## Define data, database, database management system.

**Data:** In computer science, data is anything in a form suitable for use with a computer.

**Database:** A database is a collection of information that is organized so that it can easily be accessed, managed and updated.

**Database Management System (DBMS):** DBMS is a system to achieve an organized store of a large number of dynamically associated data. In other words, DBMS is a computer application that interacts with database, end-user, other applications to capture and analyze data.

## Different types of database

### 1. Hierarchical database model

It is a data model in which the data is organized into a tree-like structure. The data is stored as records which are connected to one another through links.

### 2. Network database model

It is a type of database model wherein multiple member records can be linked to multiple owner files and vice-versa. The model can be viewed as an upside-down tree where each member information is the branch linked to the owner, which is the bottom of tree.

Nayeem Mahmood / 4AM / C161026

3. Relational Database

A relational database is based on relational model and uses a collection of tables to represent both data and the relationships among those data.

4. Embedded Database

It is a system which is tightly integrated with an application software that requires access to stored data, such that the database system is hidden from the application's end-user and requires little or no ongoing maintenance.

5. Distributed Database

It is a database system in which storage devices are not all attached to a common processor. It may be stored in multiple computers, located in the same physical location or may be dispersed over a network of interconnected computers.

⊞ Explain the problems of file processing system. How DBMS can solve these problems?

The problems of file processing system:

1. Data redundancy and inconsistency: If more than one application software generates same pieces of information but in different files, there occurs data redundancy and inconsistency.

2. Difficulty in accessing data: Conventional file-processing environments don't allow needed data to be retrieved in a convenient ~~way~~ and efficient manner.

Nayeem Mahmood / 4AM / C161026

3. Data Isolation: Because data are scattered in various files, and files may be in different formats, writing new application programs to retrieve the appropriate data is difficult.

4. Integrity Problems: When new constraints are added to a system, it is difficult to change the programs to enforce them. The problem is compounded when constraints involve several data items from different files.

5. Atomicity Problem: It is difficult to ensure atomicity in file processing system. For example, transferring $500 from account A to account B. If a failure occurs during execution, there could be a situation like $500 is deducted from account A but not credited to B.

6. Concurrent Access Anomalies: If multiple users are updating the same data simultaneously, it will result in inconsistent data state. In file-processing system, it is very difficult to handle this situation using program. Database system overcome it by time sharing system.

7. Security Problem: Enforcing security constraints in file-processing system is very difficult as the application programs are added to the system in an ad-hoc manner.

## Write down different application areas of DBMS.

1. Sales
2. Accounting
3. Human Resource Management
4. Online Retailers
5. Banking
6. Credit Card Transactions
7. Finance
8. Universities
9. Airlines
10. Telecommunications
11. Intelligence Agencies
12. Social Networking

## Define DDL, DCL and DML

DDL: A data definition language or data description language is which has similar syntax to computer programming language for defining data structures, specially database schemas.

e.g: CREATE TABLE student (S_Name varchar(20),
S_Roll int )

This DDL statement creates table student with field S_Name and S_Roll.

DCL: A data control language which has similar syntax to computer programming language and is used to control the access to data stored in database.

DML: It refers to Data Manipulation Language. It is used to manipulate data itself. Data manipulation includes data retrieval, insertion, deletion, modification of information stored in the database.

DML is of 2 types.

1. Procedural DMLs: It requires a user to specify what data are needed and how to get those data.
2. Declarative DMLs: It requires a user to specify what data are needed without specifying how to get those data (also referred as to as non-procedural DMLs).

## Who are the database users?

There are 4 different types of database-system users, differentiated by the way they expect to interact with the system. They're -

1. Naïve Users: They are unsophisticated users who interact with the system by invoking one of the application programs that have been written previously.
2. Application Programmers: They are computer professionals who write application programs.
3. Sophisticated Users: They interact with the system without writing programs. Instead, they form their requests either using a database query language or by using tools such as data analysis software.

Nayeem Mahmood / 4AM / C161026

4. Specialized Users: They are sophisticated users who write specialized database applications that don't fit into the traditional data-processing framework.

## Who is a database administrator? What are the main functions of a database administrator?

DBA: A person who has central control over both the data and the programs that access those data, is called Database Administrator (DBA).

The functions of DBA:

1. Schema definition
2. Storage structure and access-method definition
3. Schema and physical-organization modification
4. Granting of authorization for data access.
5. Routine maintenance
   a. Periodically backing up the database.
   b. Ensuring enough free disk space is available.
   c. Monitoring jobs running on the database

## What is data abstraction? Describe different levels of data abstraction and write some importance.

Data Abstraction: Data abstraction in DBMS means hiding implementation details (i.e. high level details) from end-users.

For example, in case of storage of data in database, users can only access database, but implementation details such as how the data is stored physically onto disk is hidden from users.

Different levels of Data Abstraction :

1. Physical Level : The lowest level of abstraction which describes how the data are actually stored.
2. Logical Level : The next higher level of abstraction which describes what are stored in the database and what relationships exist among those data.
3. View Level : The highest level of abstraction which describes only part of the entire database. The system may provide many views for the same database.

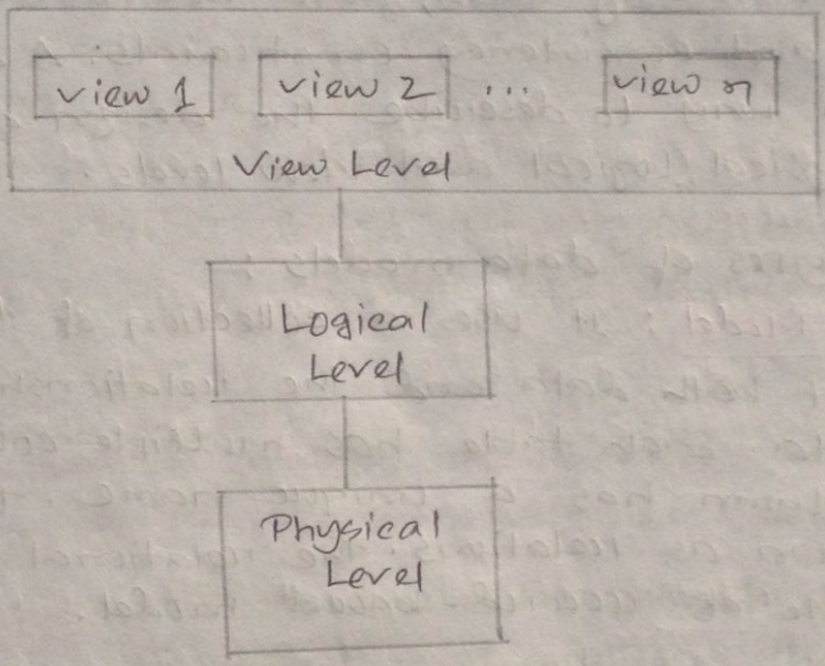



Fig: The 3 levels of Data Abstraction

## Importance of Data Abstraction

1. It helps to retrieve data efficiently.
2. It ensures the security of database by hiding the complexities from users.

## Define data model. Describe different types of data model.

**Data model:** Underlying the structure of database is the data model which is a collection of conceptual tools for describing data, data relationships, data semantics and consistency constraints. A data model provides a way to describe the design of a database at the physical, logical and view levels.

Different types of data models:

1. **Relational Model:** It uses a collection of tables to represent both data and the relationships among those data. Each table has multiple columns and each column has a unique name. Tables are also known as relations. The relational model is an example of record-based model.

2. **Entity-Relationship Model:** E-R data model uses a collection of basic objects, called entities and relationships among these objects. An "entity" is a "thing" or "object" in the real world that is distinguishable from other objects.

3. Object-Based Data Model: This is an extended version of E-R model with notions of encapsulation, functions and object identity.

4. Semi-structured Data Model: It permits the specification of data where individual data items of the same type may have different sets of attributes. The Extensible Markup Language (XML) is widely used to represent semi-structured data.

## Define Instances and Schemas:

Instances: The collection of information stored in database at a particular moment is called an instance of database.

Schemas: The overall design of the database is called the database schema. Schemas are changed infrequently, if at all.

## What is Storage Manager in Database? Write the components of it.

Storage Manager: The storage manager of a database system is the component that provides the interface between low-level data stored in the database and the application programs and queries submitted to the system. The storage manager is responsible for the interaction with the file manager.

The storage manager components include:

1. Authorization and integrity manager, which tests for the satisfaction of integrity constraints and checks the authority of users to access data.
2. Transaction manager, which ensures that the database remains in a consistent (correct) state.
3. File manager, which manages the allocation of space on disk storage.
4. Buffer manager, which is responsible for fetching data from disk storage into main memory.

## What is Query? Write down the steps/components of Query Processing.

Query: A query is a statement requesting the retrieval of information. The portion of a DML that involves information retrieval is called Query Language.

Components of Query Processing:

1. DDL Interpreter: It interprets DDL statements and records the definitions in the data dictionary.
2. DML Compiler: It translates DML statements in a query language into an evaluation plan consisting of low-level instructions that the query evaluation engine understands.
3. Query Evaluation Engine: It executes low-level instruction generated by a DML compiler.

## Define entity and entity set.

**Entity:** An entity is a "thing" or "object" in the real world that is distinguishable from all other objects.

For example, each person in an enterprise is an entity.

**Entity set:** An entity set is a set of all entities of same type that share the same properties or attributes.

For example, all persons having an account at a bank which can be defined as the entity set: customer.

## Define attributes. Write about different types of attributes with examples.

**Attributes:** Entities are described in a database by a set of attributes.

For example, the attributes dept_name, buildings and budget may describe one particular department in a university. and they form attributes of the department entity set.

Types of attributes:

1. **Simple Attribute:** They are not divided into sub-parts.

   For eg., Customer_ID.

2. **Composite Attribute:** They are divided into sub-parts.

   For eg., Customer_Name could be structured as a composite attribute consists of First_Name, Middle_Name and Last_Name.

3. Single-valued attribute: They are called single-valued attribute if they consist of single values for a particular entity.

For e.g., ID, Date_of_Birth.

4. Multi-valued Attribute: If an attribute consists of a set of values for a particular entity, it is called Multi-valued attribute.

For e.g., mobile_number, email.

5. Null Attribute: A null value is used when an entity doesn't have a value for an attribute.

For e.g., if a particular employee have no dependents, then dependents_name value for that employee will be null.

6. Derived Attribute: An attribute is called derived one if the value for this attribute can be derived from the values of other related attributes or entities.

For e.g., student_age might be called as derived attribute if there is another attribute student_dob, because by using date of birth, age can be calculated.

7. Stored Attribute: The attributes which gives the value to get the derived attribute are called stored attribute. In example above, age is derived using date of birth. Hence student_dob is a stored attribute.

8. Descriptive Attribute: Attributes of the relationship is are called descriptive attributes.

For eg., employee works for department. Here 'works for' is the relation between employee and department ~~entity~~ entities. The relation 'works for' can have attribute date_of_join which is a descriptive attribute.

## Define Relationship set.

Relation is an association among several entities. A relationship set is a set of relationships of the same type.

Formally, it is a mathematical relation on 'n' number of entity sets where $n \geq 2$. If $E_1, E_2, \dots E_n$ are entity sets, then a relationship set R is a subset of

$$\{(e_1, e_2, \dots, e_n) \mid e_1 \in E_1, e_2 \in E_2, \dots e_n \in E_n\}$$

## Explain cardinalities.

Mapping cardinalities or cardinality ratios express the number of entities to which another entity can be associated via a relationship set.

It is most useful in describing binary relationship sets.

For a binary relationship set R between entity sets A and B, the mapping cardinalities must be one of the followings:

1. One to One: An entity in A is associated with at most one entity in B and an entity in B is associated with at most one entity in A.

2. One to Many: An entity in A is associated with any number of entities in B and an entity in B is associated with at most one entity in A.

3. Many to One: An entity in A is associated with at most one entity in B and an entity in B is associated with any number of entities in A.

4. Many to Many: An entity in A is associated with any number of entities in B and an entity in B is associated with any number of entities in A.

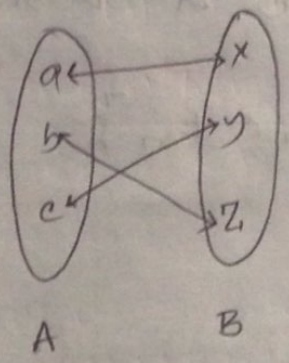



Fig: One to One

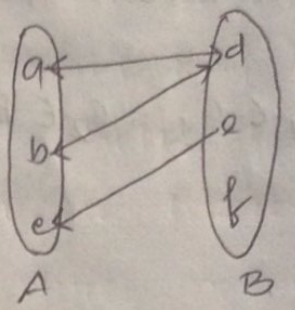



Fig: Many to One

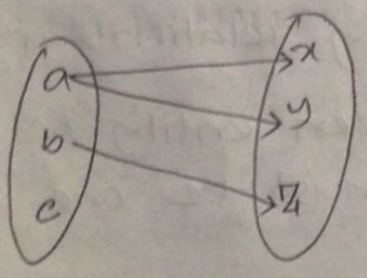



Fig: One to Many

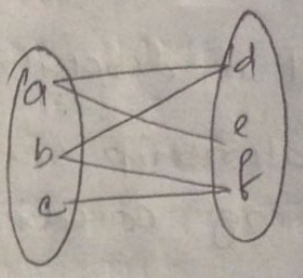



Fig: Many to Many

Nayeem Mahmood / 4AM / C161026

## Define and explain Participation constraints:

Participation constraints represents how an entity is involved in the relation.

**Total Participation:** If all the entity values are participating in any relation, then it is called total participation.

**Partial Participation:** If only few values of an entity is a part of relation, then it is partial participation.

For eg., 'Employee works for a Department'. Here, all the employees work for one or the other department. No employee is left without department. Hence, all the employees are participating in this relation. Thus, participation of employees is total participation.

But individual department will not have all the employees. Each department will have only few groups of employees working. Hence, partial participation of department is seen here.

Nayeem Mahmood / 4AM / C161026

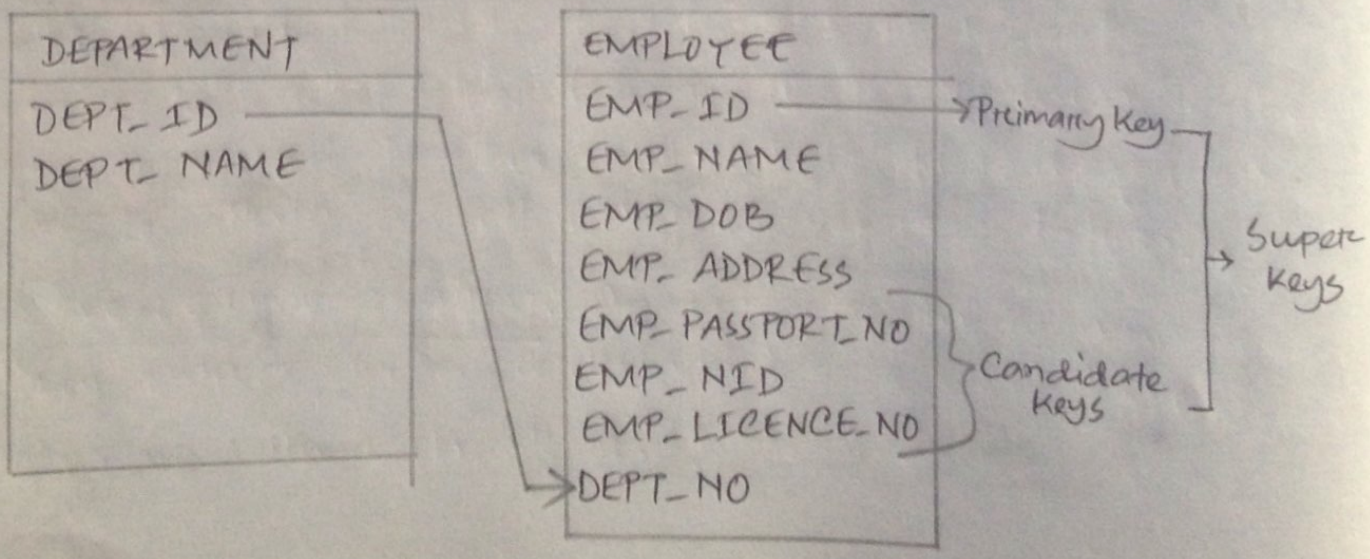
[4] What are Keys? Explain different types of keys with suitable examples.

**Keys:** Keys are the attributes of entity, which uniquely identifies the record of the entity.

Different types of keys:

1. **Super key:** Super key is the one or more attributes of the entity, which uniquely identifies the record in the database.
2. **Candidate key:** Candidate key is one or more set of keys of the entity. ~~For a person entity entity, it~~
3. **Primary key:** Primary key is the candidate key, which will be used to uniquely ~~identified~~ identify a record by the query.
4. **Foreign keys:** A primary key is called a foreign key when it helps to establish the mapping between two or more entities.

Let us consider two entities named EMPLOYEE and DEPARTMENT.



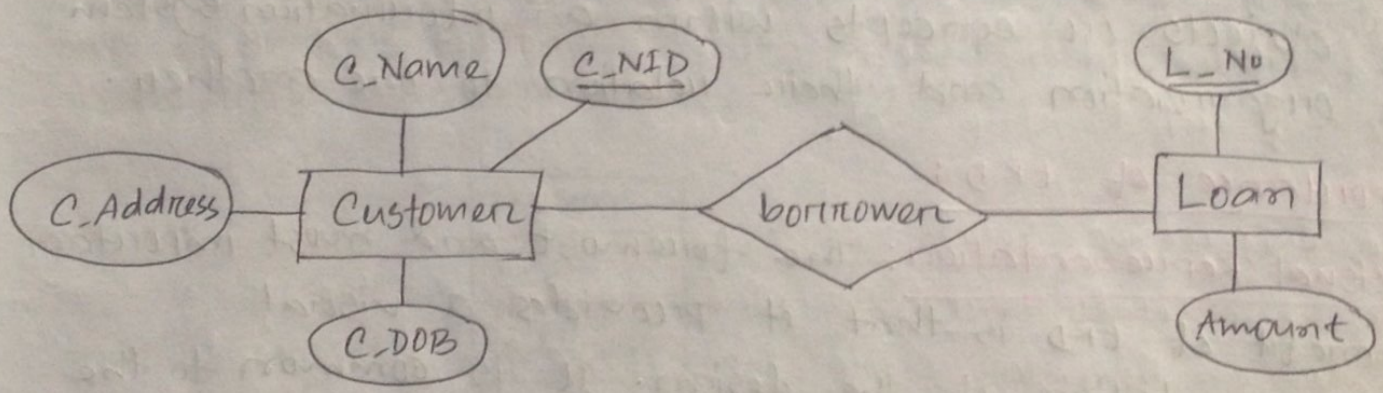
Here, an employee can be identified uniquely by EMP_ID, EMP_PASSPORT_NO, EMP_NID, EMP_LICENCE_NO. If EMP_ID is chosen as Primary key, then rest of them are candidate keys. The combined set of Primary key and Candidate keys builds a set for Super keys.

The Primary key 'DEPT_ID' from DEPARTMENT entity helps to build a relation with EMPLOYEE entity. Thus this 'DEPT_ID' is referred to EMPLOYEE entity as a foreign key.

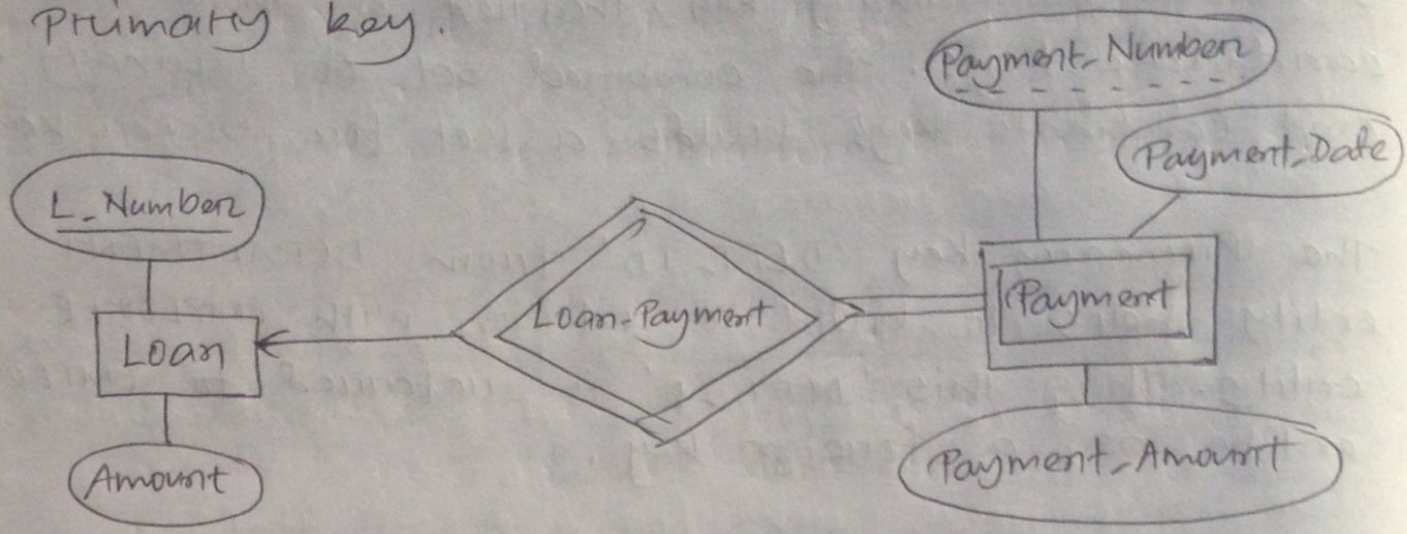
## Define Strong Entity & Weak Entity.

**Strong Entity:** An entity set is said to be strong if it has a primary key.

For e.g.,



**Weak Entity:** An entity set is said to be weak if it may not have sufficient attributes to form a primary key.




## What is ERD? What are the importances of ERD?

**ERD:** An entity-relationship diagram (ERD) is a type of data modeling that shows a graphical representation of objects or concepts within an information system or organization and their relation to one another.

**Importance of ERD:**

1. **Visual Representation:** The foremost and most important benefit of ERD is that it provides a visual representation of the design. It is common to the ERD being used together with data flow diagrams so as to attain a better visual representation.

2. **Effective Communication:** An ERD clearly communicates the key entities in a certain database and their relationship with each other.

3. Simple to Understand: ERD is easy to understand & simple to create.

4. High Flexibility: The ERD is quite flexible to use as other relationships can be derived easily from existing one.

## Explain Specialization and Generalization with necessary figure.

Specialization: An entity set may ~~have~~ include subgroupings of entities that are distinct in some way from other entities in the set. The process of designating subgroupings within an entity set is called Specialization.

Generalization: In basic term, Generalization is a procedure of removing normal attributes from two or more subgroups of entity sets and joining them into a summed up entity set.

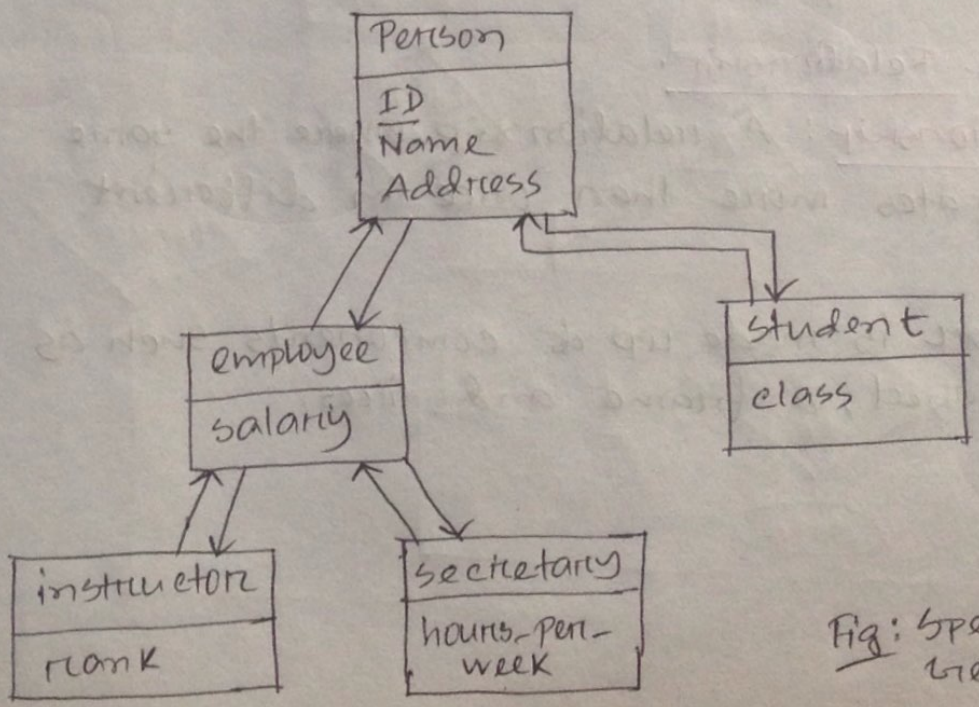



Fig: Specialization & Generalization

Here, specializations are indicated by down-arrows(↓) and generalizations is indicated by up-arrows(↑). It is clear that specialization is a top-down design process whereas generalization is a bottom-up design process.

## What do you know about the Roles?

We indicate roles in ER diagrams by labeling the lines that connect diamonds to rectangles.

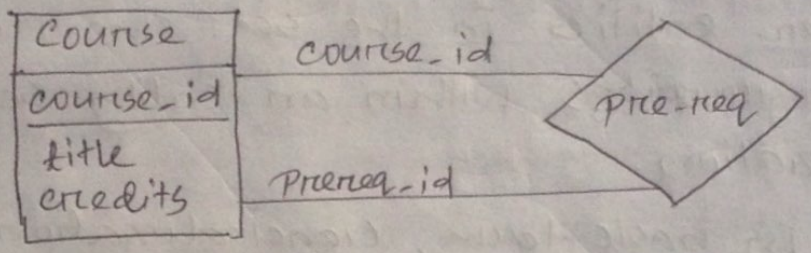



Figure above shows the role indications course_id and prereq_id between course entity set and the pre_req relationship set.

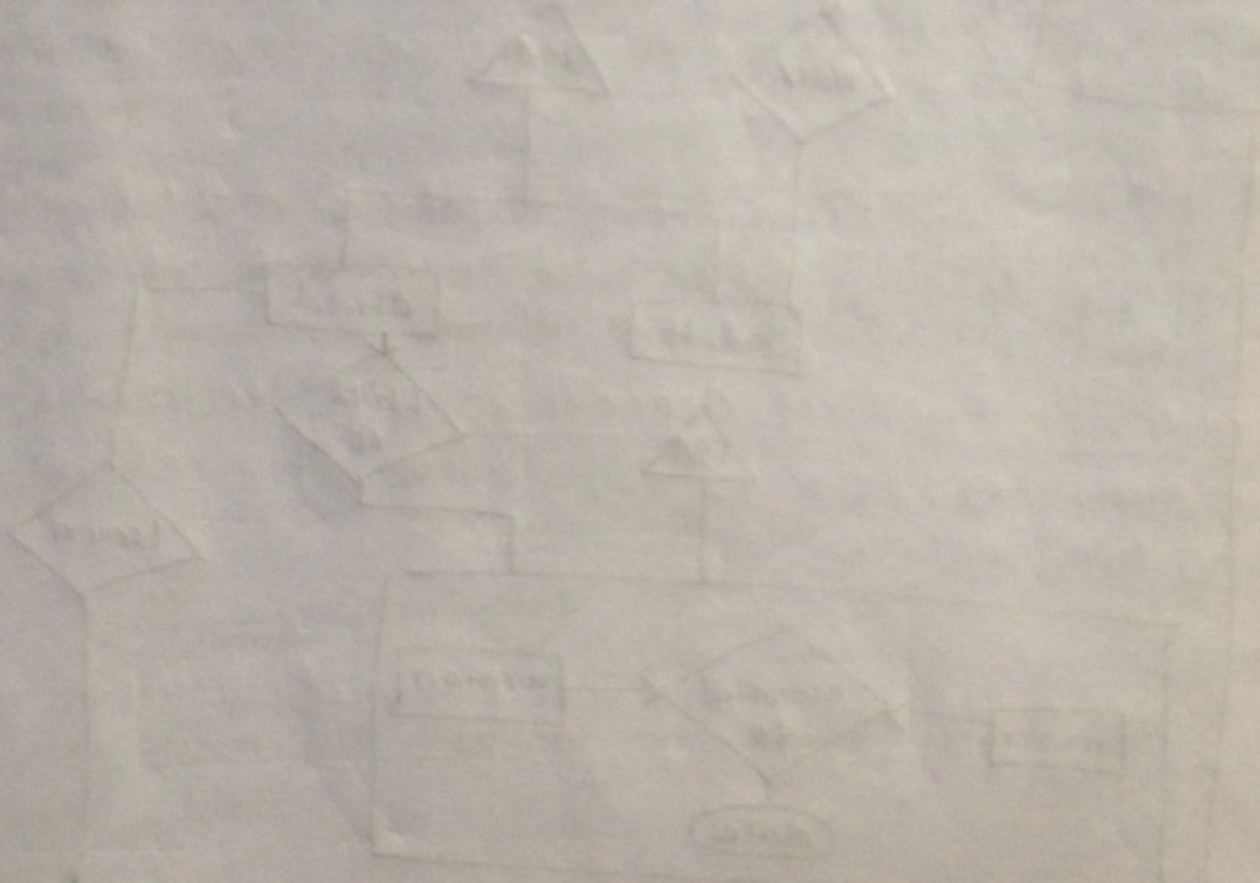
## Define Recursive Relationship.

**Recursive Relationship**: A relationship where the same entity participates more than once in different roles.

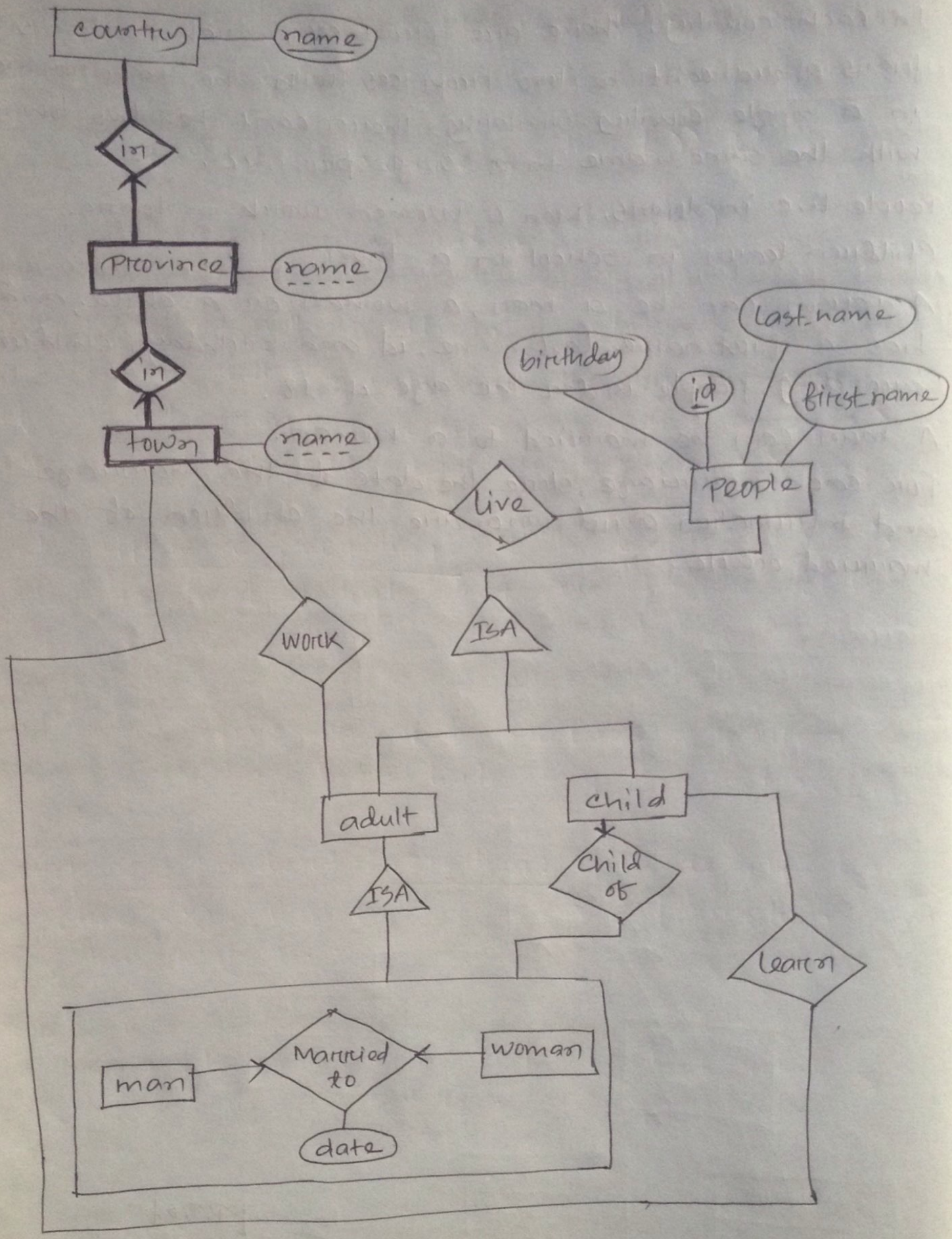
For e.g., a car is made up of components such as a steering wheel, a frame and tires.

Draw ER Diagram using following information.

1. In each country, there are provinces, which contains towns. There can't be two provinces with the same name in a single country. Similarly, there can't be two towns with the same name in a single province.
2. People live in towns. Men & Women work in towns. Children learn in school in a town.
3. A person can be a man, a woman or a child, and has a first name, last name, id and birthday. Children are any people under the age of 18.
4. A man can be married to a woman.
5. For each marriage, store the date of the marriage and information about who are the children of the married couple.



[P.T.O]




Nayeem Mahmood / 4AM / C161026

[4] Define Relational Algebra. Explain the fundamental operations of the Relational Algebra with examples.

**Relational Algebra:** It is a procedural query language that consists of a set of operations and take one or two relations as input, produces a new relation as output.

There are 6 fundamental operations of R.A. They are-

1. **Select Operation:** The select operation selects tuples that satisfy a given predicate. Greek letter sigma ($\sigma$) is used to denote selection. (lower case)

Example:

$$\sigma_{salary \geq 1000}(employee)$$

This operation will select those tuples of the employee relation where the salary of employee is 1000 ~~and~~ or above.

2. **Project Operation:** This is a unary operation that returns its argument relation, with certain attributes left out. Upper case Greek letter pi ($\pi$) is used to denote projection.

Example:

$$\pi_{Name}(employee)$$

This operation will project on "Name" field of "employee" table.

3. **Union Operation**: It is used to gather data from ~~two~~ one or more tables, denoted by $\cup$.

Example:

$$\Pi_{roll}(\sigma_{dept='CSE' \wedge CGPA \geq 3.5}(students)) \cup$$

$$\Pi_{roll}(\sigma_{dept='EEE' \wedge CGPA \geq 3.5}(students))$$

This operation will result in a relation/table with 1 field "roll" which are the roll numbers of students from CSE dept. having CGPA ~~equal to~~ $\geq 3.5$ or students from EEE dept. having CGPA $\geq 3.5$.

4. **Set-Difference Operation**: It is denoted by '$-$', allows us to find tuples that are in one relation but not in another.

Example:

The expression $r - s$ produces a relation containing those tuples in $r$ but not in $s$.

5. **Cartesian Product Operation**: It is denoted by a cross ($\times$), allows us to combine information from any two relations.

Example:

We write the Cartesian Product of relations $r_1$ and $r_2$ as $r_1 \times r_2$.

6. Rename Operation: It is used to give a name or rename a relation, denoted by the lowercase Greek letter rho ($\rho$).

Example:

$\rho_x(E)$ → this will give name 'x' to the result of expression E.

$\rho_{x/y}(E)$ → this will rename the result of expression E from 'y' to 'x'.

## Write down the formal definition of RA. List Fundamental, Additional and Extended RA operations.

Formal Def$^n$ of RA: A basic expression in RA consists of either one of the following:

- → A relation in the database
- → A constant relation

### Additional Operations

1. Set-Intersection Operation
2. Natural-Join Operation
3. Assignment Operation
4. Outer Join Operation
5. Division operation

### Extended Operations

1. Generalized Projection
2. Aggregation

## Explain Left, Right and Full Outer Join.

**Left Outer Join:** Left outer join ( ⟕ ) takes all tuples in the left relations that did not match with any tuple in the right relations, pads the tuples with null values for all other attributes from the right relation and adds them to the result of the natural join.

**Right Outer Join:** Right outer join ( ⟖ ) takes all tuples in the right relation that did not match with any tuple in the left relation, pads the tuples with null values for all other attributes from the left relation and adds them to the result of the natural join.

**Full Outer Join:** Full outer join ( ⟗ ) does both the left and right outer join operations

## Set Intersection is not a fundamental RA operation - Explain.

Set Intersection operation is a binary operation which is denoted by $\cap$, allows us to find the tuples those are in both relationships.

$$r = r_1 \cap r_2$$

This results in relation 'r' containing tuples which are common in $r_1$ & $r_2$.

We can re-write this expression as

$r_1 - (r_1 - r_2)$

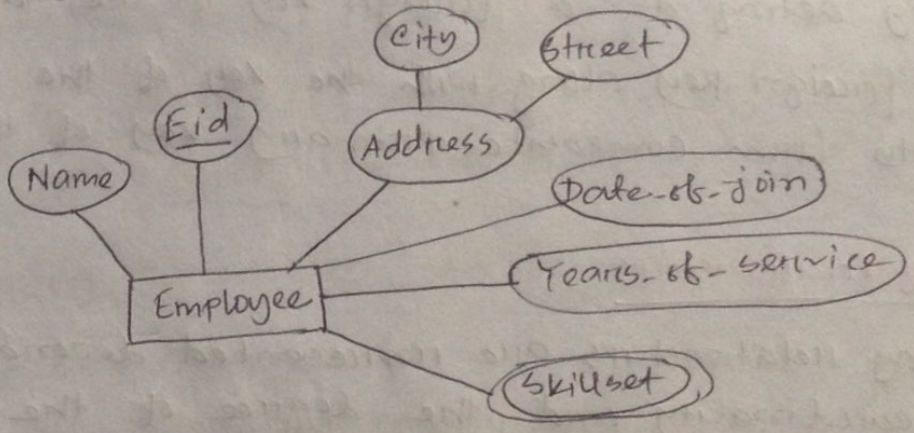
That means, above two expressions are equivalent. Thus, set intersection is not a fundamental operation and doesn't add any power to the relational algebra. It is simply more convenient to write $r_1 \cap r_2$ than to write $r_1 - (r_1 - r_2)$.

## How could you map an ER diagram to Database?

There are several steps.

### 1. Converting Strong Entity Types

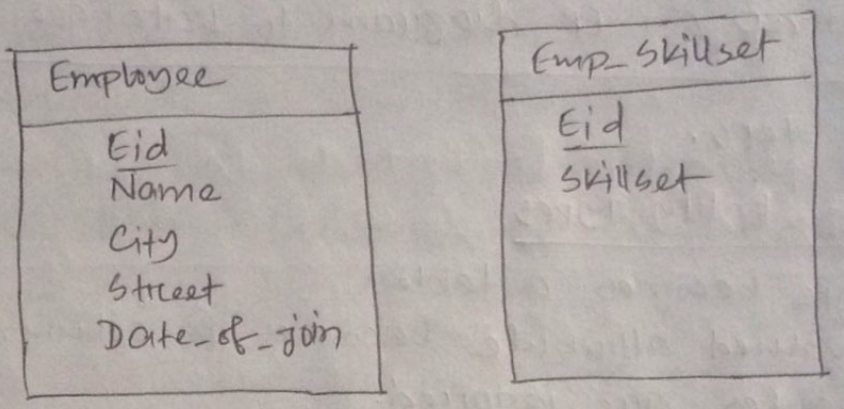
a) Each entity type becomes a table.
b) Each single-valued attribute becomes a column.
c) Derived attributes are ignored.
d) composite attributes are represented by components.
e) Multi-valued attributes are represented by a separate table.
f) The key attribute of the entity type becomes the primary key of the table.



→ Here, address is a composite attribute.
→ Years_of-service is derived attribute (can be calculated from date_of-join and current date)
→ skill set is a multi-valued attribute.

So, the relational schema be

Employee (Eid, Name, City, Street, Date_of-join)
Emp_Skillset (Eid, Skillset)




## 2. Converting Weak Entity Types

a) Weak entity types are converted into a table of their own, with the primary key of the strong entity acting as a foreign key in the table.

b) This foreign key along with the key of the weak entity form composite primary key of that table.

## 3) Converting Relationships

a) The way relationships are represented depends on the cardinality and the degree of the relationship.

b) The possible cardinalities are : 1:1, 1:M, M:N

c) The degrees are : Unary, Binary, Ternary

[H] What is nested subquery? Give two examples of nested sub-query where set membership and set comparison are used.

Nested Sub-Query : A subquery that is nested inside a SELECT, INSERT, UPDATE or DELETE statement or inside another sub-query.

Subquery Set membership

It is used to check if value of expression is matching a set of values produced by a subquery. For this, IN keyword is used.

e.g.,

```
SELECT NAME FROM EMPLOYEES WHERE EMPLOYEE IN (SELECT
COMPANY FROM COMPANIES WHERE AVG_SALES > TOTAL_SALES)
```

Sub-query Comparison

For comparing value of one expression to value of produced by subquery.

e.g.,

```
SELECT NAME FROM EMPLOYEES WHERE AVG_SALES > (SELECT
TOTAL_SALES FROM COMPANIES WHERE CITY = 'CTG')
```

Here, the following comparison can be done :

=, <>, <, <=, >, >=

where <> is the inequality comparison.

Spring- 16

④ (b)

i)

$$\Pi_{customer\_name}\left(\sigma_{branch\_name = \text{"Agrabad"}}\left(\sigma_{borrower.loan\_number = loan.loan\_number}\left(borrower \times loan\right)\right)\right)$$

ii)

$$\Pi_{customer\_name}(depositor) \cup \Pi_{customer\_name}(borrower)$$

iii)

$$\Pi_{loan\_number}\left(\sigma_{amount > 1200}(loan)\right)$$

iv)

$$\mathcal{G}_{MAX(balance)}\left(\sigma_{branch\_name = \text{"Muradpur"}}(account)\right)$$

Autumn - 15

3) (b)

1) $\pi_{s\text{-}id}\left[\sigma_{semester=\text{"Autumn"} \wedge year=2015 \wedge Dept\_name=\text{"CSE"}}(Enrolled\_by \times Offers)\right]$

2. $\pi_{c\text{-}name}\left[\sigma_{Dept\_name=\text{"BBA"}}(course \times Offers)\right]$

3. $\pi_{c\text{-}name}\left[\sigma_{s\text{-}id=01}(Enrolled\_by \times Courses)\right]$

4. $\pi_{Dept\_name, c\text{-}name, credit}\left[\sigma_{s\text{-}id=01}(Enrolled\_by \times Offers \times Courses)\right]$

Nayeem Mahmood / 4AM / C161026

Spring 15/17
4(b)

1. $\pi_{route\#, Date\text{-}of\text{-}journey, Time}\left[\sigma_{No\text{-}of\text{-}seats > 0}(Ticket \times BUS)\right]$

2. ~~$\pi_{No\text{-}of\text{-}seats}\left[\sigma_{B\text{-}id = "BUS1"}(Bus)\right]$~~

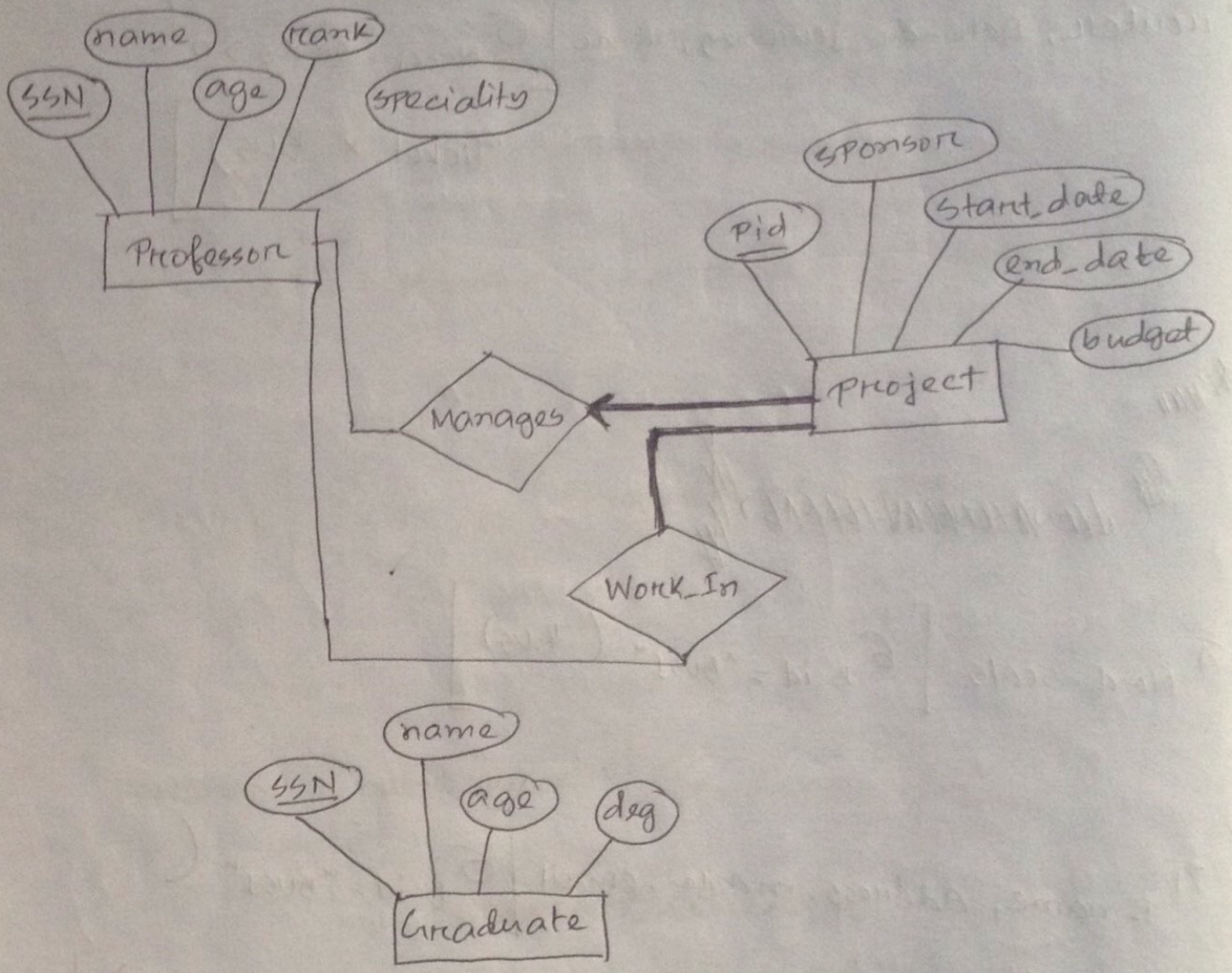
3. $\pi_{P\_name, Address, mobile, email}\left[\sigma_{B\text{-}id = "BUS1"}(Bus \times Ticket \times Passenger)\right]$

4. $\pi_{P\_name, Date\text{-}of\text{-}journey, Time, Seat\text{-}no, Amount}\left[\sigma_{P\_id = "P01"}(Passenger \times Ticket)\right]$

Nayeem Mahmood / 4AM / C161026

Autumn 16

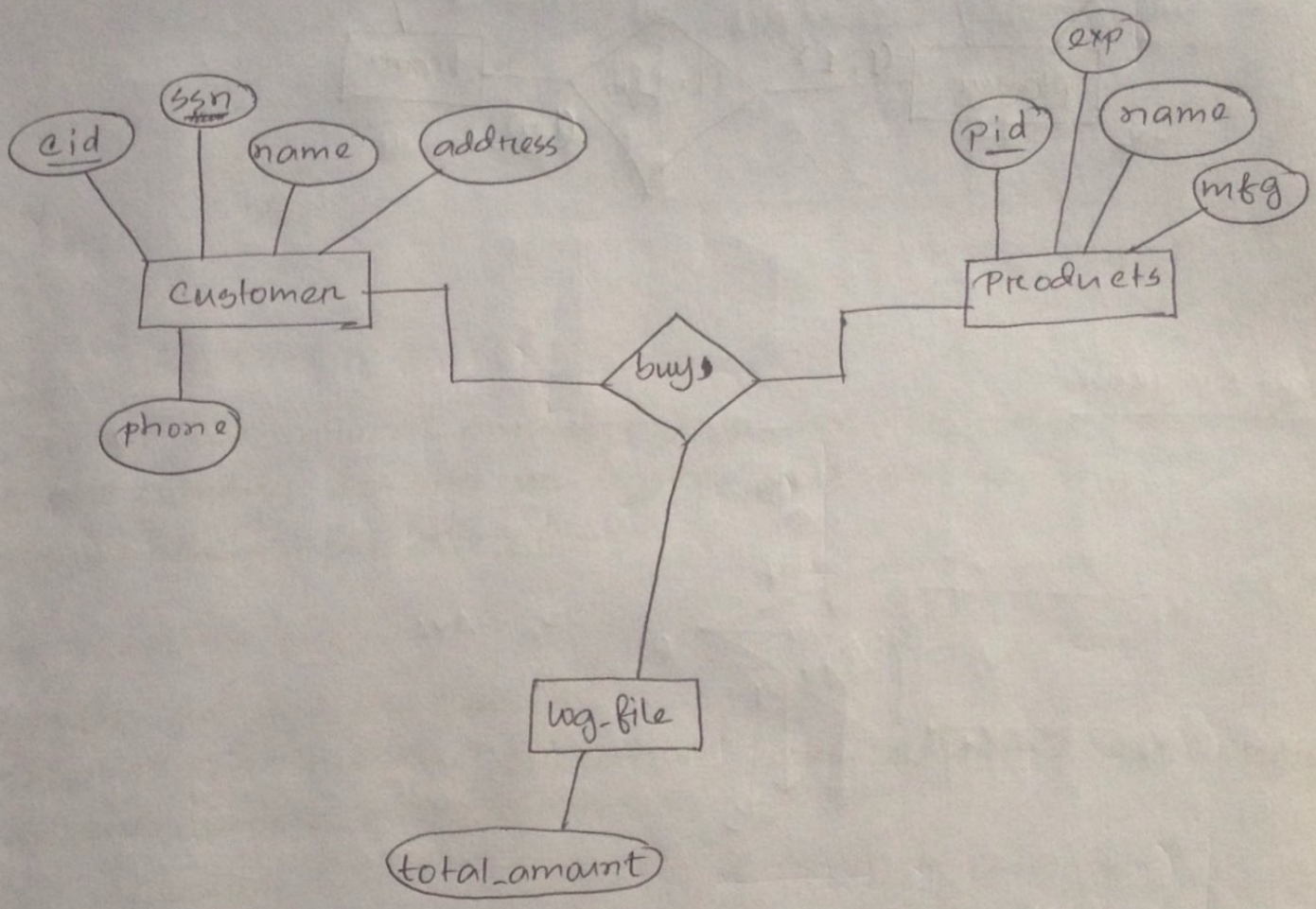
(2) (a)




Nayeem Mahmood / 4AM / C161026

Spring 16/17

2b/2c




Nayeem Mahmood / 4AM / C161026