Books" table references the corresponding "Author ID" in the "Authors" table, and if the "Publisher ID" in the "Books" table references the appropriate "Publisher ID" in the "Publishers" table. This comparison ensures that the normalization process has preserved the intended relationships between entities.

Additionally, we would verify if the attributes in the normalized tables align with the attributes specified in the ER diagram. We would ensure that no redundant data exists and that the data is properly organized according to normalization rules.

By comparing the normalization results with the ER diagram, we can validate the accuracy and consistency of the database design, ensuring that the normalized tables effectively capture the structure and relationships depicted in the ER diagram.

2.5. Reconcile differences between data

Reconciling differences between data in normalization refers to the process of resolving conflicts or inconsistencies that may arise during the normalization process. These conflicts can occur when attempting to organize and structure data into normalized tables, especially when there are dependencies or relationships between attributes and entities.

Here are some key points to understand about reconciling differences between data in normalization

1. Identify Inconsistencies: The first step is to identify any inconsistencies or conflicts within the data. This may involve analyzing the relationships, dependencies, and functional dependencies between attributes and entities.
2. Analyze Dependencies: Evaluate the dependencies between attributes and entities to determine if they are accurately represented. This includes identifying partial dependencies (where an attribute depends on only a part of the primary key) or transitive dependencies (where an attribute depends on another non-key attribute).

3. Normalize Data: Apply normalization rules, such as First Normal Form (1NF), Second Normal Form (2NF), and so on, to organize the data into well-structured tables. This may involve breaking down tables, creating separate tables for related entities, and defining appropriate primary and foreign keys.
4. Resolve Conflicts: Address any conflicts or inconsistencies that arise during the normalization process. This may involve making decisions on how to handle partial dependencies or transitive dependencies. One approach is to split tables and create additional tables to ensure that data is properly organized and dependencies are accurately represented.
5. Ensure Data Integrity: As you reconcile differences, it is crucial to maintain data integrity. This means that the data in the normalized tables should accurately represent the relationships and dependencies between entities. It also involves ensuring that there are no duplicate or redundant data.
6. Validate Results: Validate the results of the normalization process to ensure that the reconciled data aligns with the intended structure and relationships. This can be done by comparing the normalized tables with the original data, verifying that the relationships and dependencies are accurately represented.

Reconciling differences between data in normalization is an essential step in achieving a well-structured and efficient database design. It helps eliminate redundancy, reduce anomalies, and improve data integrity.

Self-Check 2

Part I: choose the best answer

1. In which normal form a table cannot hold multiple values
   - A. 3NF
   - B. 2NF
   - C. 1NF
   - D. BNCF
2. What is the purpose of reconciling differences between data in normalization?
   - a. To introduce redundancy in the data
   - b. To reduce data integrity
   - c. To maximize conflicts and inconsistencies
   - d. To ignore relationships between entities
3. Choose one of the following which is not example of anomaly.
   - a. Insert anomalies
   - b. delete anomalies
   - c. update anomalies
   - d. Entity anomalies
4. ________ is normalization concept that describes how one/more field/s determines another field.
   - a. ERD
   - b. Relationship
   - c. functional dependency
   - d. database

Part II: Matching

___1. Normalization

___**2.** 1NF

___**3.** 2NF

___**4.** 3NF

___**5.** 3.5NF

___6. 4NF

___7. 5NF

___8. Completeness

___9. Accuracy

___10. Generalization & Specialization

A. Normal form that eliminates partial dependency

B. BCNF (Boyce-Code normal form)

C. Normal form that eliminates multi-valued dependency

D. Normal form that eliminates joint dependency

E. The "basic" level of normalization

F. Normal form that eliminates transitive dependency

G. Helps eliminate data redundancy

H. The model follows the data modeling rules

I. The model correctly represents entities, attributes and relationships

J. The model contains all of the data needed

k. Functional dependency

L. Database

Part III: Answer the following questions briefly

1. Why is it important to understand and apply normalization rules to business data?
2. What are the main benefits of normalization in relation to a model data object?
3. Write some of the steps in determining the suitability of business data for a model data object?
4. What is the significance of comparing normalization results with an entity relationship (ER) diagram?

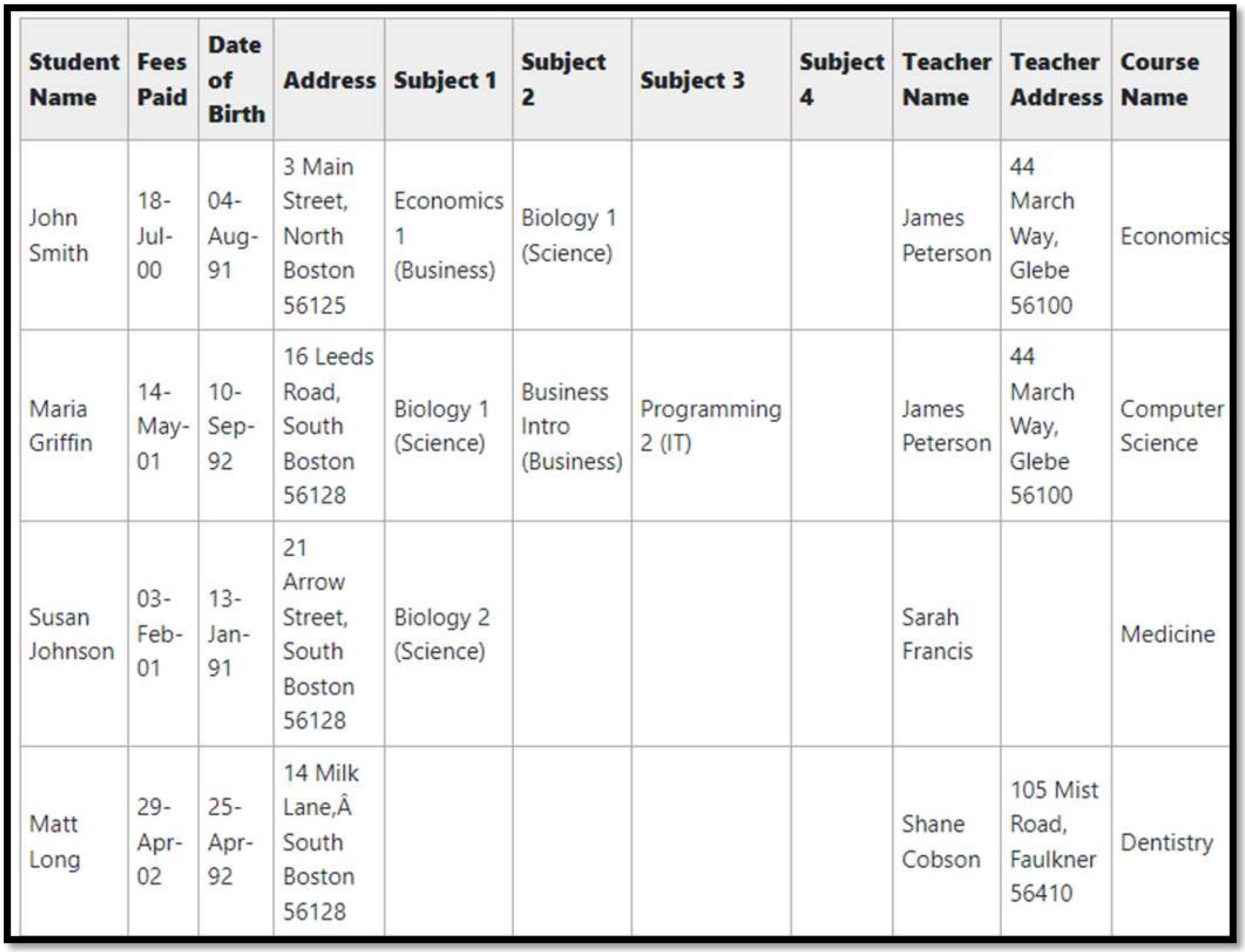
Operation Sheet-2.1Normalization

Operation title: Normalization

Purpose: To normalize the student database table

Step 1: Based on the below student table, we will start with getting the data to itsfirst normal Form.

Student table



Student Name	Fees Paid	Date of Birth	Address	Subject 1	Subject 2	Subject 3	Subject 4	Teacher Name	Teacher Address	Course Name
John Smith	18-Jul-00	04-Aug-91	3 Main Street, North Boston 56125	Economics 1 (Business)	Biology 1 (Science)			James Peterson	44 March Way, Glebe 56100	Economics
Maria Griffin	14-May-01	10-Sep-92	16 Leeds Road, South Boston 56128	Biology 1 (Science)	Business Intro (Business)	Programming 2 (IT)		James Peterson	44 March Way, Glebe 56100	Computer Science
Susan Johnson	03-Feb-01	13-Jan-91	21 Arrow Street, South Boston 56128	Biology 2 (Science)				Sarah Francis		Medicine
Matt Long	29-Apr-02	25-Apr-92	14 Milk Lane,Â South Boston 56128					Shane Cobson	105 Mist Road, Faulkner 56410	Dentistry

In apply the first normal form, There is no unique field, we need to create a new field.

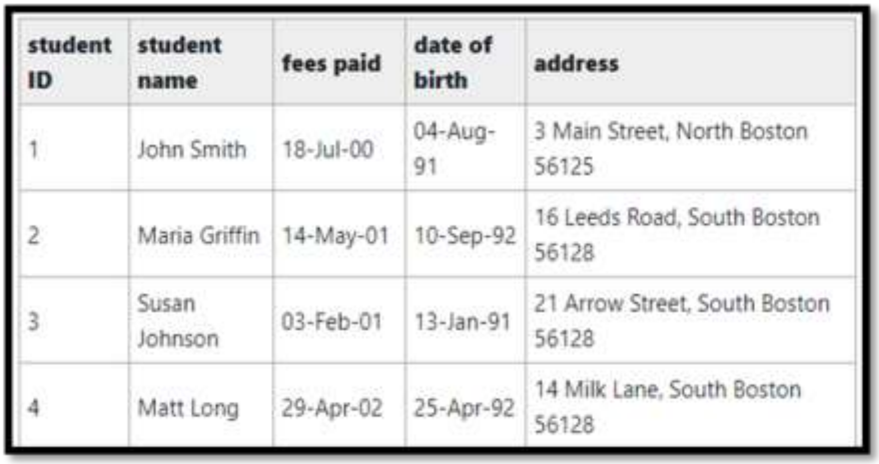
This is our new table in its first normal form with the attributes

Student (<u>student ID</u>, student name, fees paid, date of birth, address, subject 1, subject 2, subject 3, subject 4, teacher name, teacher address, course name)

This data is now in first normal form and in one table, but it's been made a little better by adding a unique value to it.

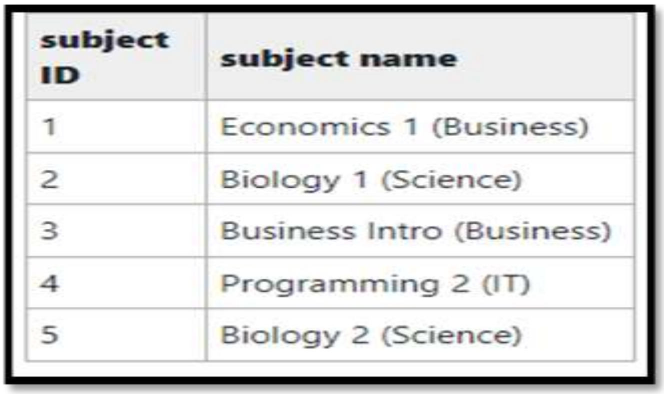
Step 2: The student ID (primary key), represents the student but not subject, teacher and course. To resolve this issue, we have to create separate table for subject, teacher and course by creating a primary key column, just like we did for student table.

Student Table



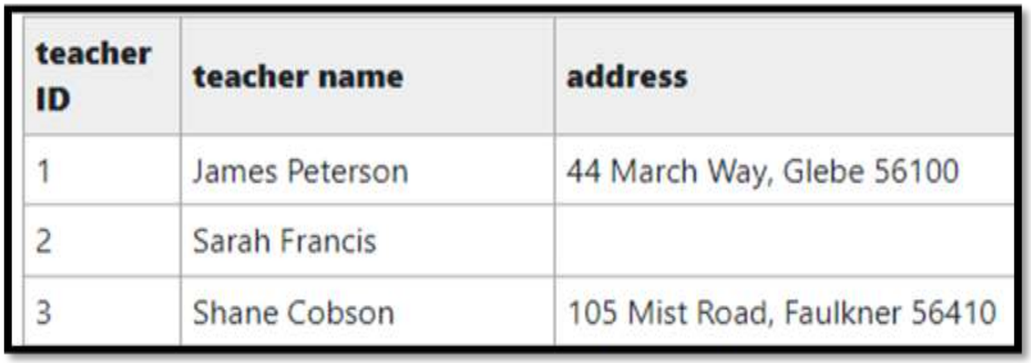
student ID	student name	fees paid	date of birth	address
1	John Smith	18-Jul-00	04-Aug-91	3 Main Street, North Boston 56125
2	Maria Griffin	14-May-01	10-Sep-92	16 Leeds Road, South Boston 56128
3	Susan Johnson	03-Feb-01	13-Jan-91	21 Arrow Street, South Boston 56128
4	Matt Long	29-Apr-02	25-Apr-92	14 Milk Lane, South Boston 56128

Subject Table



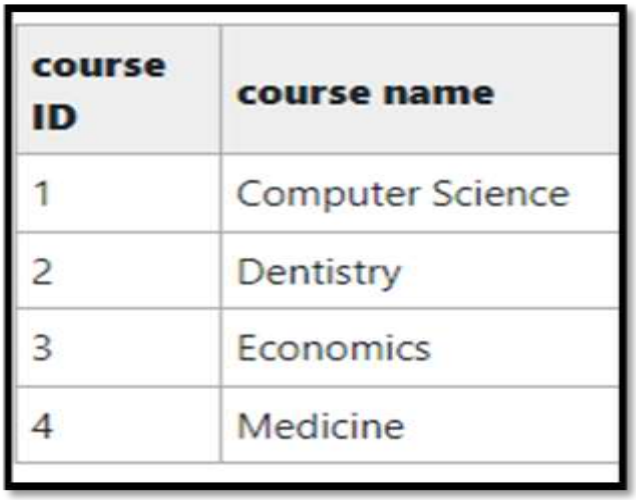
subject ID	subject name
1	Economics 1 (Business)
2	Biology 1 (Science)
3	Business Intro (Business)
4	Programming 2 (IT)
5	Biology 2 (Science)

Teacher Table



teacher ID	teacher name	address
1	James Peterson	44 March Way, Glebe 56100
2	Sarah Francis	
3	Shane Cobson	105 Mist Road, Faulkner 56410

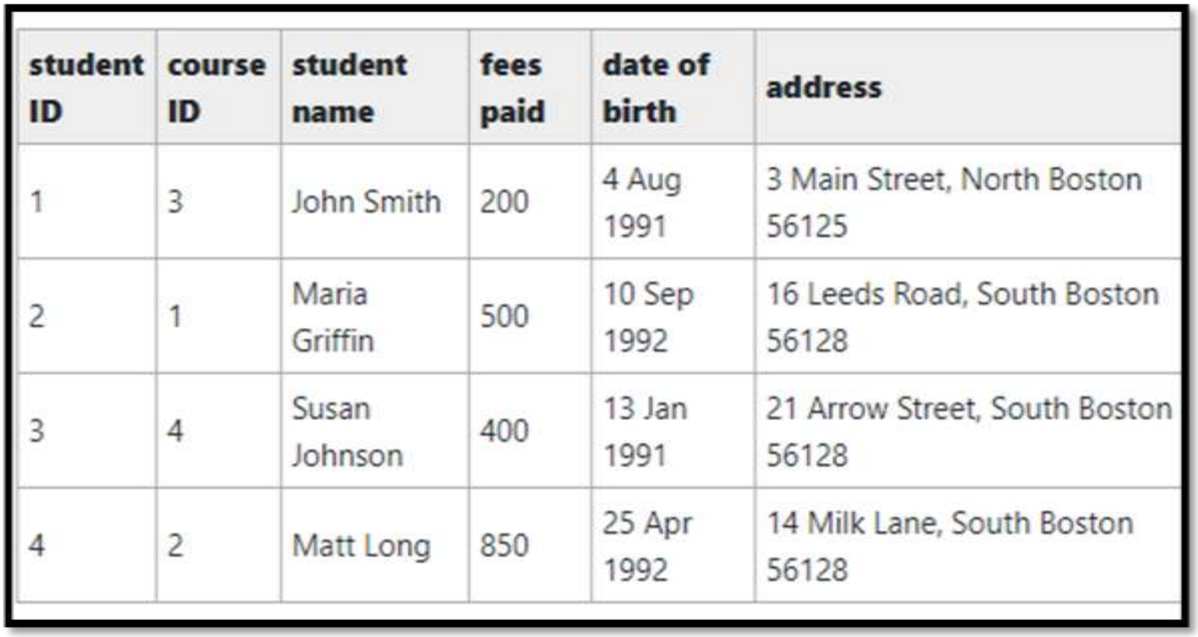
Course Table



course ID	course name
1	Computer Science
2	Dentistry
3	Economics
4	Medicine

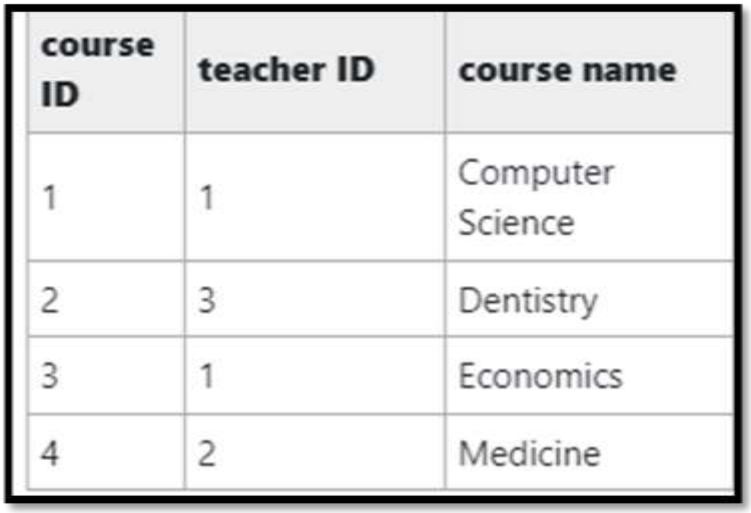
We have four separate tables, capturing different pieces of information. We need to capture students are taking certain courses, have teachers, and subjects. But the data is in different tables. So, we use the concept foreign key.

To link the student andcoursetables using a foreign key, we need to put the primary key (the underlined column) from one table into the other table.



student ID	course ID	student name	fees paid	date of birth	address
1	3	John Smith	200	4 Aug 1991	3 Main Street, North Boston 56125
2	1	Maria Griffin	500	10 Sep 1992	16 Leeds Road, South Boston 56128
3	4	Susan Johnson	400	13 Jan 1991	21 Arrow Street, South Boston 56128
4	2	Matt Long	850	25 Apr 1992	14 Milk Lane, South Boston 56128

To link the courseand teacher tables using a foreign key, we need to put the primary key (the underlined column) from one table into the other table.



course ID	teacher ID	course name
1	1	Computer Science
2	3	Dentistry
3	1	Economics
4	2	Medicine

So, we've linked the course, teacher, and student tables together so far. What about the subject table?

A student can be enrolled in many subjects at a time, and a subject can have many students in it. This relationship is many to many. We can’t represent this relationship by putting a foreign key in each table, so how can we represent it?We can represent it by using a joining table.

Subject_Enrollment (student ID, subject ID)

Step 3: In applying the third normal form, we will analyze the following concepts, this student table,

Student (student ID, course ID, student name, fees paid, date of birth, address)

we see address it might depend on zip codeSo,We can move the ZIP code to another table, along with everything it identifies, and link to it from the student table.

Student Table*: Student (student ID, course ID, student name, fees paid, date of birth, street address, address code ID)*

Address Table: *Address Code (address code ID, ZIP code, suburb, city, state)*

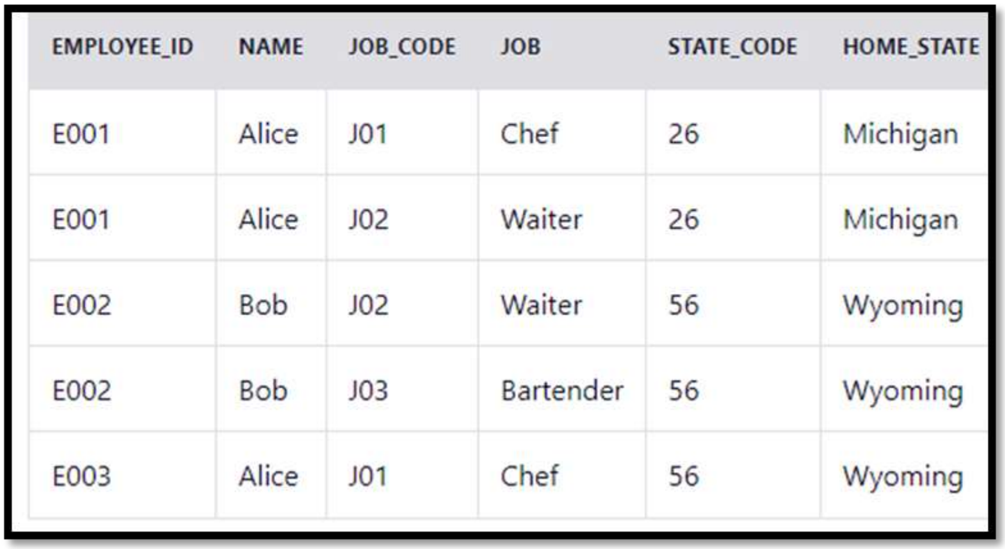
So our table would look like this:Teacher Table: *Teacher (teacher ID, teacher name, street address, address code ID)*

Address Table: *Address Code (address code ID, ZIP code, suburb, city, state)*

Lap Test

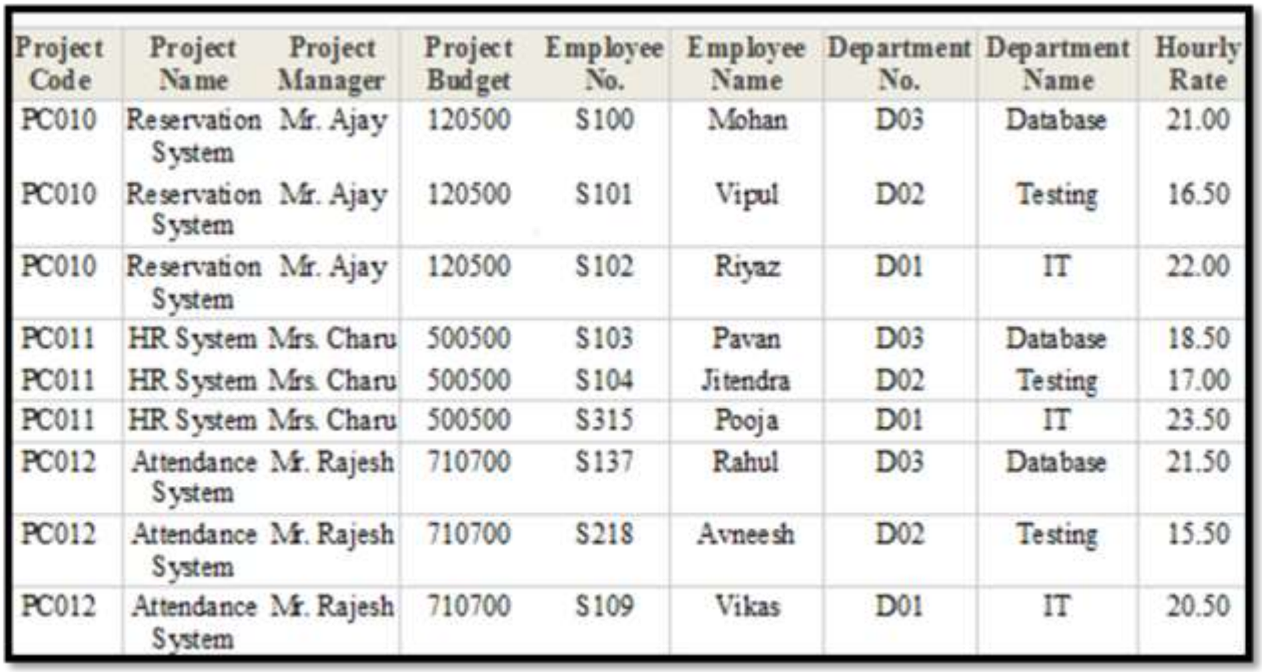
Instruction: Apply normalization rule for the given two tables

Task 1: Restaurant management table



EMPLOYEE_ID	NAME	JOB_CODE	JOB	STATE_CODE	HOME_STATE
E001	Alice	J01	Chef	26	Michigan
E001	Alice	J02	Waiter	26	Michigan
E002	Bob	J02	Waiter	56	Wyoming
E002	Bob	J03	Bartender	56	Wyoming
E003	Alice	J01	Chef	56	Wyoming

Task 2: Project table



Project Code	Project Name	Project Manager	Project Budget	Employee No.	Employee Name	Department No.	Department Name	Hourly Rate
PC010	Reservation System	Mr. Ajay	120500	S100	Mohan	D03	Database	21.00
PC010	Reservation System	Mr. Ajay	120500	S101	Vipul	D02	Testing	16.50
PC010	Reservation System	Mr. Ajay	120500	S102	Riyaz	D01	IT	22.00
PC011	HR System	Mrs. Charu	500500	S103	Pavan	D03	Database	18.50
PC011	HR System	Mrs. Charu	500500	S104	Jitendra	D02	Testing	17.00
PC011	HR System	Mrs. Charu	500500	S315	Pooja	D01	IT	23.50
PC012	Attendance System	Mr. Rajesh	710700	S137	Rahul	D03	Database	21.50
PC012	Attendance System	Mr. Rajesh	710700	S218	Avneesh	D02	Testing	15.50
PC012	Attendance System	Mr. Rajesh	710700	S109	Vikas	D01	IT	20.50

Unit Three: Data model validation

This unit is developed to provide you the necessary information regarding the following content coverage and topics

- Validation of data model with client.
- Resolve arising issues or recommendations
- Documentation ofdata model
- Client Approval Submission

This unit will also assist you to attain the learning outcomes stated in the cover page. Specifically, upon completion of this learning guide, you will be able to:

- Validate data model with client
- Understand problem resolution and recommendation skills
- Document completed data model
- Have effective client approval submission skills

3.1. Validation of data model with client

Model validation essential parts of the model development process if models to be accepted and used to support decision making. One of the very first questions that a person who is promoting a model is likely to encounter is "has your model been validated?"

Does the model represent and correctly reproduce the behaviors of the real-world system?

- Validation ensures that the model meets its intended requirements in terms of the methods employed and the results obtained
- The ultimate goal of model validation is to make the model useful in the sense that the model addresses the right problem, provides accurate information about the system being modeled, and to makes the model actually used.

Validating the data model with the client in web development and database administration refers to the process of confirming the accuracy, completeness, and appropriateness of the data model design in collaboration with the client or stakeholders.

Here are a few key points to understand about validating the data model with the client:

- Collaboration: The validation process involves close collaboration with the client or stakeholders. It is essential to actively involve them throughout the data modeling process, from the initial requirements gathering to the final validation stage.
- Requirements Alignment: The data model should align with the client's requirements and objectives. It is crucial to ensure that the data model accurately represents the desired functionality, relationships, and constraints of the web application or database system.
- Review and Feedback: Present the data model to the client for review and feedback. This can be done through meetings, presentations, or documentation. The client should have the opportunity to evaluate the data model and provide input, suggestions, or corrections based on their understanding and needs.
- Accuracy and Completeness: Validate that the data model accurately represents the client's requirements and that no essential elements have been overlooked. This includes verifying that all necessary entities, attributes, relationships, and constraints have been properly captured in the data model.
- Consistency and Usability: Ensure that the data model is consistent, organized, and easy to understand. It should follow established conventions and best practices in database design.

The client should be able to navigate and comprehend the data model easily, facilitating effective communication and decision-making.

- Iterative Process: Validating the data model is typically an iterative process. It may involve multiple rounds of review and revision based on client feedback. Each iteration aims to refine and improve the data model until it meets the client's requirements and expectations.
- Documentation: Document any changes, modifications, or clarifications made during the validation process. This helps maintain a record of the decisions made and ensures that all stakeholders are on the same page regarding the finalized data model.

Validating the data model with the client is crucial to ensure that the resulting web application or database system meets the client's needs and expectations. It helps identify any gaps or inconsistencies early on and allows for necessary adjustments to be made before implementation. By actively involving the client in the validation process, you can enhance transparency, collaboration, and ultimately deliver a robust and effective solution.

3.2. Resolve arising issues or recommendations

Resolving arising issues or making involves addressing challenges or suggesting improvements to ensure the smooth functioning and optimal performance of web applications and database systems. Here are some key steps to follow:

- Identify the Issue: The first step is to identify and understand the specific issue or challenge at hand. This could be related to performance, security, scalability, user experience, or any other aspect of web development or database administration.
- Gather Information: Collect relevant data and information related to the issue. This may include analyzing system logs, monitoring metrics, user feedback, or conducting performance tests. The more information you have, the better you can diagnose the problem or make informed recommendations.
- Analyze the Root Cause: Conduct a thorough analysis to determine the underlying cause of the issue. This may involve examining the codebase, database design, network configuration, or any other relevant factors. The goal is to identify the root cause so that appropriate measures can be taken.

- Propose Solutions: Based on the analysis, propose potential solutions to address the issue. This could involve making changes to the code, optimizing database queries, adjusting system configurations, enhancing security measures, or implementing new technologies.
- Evaluate Trade-offs: Assess the potential impact and trade-offs of each proposed solution. Consider factors such as development effort, cost, time, compatibility, and potential risks. This evaluation helps prioritize the solutions and select the most suitable approach.
- Implement and Test: Once a solution is chosen, implement the necessary changes or improvements. Ensure thorough testing is conducted to verify that the issue has been resolved and that the system is functioning as expected. This may involve unit testing, integration testing, load testing, or any other relevant testing methods.
- Monitor and Iterate: Continuously monitor the system after implementing the solution. Keep an eye on performance metrics, user feedback, and any new issues that may arise. If necessary, iterate on the solution or make further adjustments to optimize the system's performance and address any new challenges.
- Document and Communicate: Document the issue, solution, and any changes made for future reference. Communicate the findings and recommendations to relevant stakeholders, such as developers, database administrators, project managers, or clients. This ensures that everyone is aware of the actions taken and the impact on the system.

Resolving issues and making recommendations in web development and database administration is an ongoing process. It requires a combination of technical expertise, analytical skills, and effective communication to identify and address challenges effectively, leading to improved performance, security, and user satisfaction.

3.3. Documentation of completed data model

Documenting the completed data model is crucial for maintaining a clear and comprehensive record of the data model's structure, relationships, and constraints. This documentation serves as a valuable reference for developers, database administrators, and other stakeholders involved in the project. Here are some key components to include when documenting a completed data model:

- Entity Relationship Diagram (ERD): Start by including an ERD that visually represents the entities, attributes, and relationships in the data model. This diagram provides a clear overview of the data model's structure and helps stakeholders understand the relationships between entities.

- Entity Descriptions: Provide detailed descriptions of each entity in the data model. Include information such as the entity's purpose, key attributes, and any constraints or business rules associated with it. This description helps stakeholders understand the role and significance of each entity in the system.
- Attribute Definitions: Document the attributes present in each entity, including their names, data types, lengths, and any applicable constraints (such as primary keys, foreign keys, or unique constraints). This information helps ensure consistency and accuracy when working with the data model.
- Relationship Descriptions: Describe the relationships between entities, including their types (such as one-to-one, one-to-many, or many-to-many) and any constraints or rules that apply to them. Clearly explain how the entities are connected and the implications of these relationships in the system.
- Data Constraints: Document any additional constraints or rules that apply to the data model, such as domain constraints, referential integrity rules, or check constraints. These constraints ensure the data remains valid and consistent throughout the system.
- Indexes and Performance Considerations: If applicable, document any indexes or performance considerations that have been implemented in the data model. This information helps developers and administrators optimize query performance and improve overall system efficiency.
- Data Dictionary: Create a data dictionary that provides a centralized reference for all the entities, attributes, relationships, and constraints in the data model. This dictionary should include clear definitions and explanations of each element, serving as a comprehensive guide for anyone working with the data model.
- Version Control and Change History: Maintain a version control system to track changes made to the data model over time. Include a change history log that documents the modifications, reasons for changes, and the individuals responsible for making the updates. This ensures transparency and helps with future troubleshooting or system audits.

- Diagram Notations and Conventions: Include a section that explains the notations and conventions used in the data model documentation. This helps ensure consistency and understanding among team members who may be working on different aspects of the project.
- Glossary of Terms: Create a glossary of terms used in the data model documentation to clarify any technical or domain-specific terminology. This can be particularly helpful for stakeholders who may be less familiar with the technical aspects of the data model.

By documenting the completed data model thoroughly, you provide a valuable resource for understanding and working with the system. The documentation helps facilitate effective communication, collaboration, and maintenance of the data model throughout the web development and database administration lifecycle.

3.4. Client approval submission

Submitting the modeling data object to the client for final approval involves presenting the data model to the client and seeking their confirmation and acceptance. Here are some steps to follow when submitting the modeling data object to the client for final approval:

- Prepare the Presentation: Create a clear and concise presentation of the modeling data object. This may include visual representations such as entity-relationship diagrams, data flow diagrams, or any other relevant diagrams to help the client understand the structure and relationships of the data model.
- Explain the Purpose and Benefits: Start the presentation by explaining the purpose of the data model and how it aligns with the client's requirements and objectives. Highlight the benefits and advantages of the data model, such as improved data organization, increased efficiency, or enhanced reporting capabilities.
- Walkthrough the Data Model: Take the client through a detailed walkthrough of the data model. Explain the entities, attributes, relationships, and any constraints or rules that have been incorporated. Focus on ensuring that the client understands the logical representation and the intended functionality of the data model.
- Address Questions and Concerns: During the presentation, encourage the client to ask questions or raise any concerns they may have. Take the time to provide clarifications and address any uncertainties. This ensures that the client has a clear understanding of the data model and can make an informed decision.

- Discuss Potential Revisions or Modifications: If the client raises any concerns or suggests changes to the data model, engage in a constructive discussion. Consider their feedback and evaluate the feasibility and impact of the proposed revisions. Collaborate with the client to find the most appropriate solutions.
- Document Client Feedback: Document any feedback or changes requested by the client during the presentation. This serves as a reference for future discussions and helps ensure that the final data model accurately reflects the client's requirements.
- Seek Approval: Once the presentation and discussion are complete, formally request the client's approval of the modeling data object. Clearly communicate the next steps and the timeline for implementation.
- Address Final Revisions: If the client requests any final revisions or modification, incorporate them into the data model as agreed upon. Document these changes and ensure that they align with the client's expectations.
- Obtain Signed Approval: Once all revisions have been made and the data model is finalized, request the clients to provide their formal approval in writing. This can be in the form of a signed document or an email confirming their acceptance of the data model.
- Maintain Documentation: Keep a copy of the approved data model, along with any related documentation, for future reference. This ensures that there is a record of the agreed-upon data model and provides a basis for any future updates or enhancements.

Submitting the modeling data object to the client for final approval is a critical step in the data modeling process. It ensures that the client is satisfied with the proposed solution and provides a basis for moving forward with implementation. Effective communication, collaboration, and documentation are the key to successfully obtaining client approval and ensuring the data model meets their requirements.

Self-Check 3

Part I: Choose the best answer

1. What is the purpose of submitting a website and database to the client for final approval?

   a. To ensure that the developed website and database meet the client's requirements and expectations.

   b. To obtain the client's sign-off and acknowledgement that the project is complete.

   c. To allow the client to review, test, and provide feedback on the functionality and design of the website and database.

   d. All of the above.

2. Why is it important to document the completed data model when developing a website and database?

   a. To provide a reference for future developers or team members who may need to work on the project.

   b. To ensure clear understanding of the structure and relationships within the database.

   c. To maintain a clear and comprehensive record of the data model's structure.

   d. All of the above.

3. When validating a data model with the client, who is responsible for providing feedback and suggestions?

4. a) Project manager c) System administrator

   b) Data analyst d) Client/stakeholder

Part-II: Answer the following questions accordingly

1. How does client involvement in the validation process contribute to the success of the data model?
2. How should identified issues or recommendations be addressed and resolved?
3. Why is it important to document the completed data model?
4. What is the purpose of client approval submission in the context of a data model?

Part-III: Say true or false

1. Validation of the data model is not necessary once it has been developed.
2. Client feedback is not important during the validation of a data model.
3. Prioritizing critical issues is not important when resolving issues in a data model.
5. Documenting a data model helps in maintaining its consistency and accuracy.

6. Client approval ensures that the data model meets the client's requirements.

Reference

Books

Database Design - 2nd Editionby Adrienne Watt and Nelson
Database Management Systems by Raghu Ramakrishnan and Johannes Gehrke

Open-Source Database Design by Max Ortiz Catalan: at

Data Modeling and Relational Database Design by University of California, Berkeley.

URL

https://opentextbc.ca/database/.
https://www.db-book.com/.
https://www.dcs.bbk.ac.uk/~ptw/teaching/DBM/er.pdf
https://www2.eecs.berkeley.edu/Courses/CS186/
https://www.geeksforgeeks.org

Developer's Profile

No	Name	Qualification	Field of Study	Organization/ Institution	Mobile number	E-mail
1	Frew Atkilt	M-Tech	Network & Information Security	Bishoftu Polytechnic College	0911787374	frew.frikii@gmail.com
2	Gari Lencha	MSc	ICT Managment	Gimbi Polytechnic	0917819599	Garilencha12@gmail.com
3	Kalkidan Daniel	BSc	Computer Science	Entoto Polytechnic	0978336988	kalkidaniel08@gmail.com
4	Solomon Melese	M-Tech	Computer Engineering	M/G /M/Polytechnic College	0918578631	solomonmelese6@gmail.com
5	Tewodros Girma	MSc	Information system	Sheno Polytechnic College	0912068479	girmatewodiros@gmail.com
6	Yohannes Gebeyehu	BSc	Computer Science	Entoto Polytechnic College	0923221273	yohannesgebeyehu73@gmail.com