**Ch 01 Part II Data Model**

**Data Model:** It is the collection of conceptual tools for describing Data, Data relationship, data Schema and consistency constraints.

**Data Models are classified into following categories:**

1. Object Based Logical Data Model
2. Record Based Logical Data Model
3. Physical Data Model

**Object Based Logical Data Model:** These models are used in describing data at Logical and View Level.

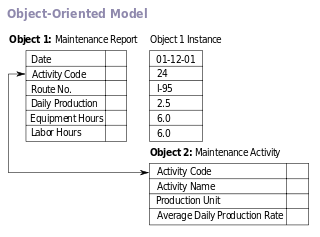
Following are the Object Based Logical Model

1. Entity Relationship Model(E-R Model)
2. Object Oriented Model
3. Semantic Data Model
4. Functional Data Model

**E-R Model:** It consist of a collection of basic object, called ***entities*** and ***relationship*** among these objects. ***An entity is an object that is distinguishable from other object by a specific set of attributes. A Relationship is an association among several entities.***

An object may contain, in addition to relations and entities, ER Model also represents certain constraints to which the contents of a database must conform. One important constraint is ***Mapping Cardinality,*** which express the number of entities to which another entity can be associated via a relationship set.

**Object Oriented Model:** it is based on a collection of objects. An object contains values stored in instance variables within the object and bodies of code that operate on object. The bodies of code are called Methods.

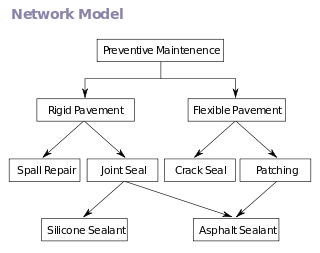


Objects that contain the same type of values and the same methods are grouped together into classes. A class may be viewed as a type definition for objects. The only way in which one object can access the data of another object is by invoking a method of that other object. This is called as sending message to the object.

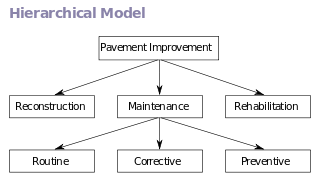
**Record Based Logical Model:** These are used in describing data at: **i.** Logical Data **ii.** View Level

In this model, data is structured in fixed format record of several types. Each type defines fixed a number of fields or attributes and each field is usually of fixed length. Following are the types of Record based Logical Model

1. **Relational Model**: It uses a collection of tables to represent both data and relationship among those data. Each table has multiple columns and each column has a unique name.
2. **Network Model**: In this, the data is represented as collection of records and relationship among data are represented by links, which can be viewed as pointers. The records in database are organized as collection of **arbitrary graphs**.



1. **Hierarchical Model**: It is similar to network model. In this is model the records in the database are organized as collections of **trees** rather than arbitrary graphs.



The hierarchical database model mandates that each child record has only one parent, whereas each parent record can have one or more child records. In order to retrieve data from a hierarchical database the whole tree needs to be traversed starting from the root node. This model is recognized as the first database model created by IBM in the 1960s.

**Physical Data Model:** It is used to describe data at the lowest level.

1. Unifying Model
2. Frame Memory model

**Entity-Relationship Model:** Basic Concepts: E-R Model is based on perception of a real world that consists of a set of basic objects called entities and relationship among these objects.

**Entity:**

* It is an object in the real world that is distinguishable from all other objects.
* An entity is a thing in the real world with an independent existence.
* It has a set of properties or attributes and the values for some set of attributes or properties may uniquely identify an entity.

e.g Student, Account are the entity

student with attributes (Roll\_No, Name, Address,Class,DoB)

Acccount(Acc\_no,name, Type)

**Entity Set**  is a set of entities of the same type that share the same properties or attributes. E.g The set of all students is defined as entity set student.

**Attribute:**

Each entity has a set of attributes. Each attribute has a domain from which the values for this attribute are drawn. Following are the attribute types:

**i. Simple and Composite Attribute**: Simple attribute cannot be divided into subparts. Composite attribute are divided into subpart.

e.g roll\_no is simple attribute while address which is subdivided into House\_no, street,lane,city,state is composite attribute.

**ii. Single Value and Multi Valued Attribute:** Single value attribute has single value while the multivalue attribute has multiple values. E.g. Cust\_no every customer has single cust\_no. While Phone\_no it may be possible that the customer will have more than 1 phone no hence phone no is multivalued attribute.

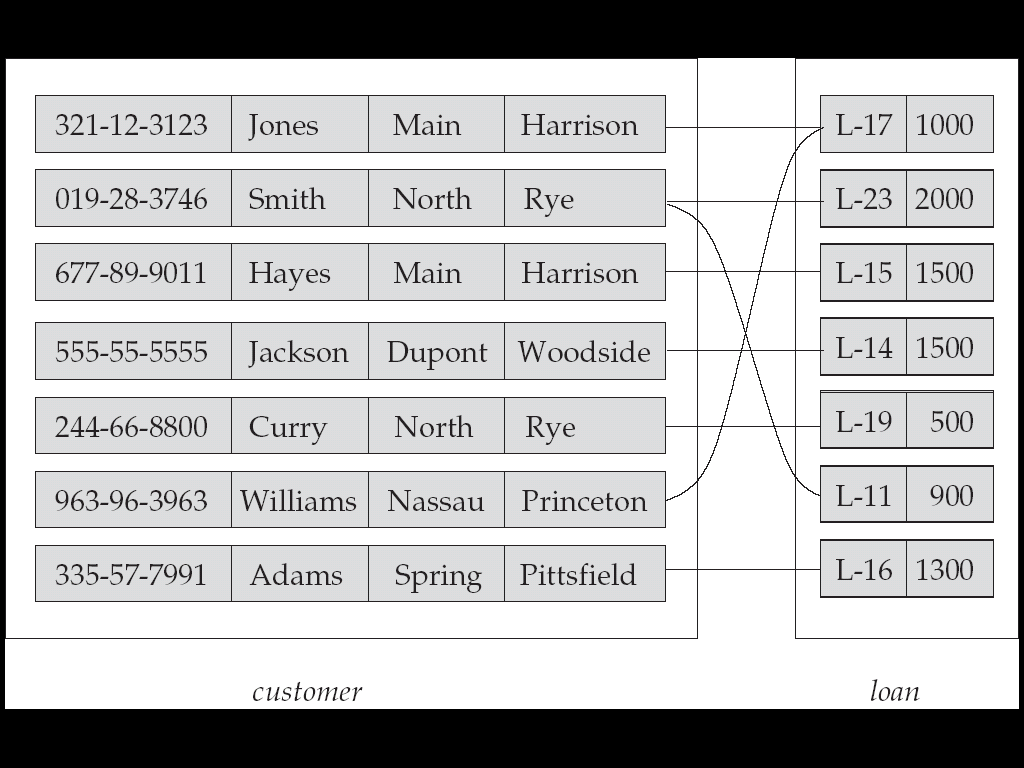
**iii. Null Attribute:** It may be possible that the value is not present for the specific attribute. It is also called as missing attribute. E.g. no\_of\_dependent attribute which is depends on the Person and person have no dependent.

1. **Derived Attribute:** The existence of these attributes will depends on another attribute. E.g age attribute. Age will depends on the date\_of\_birth and current\_date. So age is called derived attribute.

**Relationship Sets:**

A Relationship is an association among several entities. A relationship set is a set of relationships of the same type. It is a mathematical relation on n>=2(possibly distinct) entity set. If *E1,E2,……En and entity sets, then a relationship set R is a subset of*

*{(e1,e2…en)|e1ε E1,e2 ε E2,e3 ε E3)} where (e1,e2..en) is a relationship.*



Consider two entity set *customer* and *loan.* We define the relationship set **borrower** to denote the association between customer and the bank loans that the customer have.

**Participation:** The association between entity set is referred to as participation. i.e. the entity sets *E1,E2,…En* participate in relationship set R

**Relationship Instance:** in an E-R schema represents an association between the named entities in the real-world enterprise that is being modeled. E.g. From above tables **Hayes whose identifier is 677-89-9011, and the loan entity L-15 participate in a relationship instance of *borrower.***

***Entity Role:*** *The function that an entity plays in the relationship is called that entity’s role.*

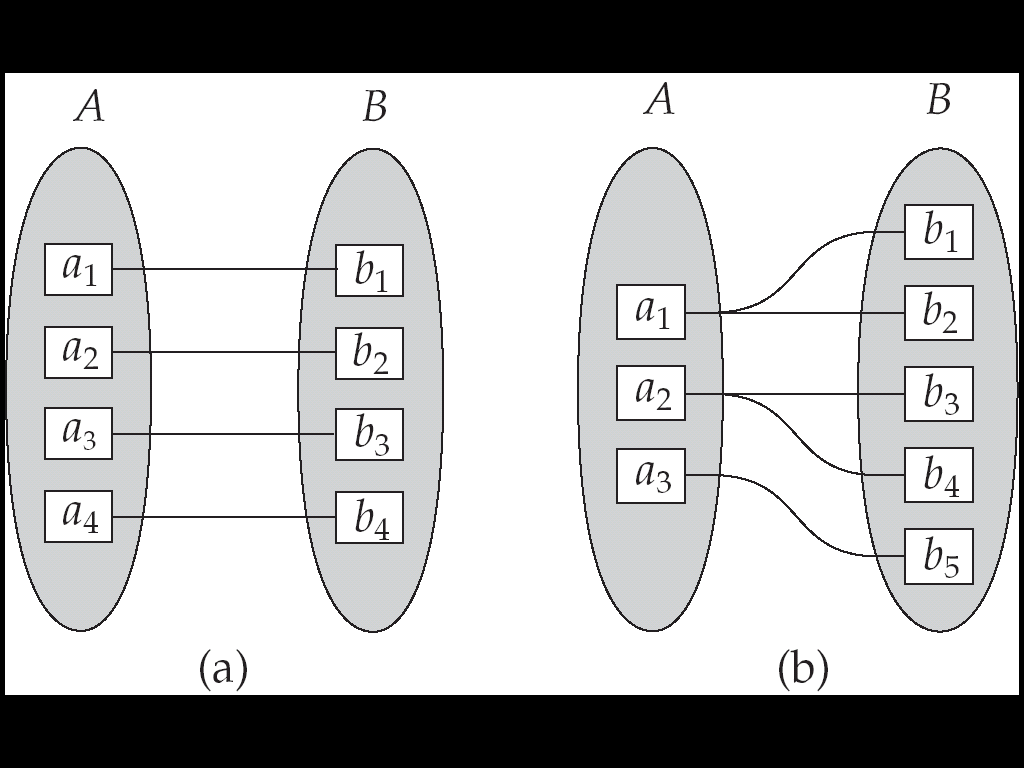
***Degree of Relationship Set:*** *The no. of entity sets that participate in a relationship set is also called the degree of the relationship set.*

***Binary Relationship****:* If there are two entities participate in the relation then it is called as Binary relationship.

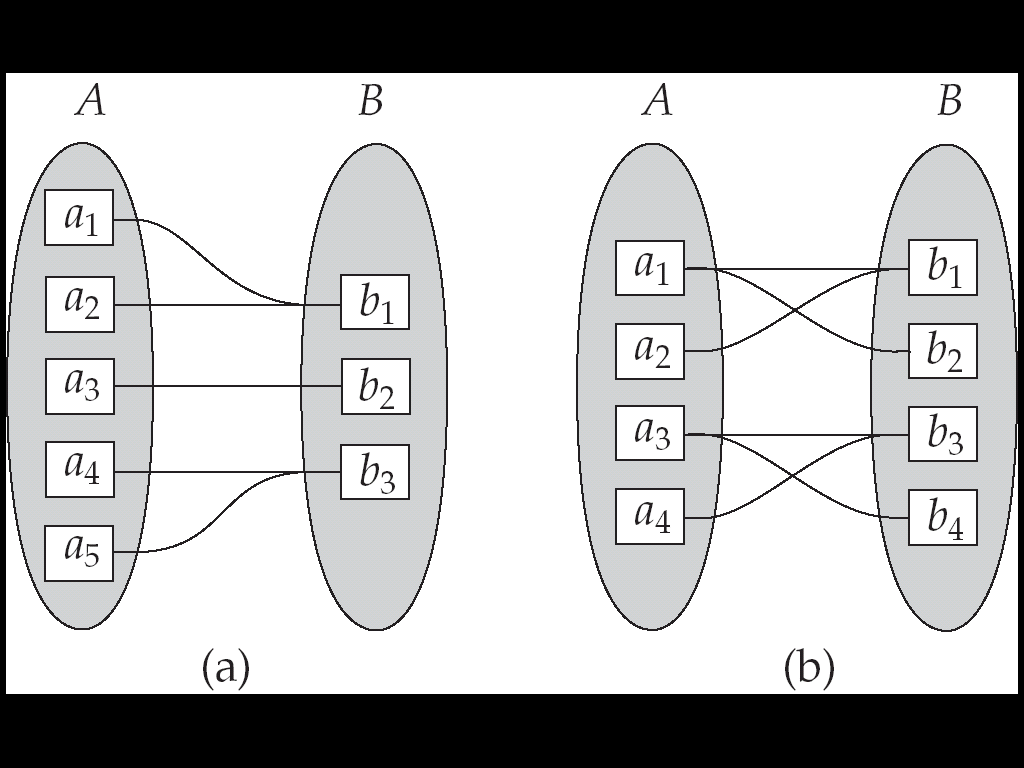
**n-ary Relationship** If there are more than two entities participate in the relation then it is called as **n-ary** relationship.

***Constraints:*** An E-R enterprise schema may define certain constraints to which the content of a database must conform.

1. **Mapping Cardinalities:** It express the no. of entities to which another entity can be associated via a relationship set. This is commonly binary relationship in the E-R diagram.
2. **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.
3. **One to Many**: An entity in A is associated with any number (zero or more) of entities in B. An entity in B, however, can be associated with at most one entity in A.
4. **Many to One**: An entity in A is associated with at most one entity in B. An entity in However, can be associated with at most one entity in A. An entity in B, however, can be associated with any number (zero or more) of entities in A.
5. **Many to Many:** An entity in A associated with any number (zero or more) of entities in B, and an entity in B is associated with any number (zero or more) of entities in A

****

**One to One One to Many**

****

**Many to one Many to Many**

**Participation Constraint:**

**Total Participation:**The participation of an entity set E in a relationship set R is said to be **Total** if every entity in E participates in at least one relationship in R.

**e.g.** We except every loan entity to be related to at least one customer through borrower relationship. Therefore the participation of loan entity in relationship borrow is **total participation**.

**Partial Participation:** If only some entities in E participate in relationship in R, the participation of entity set E in relationship R is said to be partial.

**e.g.** An individual is bank customer whether or not she has a loan with the bank. Hence, it is possible that only some of the customer entities are related to the loan entity set through the borrower relationship, and the participation of customer is borrow relationship is **partial**.

**Existence Dependency:** If the existence of entity ***x*** *depends on existence of entity* ***y****, then* ***x*** *is said to be existence dependent on* ***y****.* ***if y is deleted, x should also be deleted****. Entity y is said to be* ***dominant entity*** *and* ***x*** *is said to be* ***sub-ordinate entity****.*

*e.g. Consider Relation*

*account(acc\_no,balance)*

*Tranaction(Tranaction\_id,amount)*

*Log(Acc\_no, Tranaction\_id)*

*Every transaction is associated with one at least one account. But ,Each account may have more than one transaction. If the account is deleted then transaction related with that account is also deleted. Hence Account is called as dominant* ***(strong entity set)*** *while transaction is called as sub-ordinate* ***(weak entity set).***

**Keys:** To distinguish entities in the given entity set we have to use keys. Conceptually, individual entities are distinct; from a database perspective, however, the difference among them must be expressed in terms of their attributes.

The values of the attributes values of an entity must be such that they can *uniquely identify* the entity. Keys allowed us to identify a set of attributes that suffice to distinguish entities from each other.

Super Key: A super key is a set of one or more attributes that, taken collectively ,allow us identify uniquely an entity in the entity set. E.g. consider the following relation

*Customer (cust\_id,cust\_name,cust\_street) In this relationship cust\_id attribute of entity set customer is sufficient to distinguish one customer entity from another. Hence, cust\_id is superkey. Similarly, the combination of cust\_name and cust\_id is a superkey for the entity set customer. While the cust\_name is not sufficient because there are people with same name.*

*A superkey may contain extraneous attributes. If K is a superkey, then so is any superset of K.*

**Candidate key:** A superkey for which no proper subset is a superkey. Such minimal superkeys are called ***candidate key.***

It is possible that several distinct set of attributes could serve as a candidate key. Suppose that a combination *cust\_name* and *cust\_street* is sufficient to distinguish among members of the customer entity set. Then, both cust\_id, and {*cust\_name*, *cust\_street} are candidate keys.* The combination of the above can’t form the candidate key. Hence, cust\_id is candidate key because it alone identify the customer uniquely.

**Primary Key:** Primary key is a candidate key chosen by the database designer as the principle means of identifying entities within an entity set. The primary key does not accept the repeated values and null values. We must chose the primary key /candidate key carefully.

A entity set which has a primary key is called as **strong entity set**. Member of strong entity set is dominant entity and member of weak entity set is subordinate entity.

***Members of weak entity set are distinguished by a set of attributes which is called as discriminator of that weak entity set.***

*The primary key of weak entity set is formed by the primary key of strong entity set on which it is existence dependent plus the discriminator.*

**Relationship Set:** Let R be a relationship set involving entity sets E1, E2,E3,… En . Let P(Ei) denote primary key of Ei  entity set. If R has no attributes associated with it, then the set of attributes of R is given as:

P(E1) U P(E2)U P(E3)U….U P(En).

If R has attributes a1,a2….an associated with it then the set of attributes of R is

P(E1) U P(E2)U P(E3)U….U P(En)U{ a1,a2….an}

The primary key of relationship set depends on the mapping cardinality of the relationship set.

**Case 1:** If the relationship is many-to-many, primary key of **relationship set** is union of primary key of both entity sets. E.g. One customer having more than one account and one account is has more than one customer.

*Customer(cust\_name,social\_security)*

*Account(Acc\_no,balance)*

*Depositor(Social\_Security, Acc\_no)* ***Depositor is the relationship set***

**Case 2:** If the relationship is one-to-many or many-to-one , primary key of **relationship set** is primary key of one entity sets. E.g. One customer having more than one account/One account has more than one customer. The primary key of relationship set Depositor is **Acc\_no / Social\_Security.**

**Case 3:** In case of one-to-one relationship there is no primary key.

**Entity-Relationship Diagram:** Overall logical structure of database can be expressed graphically by an E-R diagram.

|  |  |  |
| --- | --- | --- |
| **Sr. No** | **Pictorial Representation** | **Description** |
| 1 | Rectangle | Represent Entity Set |
| 2 | Ellipse | Attributes |
| 3 | Double Ellipse | Multi-valued attribute |
| 4 | Dashed Ellipse | Derived Attribute |
| 5 | Double Rectangle | Weak Entity Set |
| 6 | Diamond | Relationship Set |
| 7 | Lines | Link attributes to entity set and relationship set |
| 8 | Double Lines | Indicate total participation of an entity in a relationship set |

**Showing Mapping cardinality in E-R Diagram:**

Consider the entities ***Customer(cust\_name,social\_security,address)***

***Account(Acc\_no,balance)***

***Depositor(Social\_security,acc\_no) ///*** *Relationship set*

**Many-to-Many**: One customer having more than one account and one account is has more than one customer.

**

**One-to-Many**: One customer may have any no. of accounts, but one account belongs to only one customer then

**

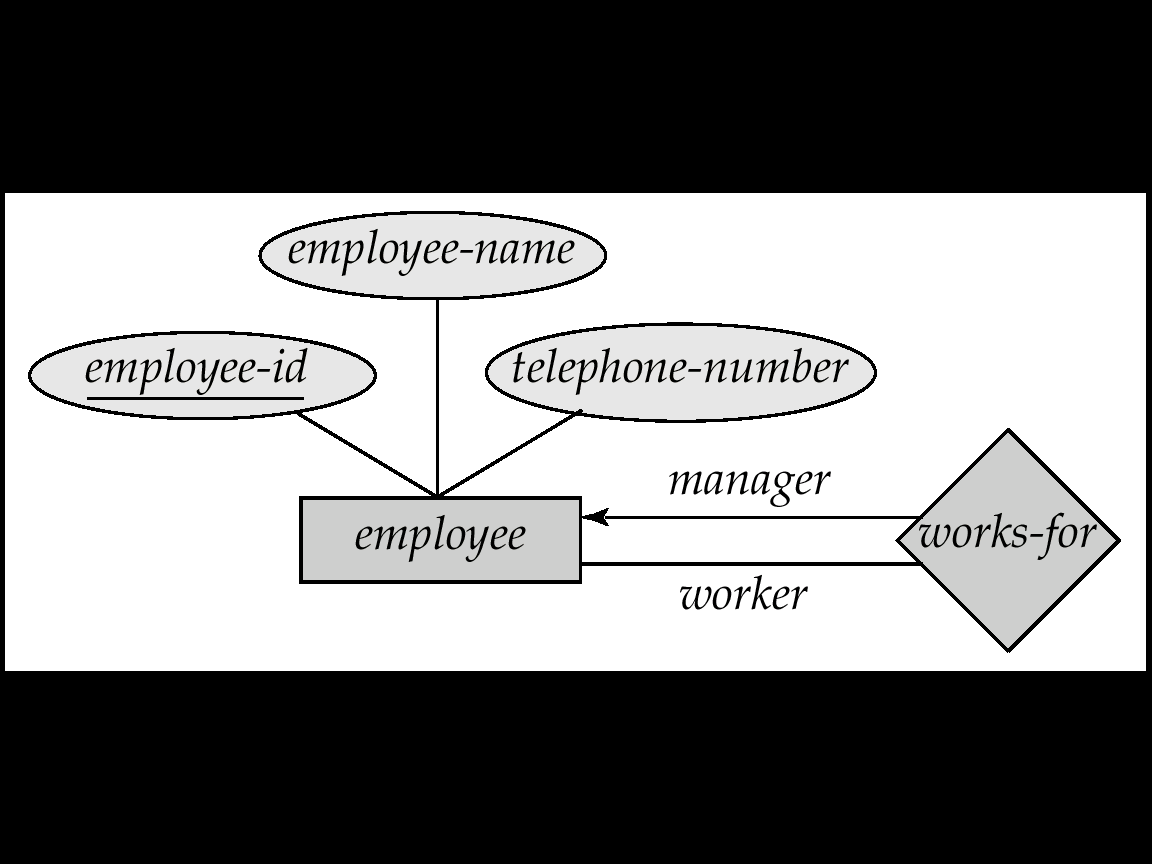
**Many-to-one**: Many Customers can share one account.

**

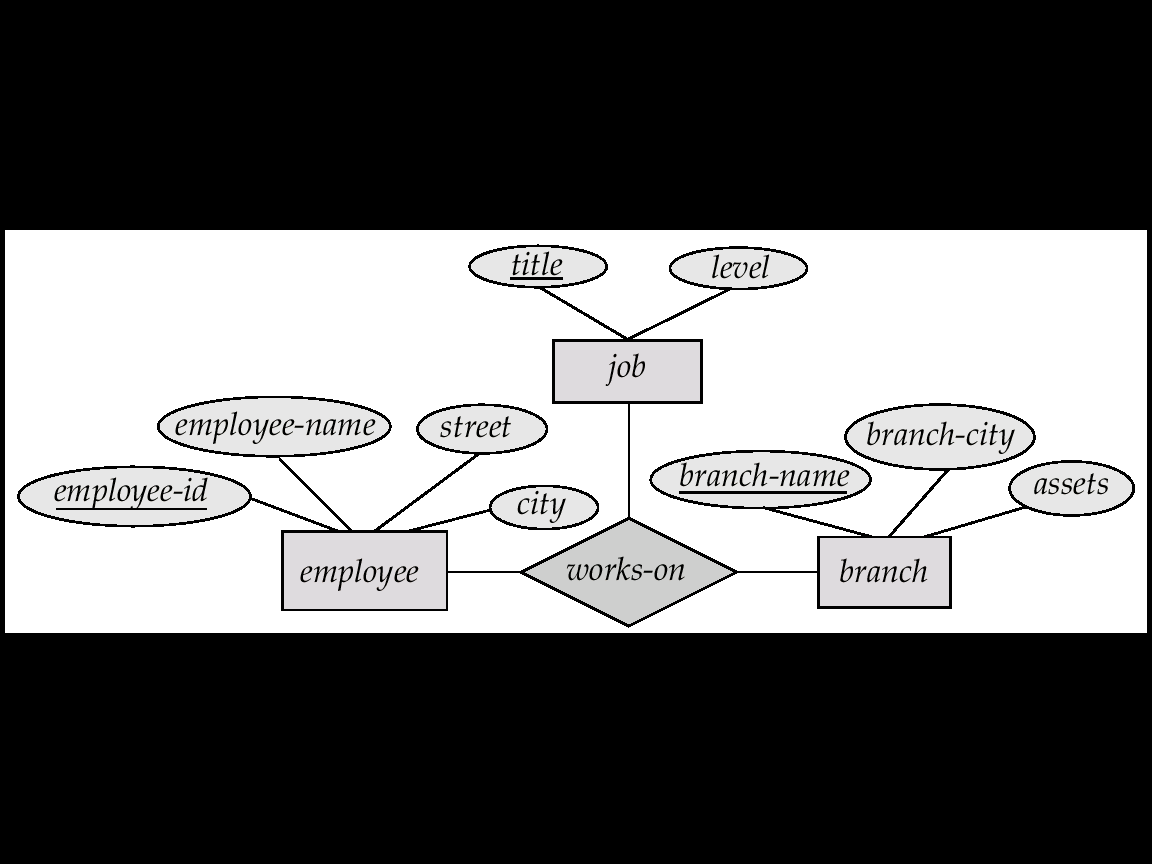
**One-to-one**: Each customer has one account and one account belongs to one customer only.

**

**Representation of Role:** Role of a given entity is indicated by labelling the lines that connect the entities and relationship set. Consider an entity Employee which stores information about all customers who work for some given factory. An employee may be a worker or manager. This role is represented as below



**Representation of Non-binary (ternary) relationship**:

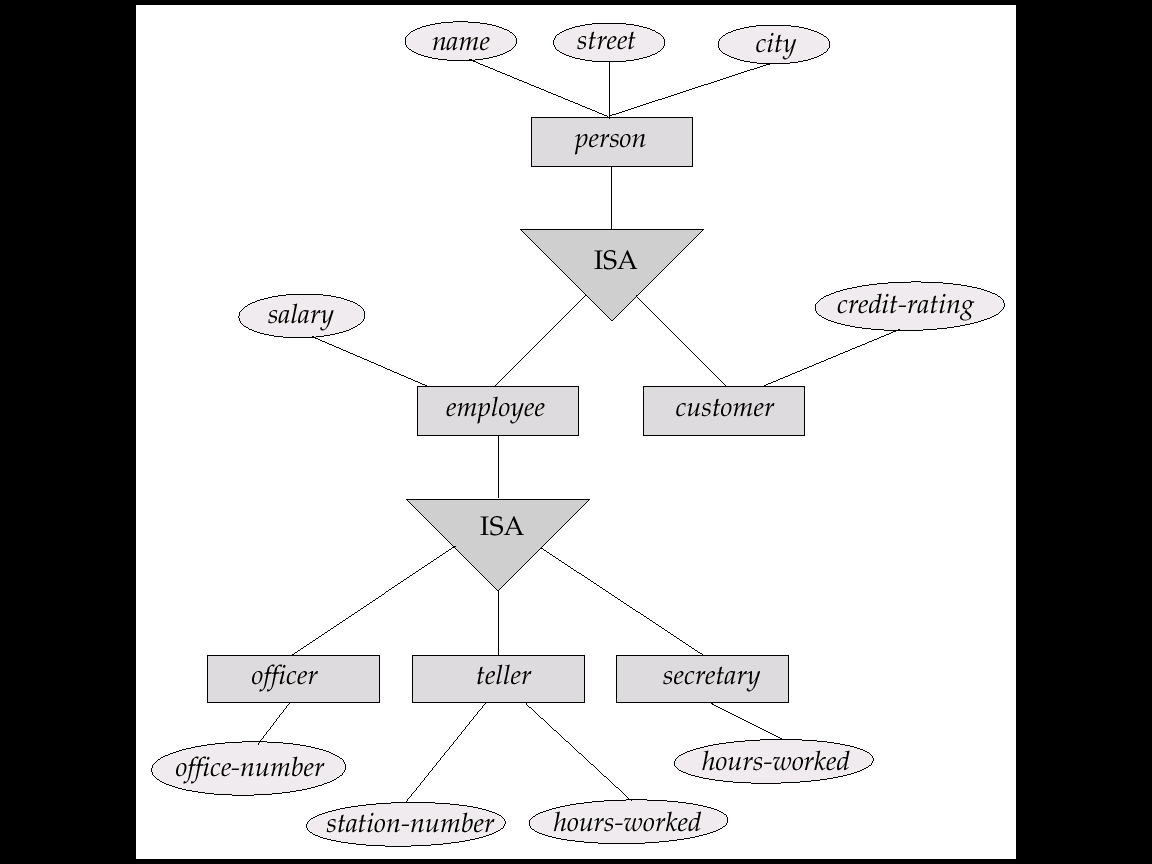


**Extended E-R Features:**

1. **Specialization**: This is Top-down design process. An entity set may include subgrouping of entities that are distinct in some way from other entities in the set. i.e. A subset of entities within an entity set may have attributes that are not shared by all the entities in the entity set.

e.g. an Entity set *person* with attributes *name, street and cit.* A person may be further classified as *customer* or *employee.* Each of these person types is described by a set of attributes that includes all the attributes of entity set person plus possibly additional attributes. E.g. , *customer carrying attributes like cust\_id, whereas employee (employee\_id, salary).*

***The process of designating subgrouping within an entity set is called as specialization. This is represented by triangle with text “IS A” in E-R Diagram.***

******

1. **Attribute Inheritance:** a lower-level entity set inherits all the attributes and relationship participation of the higher-level entity set to which it is linked.
2. **Generalization:** A bottom-up design process – combine a number of entity sets that share the same features into a higher-level entity set. *It uses the concept of superclass and subclass. In the above example person entity is superclass while customer and employee are subclass.*

Specialization and generalization are simple inversions of each other; they are represented in an E-R diagram in the same way. The terms specialization and generalization are used interchangeably.

The ISA relationship also referred to as superclass - subclass relationship.

Specialization and Generalization Can have multiple specializations of an entity set based on different features.

E.g. *permanent-employee* vs. *temporary-employee*, in addition to *officer* vs. *secretary* vs. *teller*

Each particular employee would be

* + a member of one of *permanent-employee* or *temporary-employee*,
  + and also a member of one of *officer*, *secretary*, or *teller*

**Design Constraints on a Specialization/ Generalization**

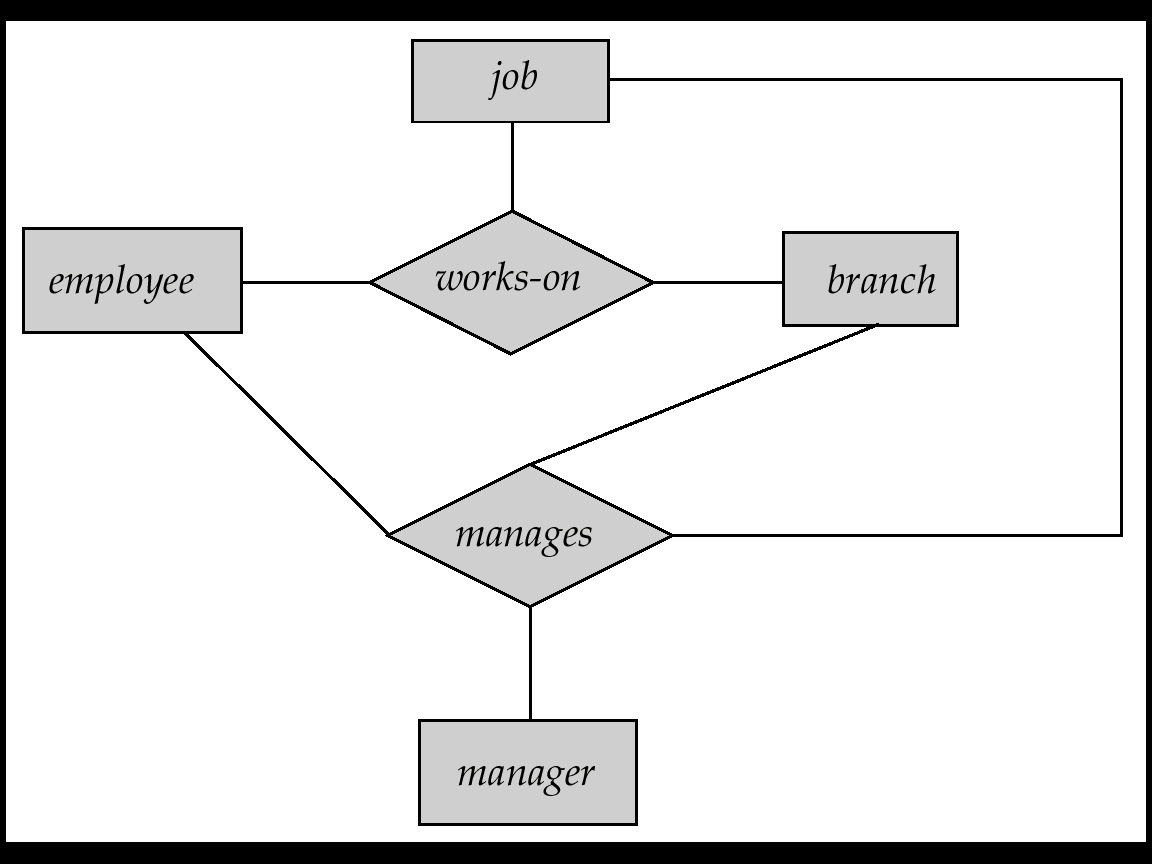
Constraint on which entities can be members of a given lower-level entity set.

* + **condition-defined**

E.g. all customers over 65 years are members of *senior-citizen* entity set; *senior-citizen* ISA *person*.

* + **user-defined** Constraint on whether or not entities may belong to more than one lower-level entity set within a single generalization.
  + **Disjoint** an entity can belong to only one lower-level entity set.Noted in E-R diagram by writing *disjoint* next to the ISA triangle
  + **Overlapping** an entity can belong to more than one lower-level entity set
  + **Completeness constraint** specifies whether or not an entity in the higher-level entity set must belong to at least one of the lower-level entity sets within a generalization.
    - **total :** an entity must belong to one of the lower-level entity sets
    - **partial:** an entity need not belong to one of the lower-level entity sets

1. **Aggregation :**

****

Consider the ternary relationship *works-on*, which we saw earlier, Suppose we want to record managers for tasks performed by an employee at a branch Aggregation is an abstraction through which relationships are treated as higher-level entities.

**Relationship sets *works-on* and *manages* represent overlapping information**

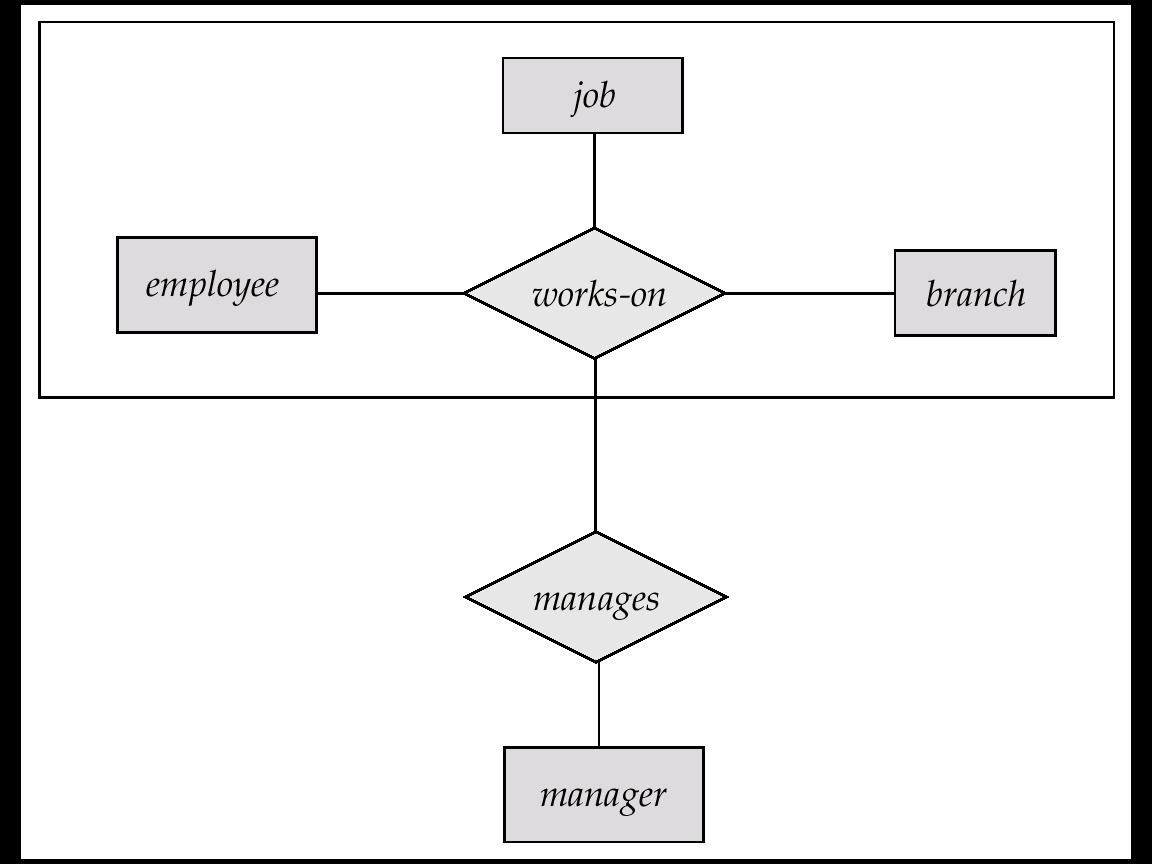
* + Every *manages* relationship corresponds to a *works-on* relationship
  + However, some *works-on* relationships may not correspond to any *manages* relationships So we can’t discard the *works-on* relationship

**Eliminate this redundancy via *aggregation***

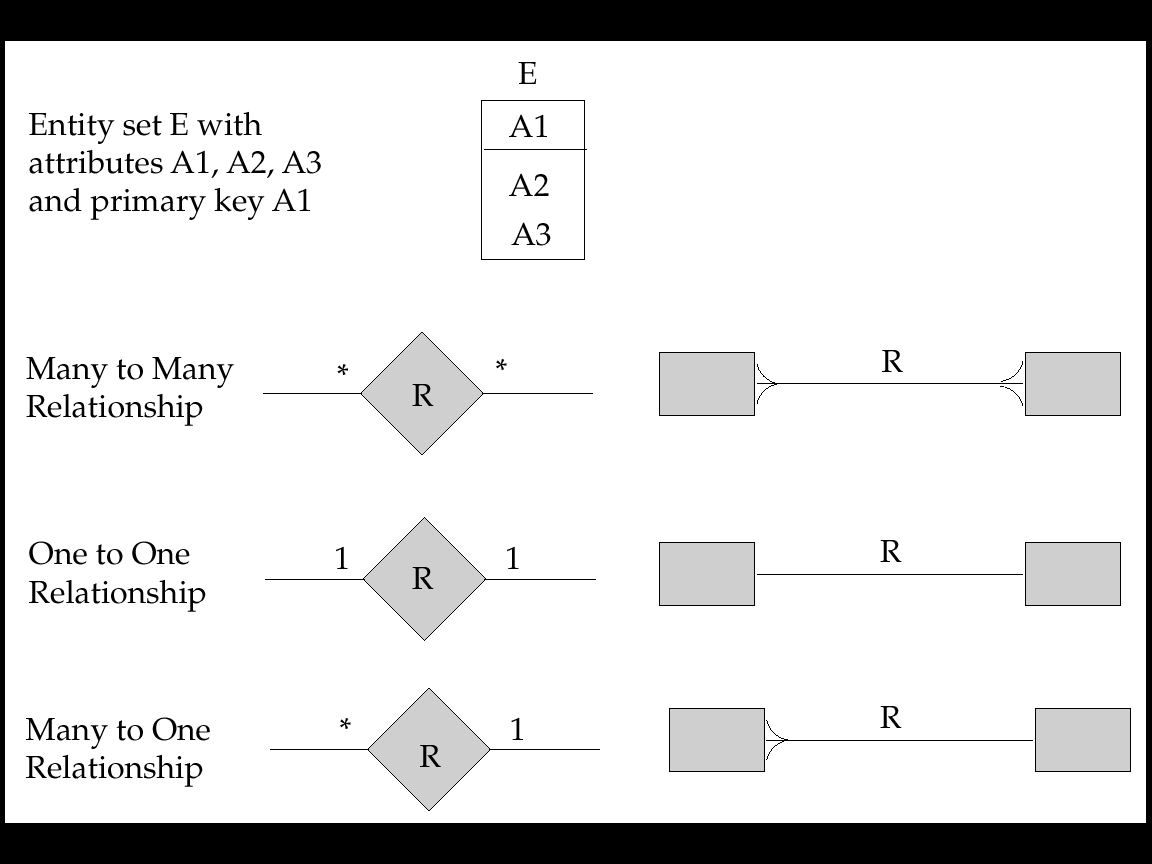
* + *Treat relationship as an abstract entity*
  + *Allows relationships between relationships*
  + *Abstraction of relationship into new entity*

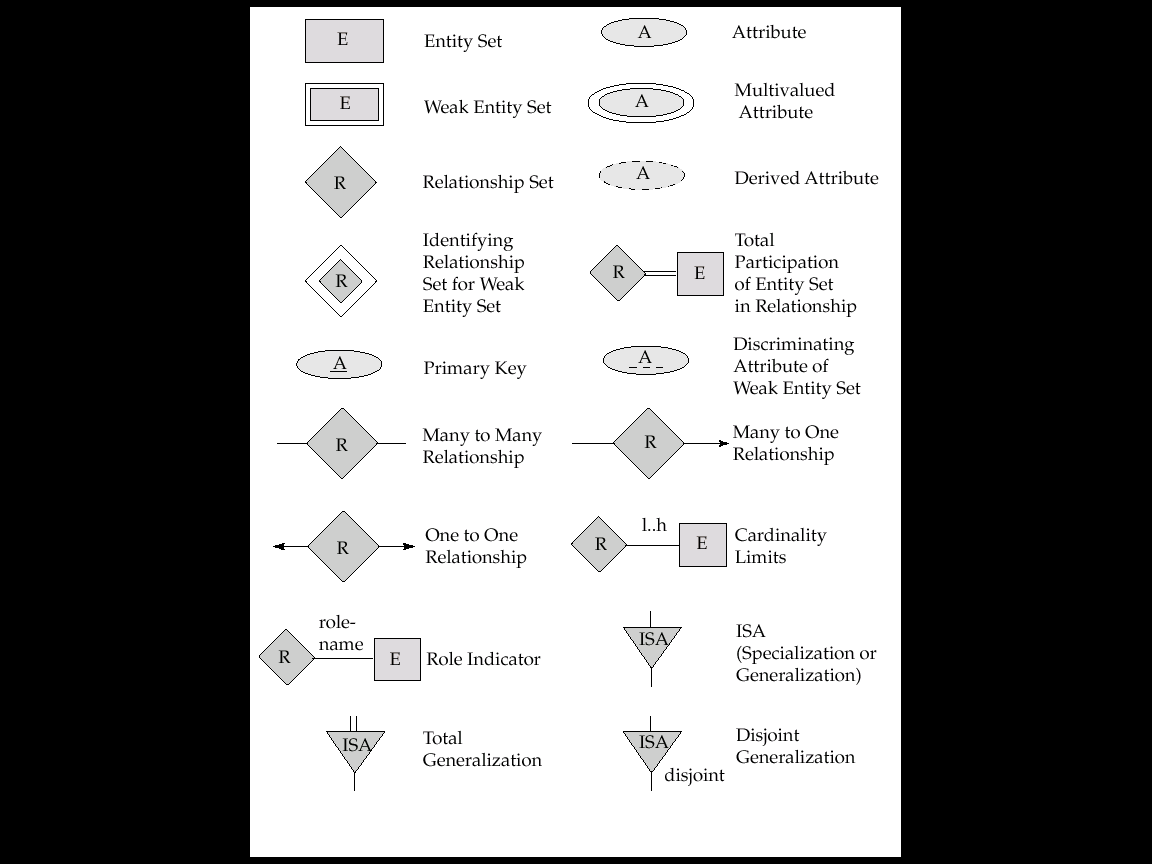
**Without introducing redundancy, the following diagram represents:**

* + An employee works on a particular job at a particular branch
  + An employee, branch, job combination may have an associated manager



**Summary of Symbols Used in E-R Notation**

****

**Reduction of an E-R Schema to Tables:**

* Primary keys allow entity sets and relationship sets to be expressed uniformly as *tables* which represent the contents of the database.
* A database which conforms to an E-R diagram can be represented by a collection of tables.
* For each entity set and relationship set there is a unique table which is assigned the name of the corresponding entity set or relationship set.
* Each table has a number of columns (generally corresponding to attributes), which have unique names.
* Converting an E-R diagram to a table format is the basis for deriving a relational database design from an E-R diagram.
* **A strong entity set reduces to a table with the same attributes.**
* **Composite attributes are flattened out by creating a separate attribute for each component attribute**

E.g. given entity set *custome*r with composite attribute *name* with component attributes *first-name* and *last-name* the table corresponding to the entity set has two attributes  
 *name.first-name* and *name.last-name*

* **A multivalued attribute M of an entity E is represented by a separate table EM**

Table EM has attributes corresponding to the primary key of E and an attribute corresponding to multivalued attribute M

**E.g. Multivalued attribute *dependent-names* of *employee* is represented by a table *employee-dependent-names*( *employee-id, dname*)**

Each value of the multivalued attribute maps to a separate row of the table EM.

E.g.,an employee entity with primary key John and dependents Johnson and Johndotir maps to **two rows:** (John, Johnson) and (John, Johndotir)

* **Representing Weak Entity Sets :** A weak entity set becomes a table that includes a column for the **primary key of the identifying strong** entity set.

**Reduction of E-R Schema to Tables:**

E-R database schema can be a collection of tables. For each entity set and for each relationship set, there is a unique table. Each table has multiple columns; each of which corresponds to attribute of that set.

1. **Representation of strong entity set:** Let ‘A’ is a strong entity set with attributes a1,a2,…an. ‘A’ entity is represented by a table with n- columns, each of which corresponds to attribute ‘A’. Each row in this table corresponds to one entity of the entity set ‘A’.
2. **Representation of Weak entity set:**  Let ‘A’ is weak entity set with attributes a1,a2…an. And depends on entity set ‘B’.Let ‘B’ is the strong entity set with attributes b1,b2…bn. ‘A’ can be represented by table with one column for each attribute of the set {a1,a2…an}U{b1,b2…bn}
3. **Representation of Relationship Set:** Let R be the relationship that relates the entity set A & B. A has attributes (a1,a2..an) and B has(b1,b2…bn). We represent Relationship set R with attributes with one column for each attribute of the set :{a1,a2..an}U{b1,b2..bn}

**e.g. relationship *borrower*** *The* ***customer with primary key cust\_id*** *and* ***loan with loan\_no.The relationship borrower with no attributes, the borrower table has two columns cust\_id ,loan\_no.***

**d. Representation of Generalization:**



**Method 1**: Create a table for the higher level entity set. For each lower-level entity set, create a table which includes a column for each attribute of the primary key of the higher-level entity set and one column for each attribute of that entity set. In this we have three tables

***Account(acc\_no,balance)***

***Saving\_account(acc\_no,interest\_rate)***

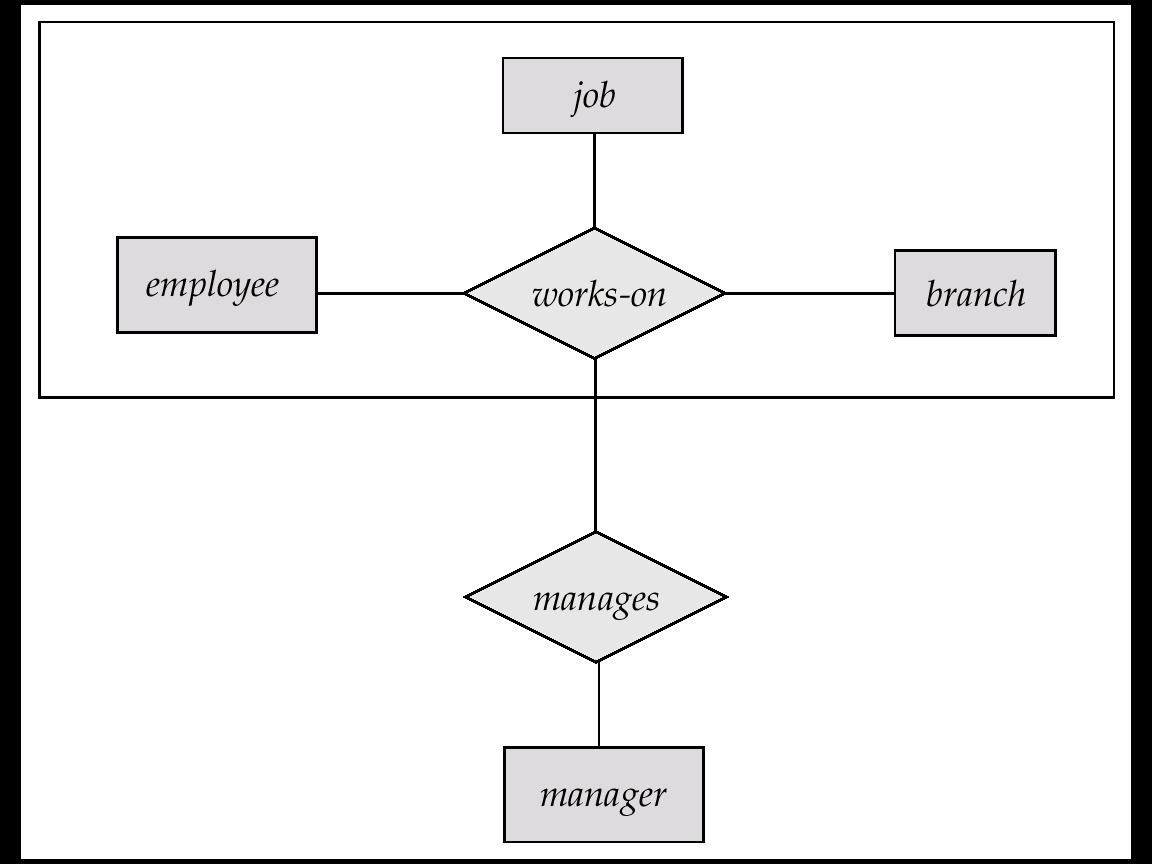
***Checking\_account(acc\_no,over\_draft\_amount)***

**Method 2**: Create table for each lower-level entity which includes columns for each of the attributes of that entity set plus column for each attribute of the higher-level entity set.

***Saving\_account(acc\_no,balance,interest\_rate)***

***Checking\_account(acc\_no,balance,over\_draft\_amount)***

1. **Representation of Aggregation :** create a schema containing primary key of the aggregated relationship, the primary key of the associated entity set any descriptive attributes.

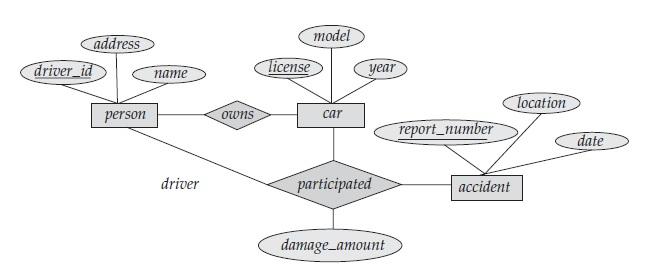
**`**

For example, to represent aggregation manages between relationship works\_on and entity set manager, create a schema ***manages* (*employee\_id, branch\_name, title, manager\_name*)**

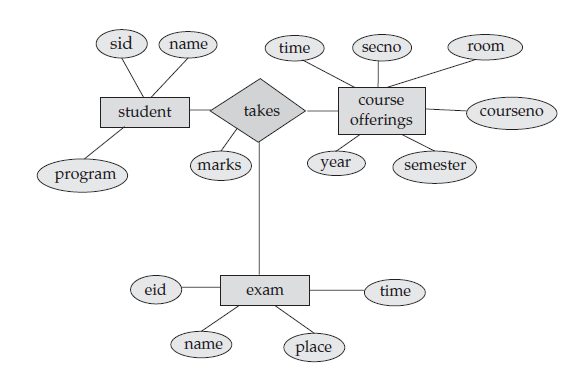
Schema *works\_on* is redundant provided we are willing to store null values for attribute *manager\_name* in relation on schema *manages*

**Questions:**

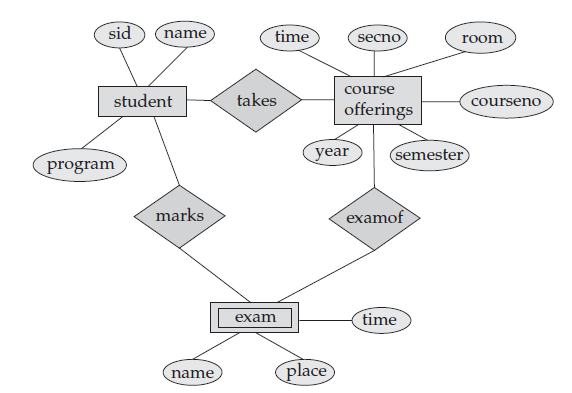
1. Construct an E-R dia. for car insurance company whose customers own one or more cars each. Each car is associated with it zero to any number of recorded accidents



1. Consider a database used to record the marks that students get in different exams of different course offerings.
   1. Construct an E-R diagram that models exams as entities, and uses a ternary relationship, for the above database.



* 1. Construct an alternative E-R dia. That uses only a binary relationship between students and course-offering. Make sure that only one relationship exists between a particular student and course-offering pair, yet you can represent the marks that a student gets in different exams of a course offering.



1. An university registrar’s office maintain data about the following entities
   * 1. courses, including number,title, credits,syllabus, and prerequisite ;
     2. Course offering,including course no,year,semester,section no,instructors timings,and classroom
     3. Student including student\_id,name and program
     4. Instructor including identification no,name,department and title. The enrollment of students in course and grades awarded to students in each course they are enrolled for must be appropriate modeled.

