Convert ER Model to Tables

By :
Dr. Rinkle Rani
Associate Professor, CSED
TIET, Patiala

Conversion of E-R Diagram to Tables

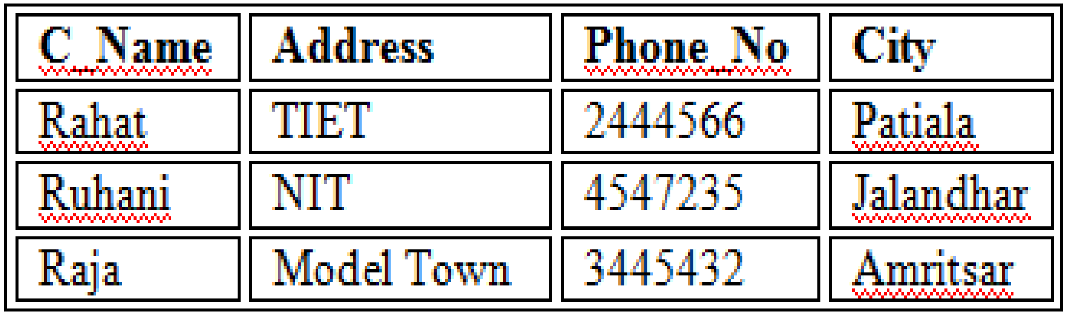
ÅFor each entity set and relationship set there is a unique table, which is assigned the name of the corresponding entity set or relationship set.

ÅEach table has a number of columns (generally corresponding to attributes), which have unique names.

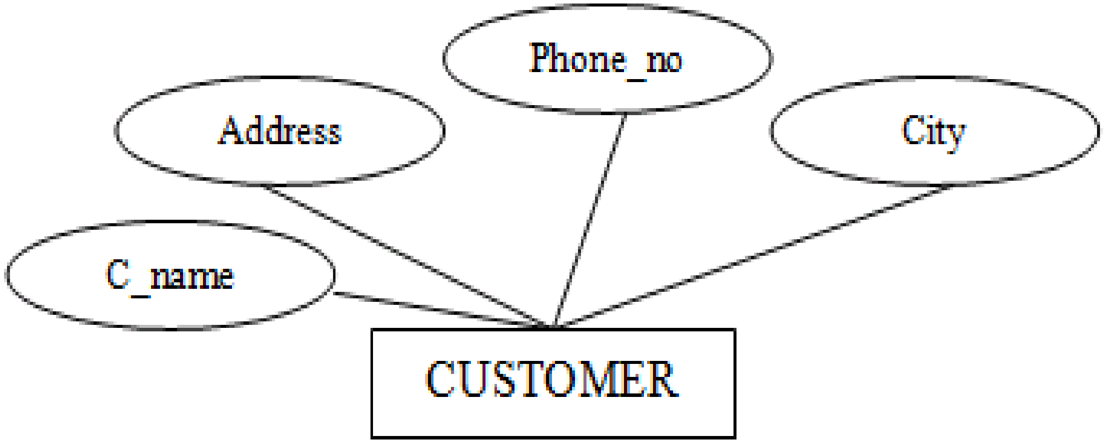
ÅPrimary keys allow entity sets and relationship sets to be expressed uniformly as tables, which represent the contents of the database.

Representing Entity sets as Tables

A strong entity set reduces to a table with the same attributes as represented in E-R diagram. For example a strong entity set Customer is represented as shown in figure can be simply converted to a table CUSTOMER with field names similar to attributes of an entity as shown in figure



C_Name	Address	Phone_No	City
Rahat	TIET	2444566	Patiala
Ruhani	NIT	4547235	Jalandhar
Raja	Model Town	3445432	Amritsar



Composite and Multi-Value Attributes

In order to convert an entity having composite attributes, the composite attributes are flattened out by creating a separate attribute for each component attribute.

For example: If in given entity set CUSTOMER Address attribute is composite attribute having House Number and locality as component attributes as shown in figure 4.15, the composite attribute address ids flattened to produce final table CUSTOMER with following fields:

C_Name, House Number, Locality, Phone Number and City.

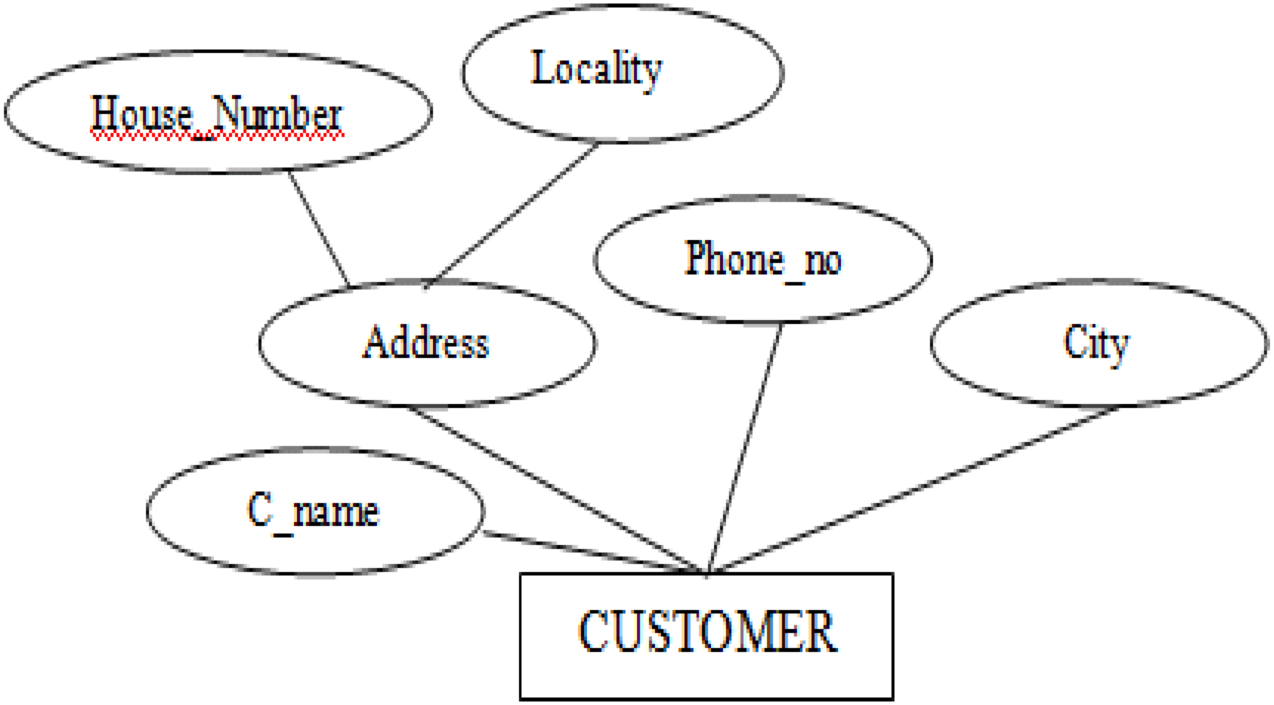
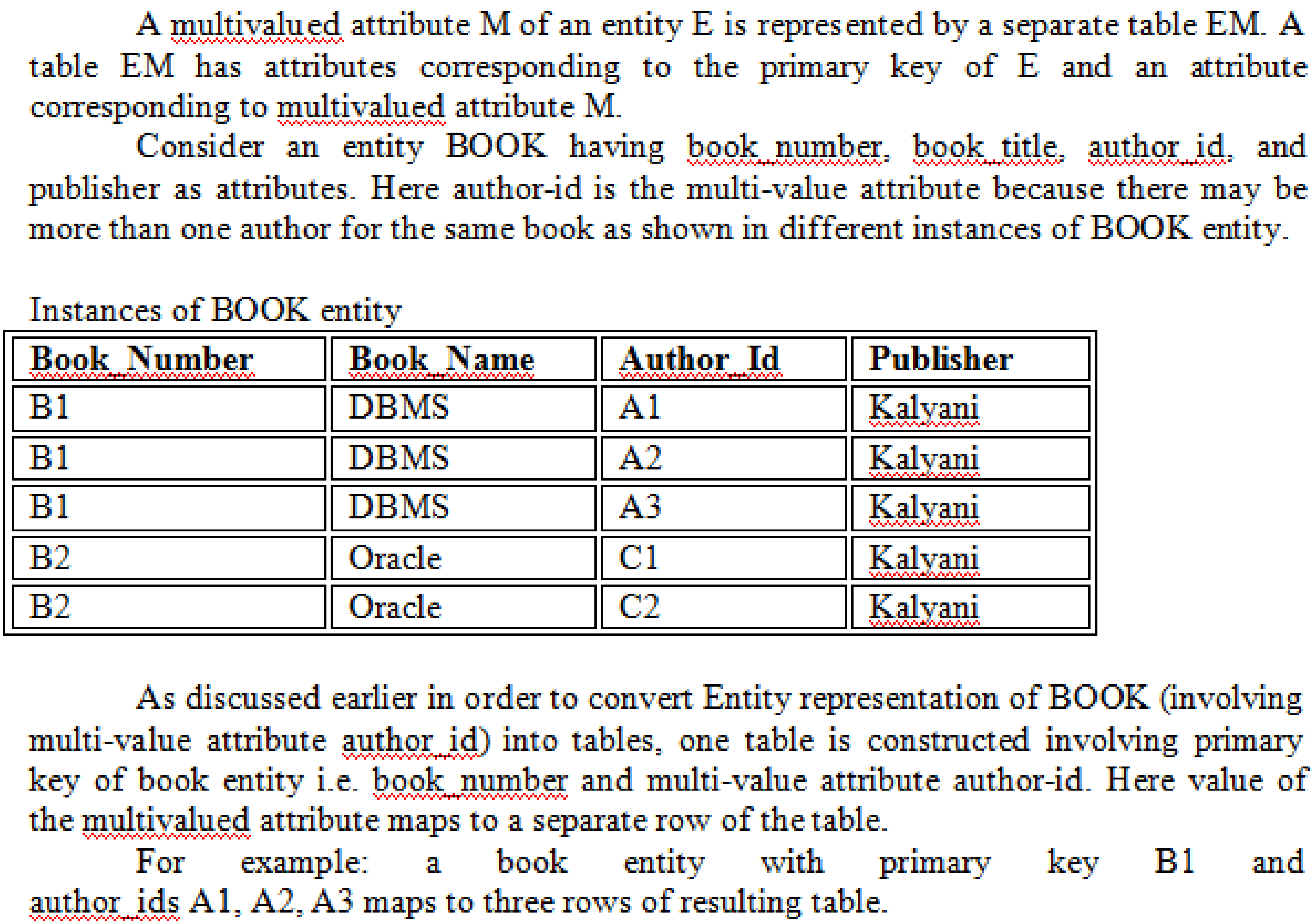


Figure 4.15 CUSTOMER entity with Address as composite attribute



A multivalued attribute M of an entity E is represented by a separate table EM. A table EM has attributes corresponding to the primary key of E and an attribute corresponding to multivalued attribute M.

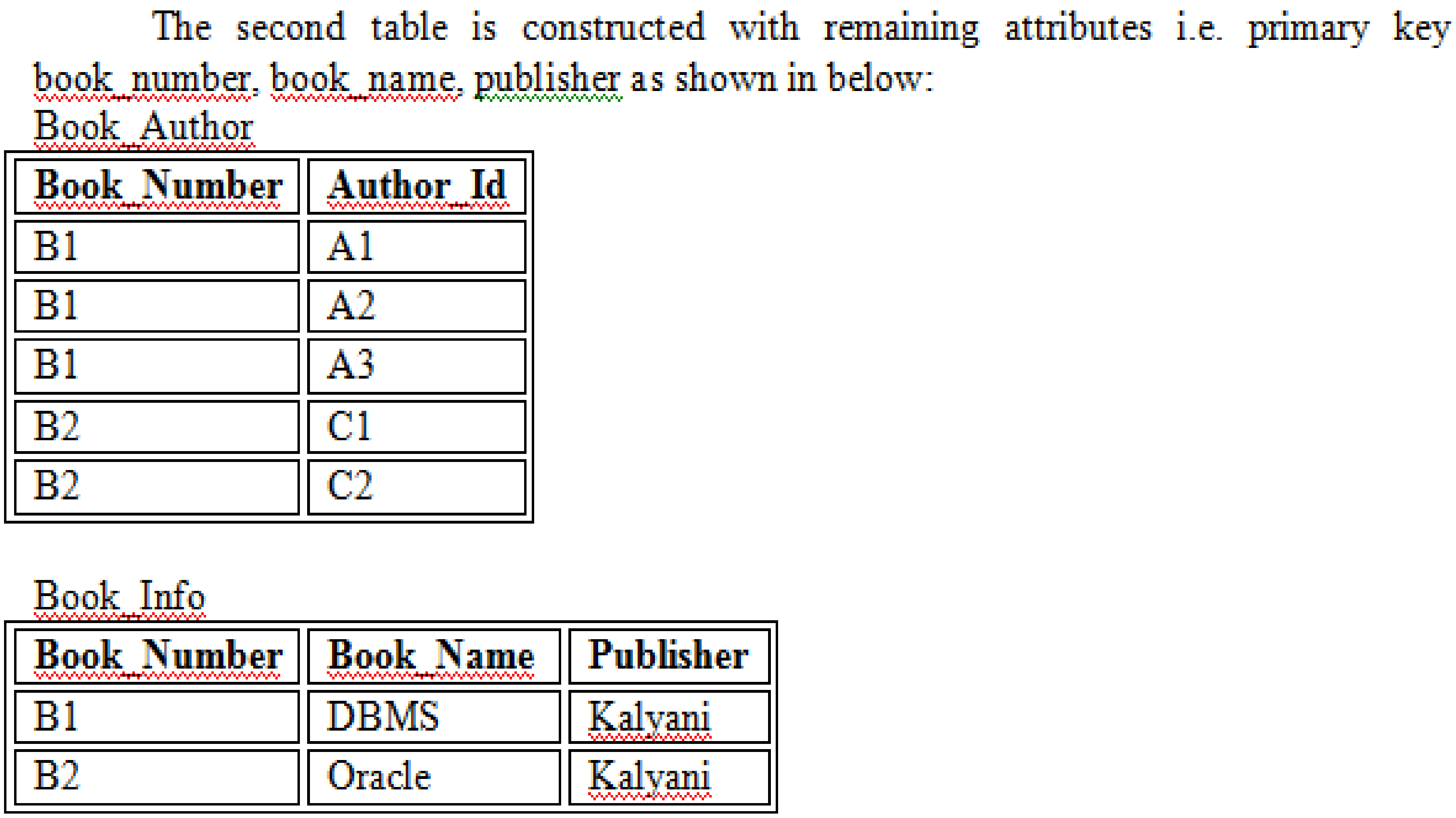
Consider an entity BOOK having book_number, book_title, author_id, and publisher as attributes. Here author-id is the multi-value attribute because there may be more than one author for the same book as shown in different instances of BOOK entity.

Instances of BOOK entity

Book_Number	**Book_Name**	**Author_Id**	**Publisher**
B1	DBMS	A1	Kalyani
B1	DBMS	A2	Kalyani
B1	DBMS	A3	Kalyani
B2	Oracle	C1	Kalyani
B2	Oracle	C2	Kalyani

As discussed earlier in order to convert Entity representation of BOOK (involving multi-value attribute author_id) into tables, one table is constructed involving primary key of book entity i.e. book_number and multi-value attribute author-id. Here value of the multivalued attribute maps to a separate row of the table.

For example: a book entity with primary key B1 and author_ids A1, A2, A3 maps to three rows of resulting table.



The second table is constructed with remaining attributes i.e. primary key book_number, book_name, publisher as shown in below:

Book_Author

Book_Number	Author_Id
B1	A1
B1	A2
B1	A3
B2	C1
B2	C2

Book_Info

Book_Number	Book_Name	Publisher
B1	DBMS	Kalyani
B2	Oracle	Kalyani

Representing weak entity set as table

A weak entity set becomes a table that includes a column for the primary key of the identifying strong entity set.

For example:

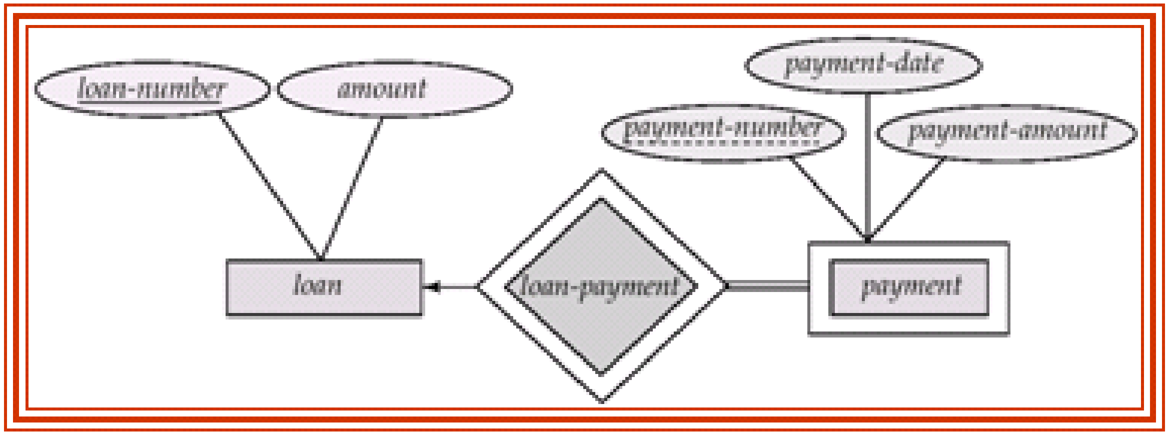
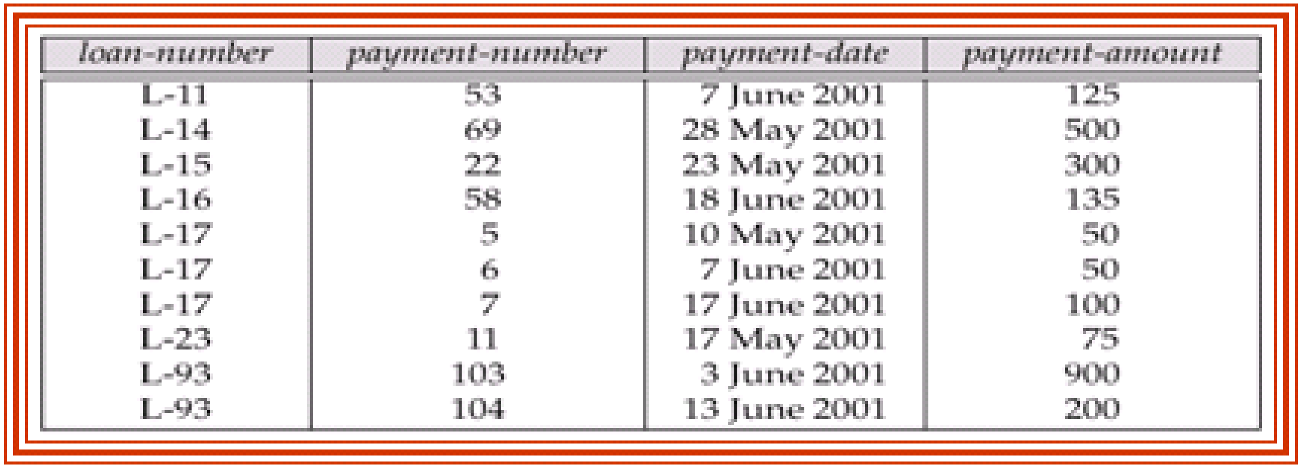


Figure 4.16: Weak entity set



loan-number	payment-number	payment-date	payment-amount
L-11	53	7 June 2001	125
L-14	69	28 May 2001	500
L-15	22	23 May 2001	300
L-16	58	18 June 2001	135
L-17	5	10 May 2001	50
L-17	6	7 June 2001	50
L-17	7	17 June 2001	100
L-23	11	17 May 2001	75
L-93	103	3 June 2001	900
L-93	104	13 June 2001	200

Representing Relations as Tables

many-many relationships are always mapped to a separate relation

A many-to-many relationship set is represented as a table with columns for the primary keys of the two participating entity sets, and any descriptive attributes of the relationship set.

For example: table for relationship set *borrower*

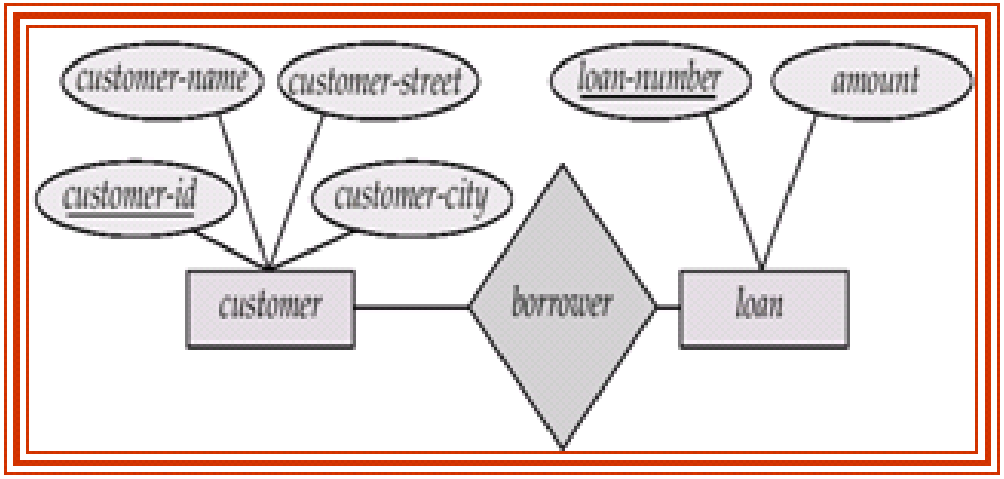
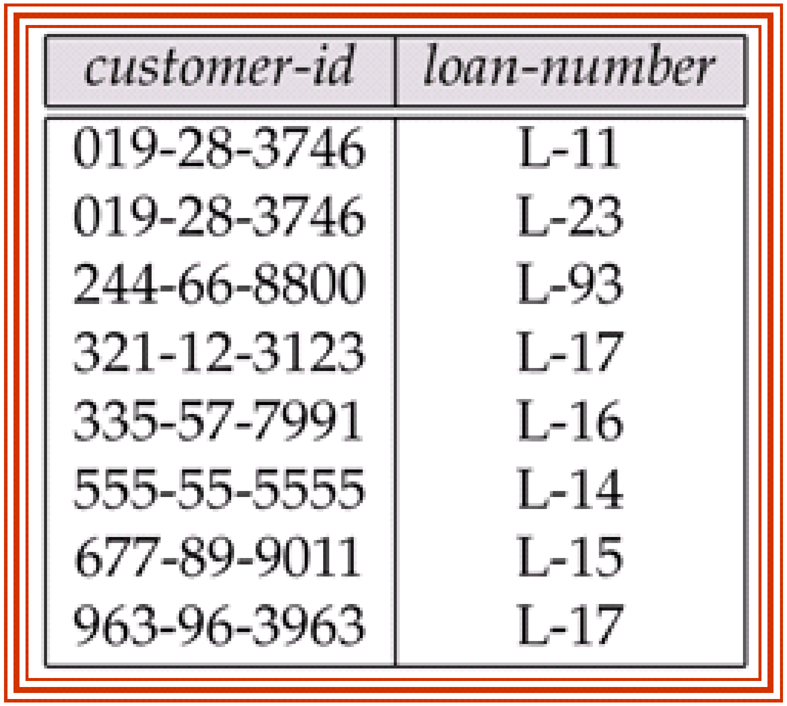


Figure 4.17: M: M relation



customer-id	*loan-number*
019-28-3746	L-11
019-28-3746	L-23
244-66-8800	L-93
321-12-3123	L-17
335-57-7991	L-16
555-55-5555	L-14
677-89-9011	L-15
963-96-3963	L-17

1-1 and 1-many relationships between separate entitites need not be mapped to a relation; the primary key attributes of the "1" relation become foreign key attributes of the "many" relation

For example: Instead of creating a table for relationship *account-branch*, add an attribute *branch* to the entity set *account*

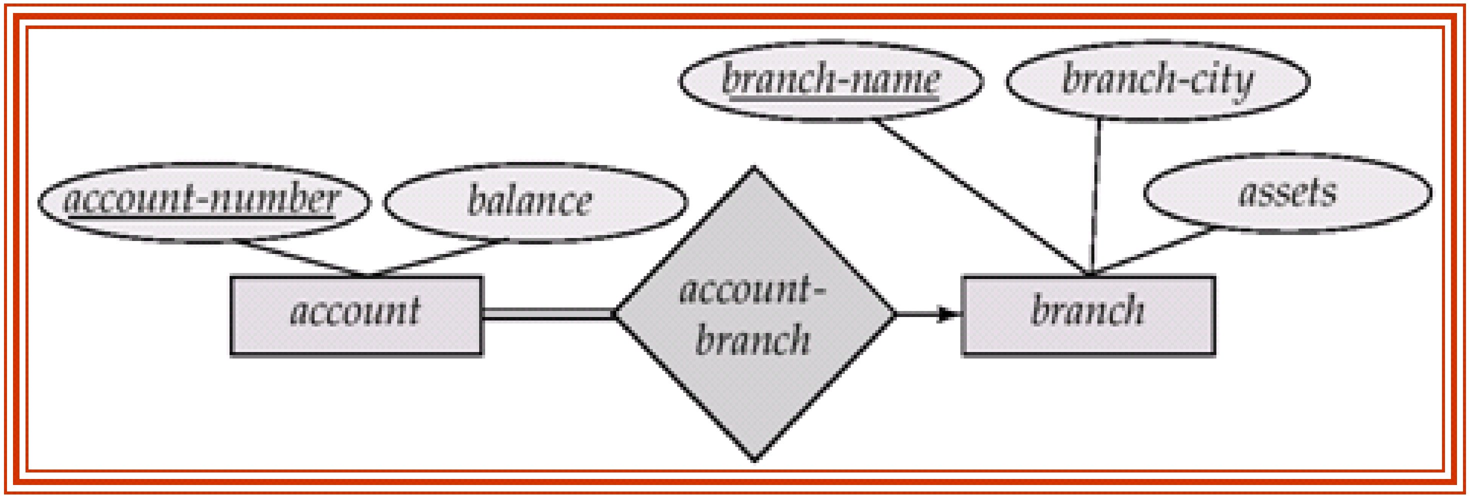
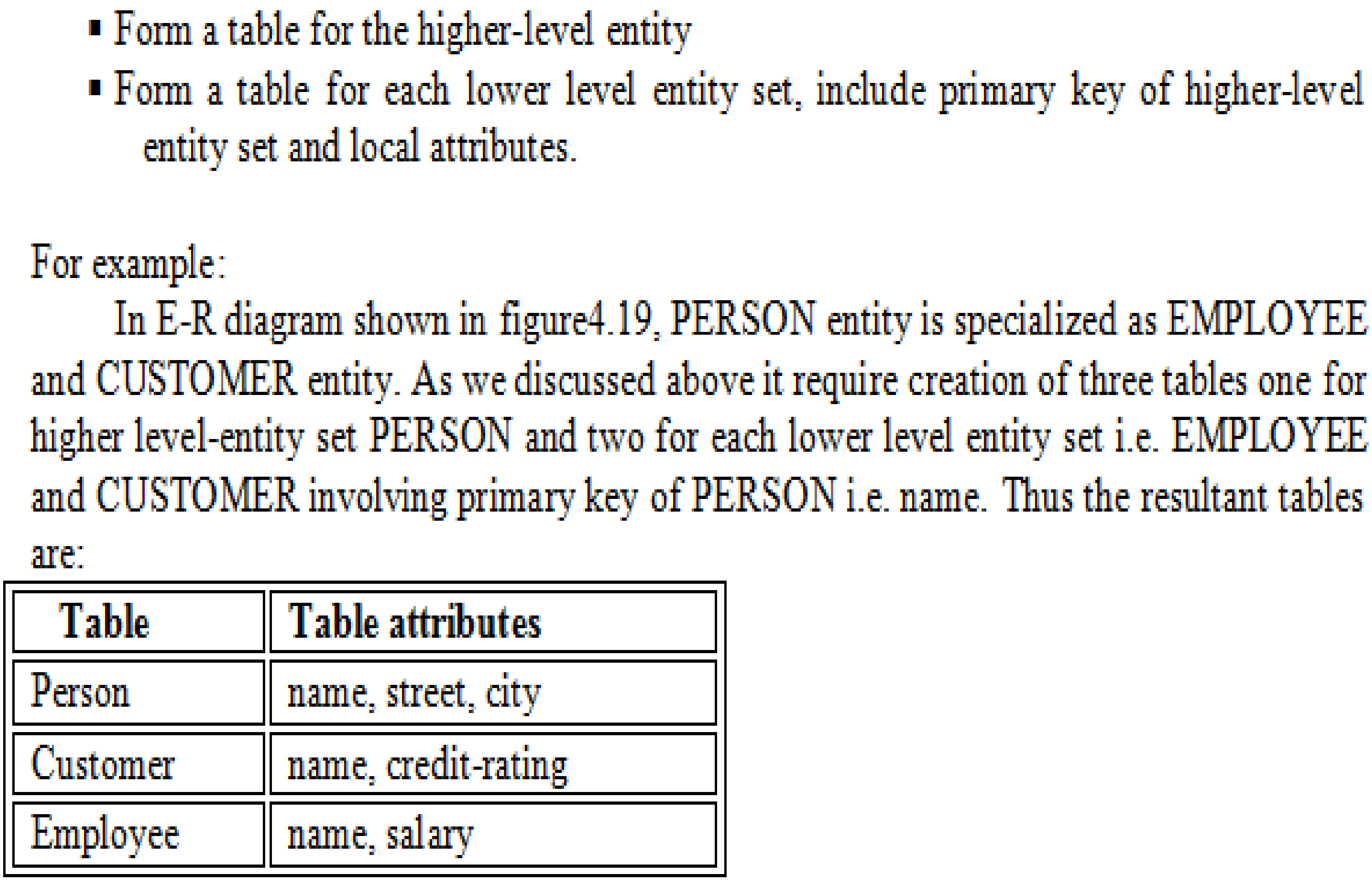


Figure 4.18: M: 1 relationship

Any attributes of the _1-1 or 1-many_ relationship may be attached to the "many" relation.

Representing specialization as tables :

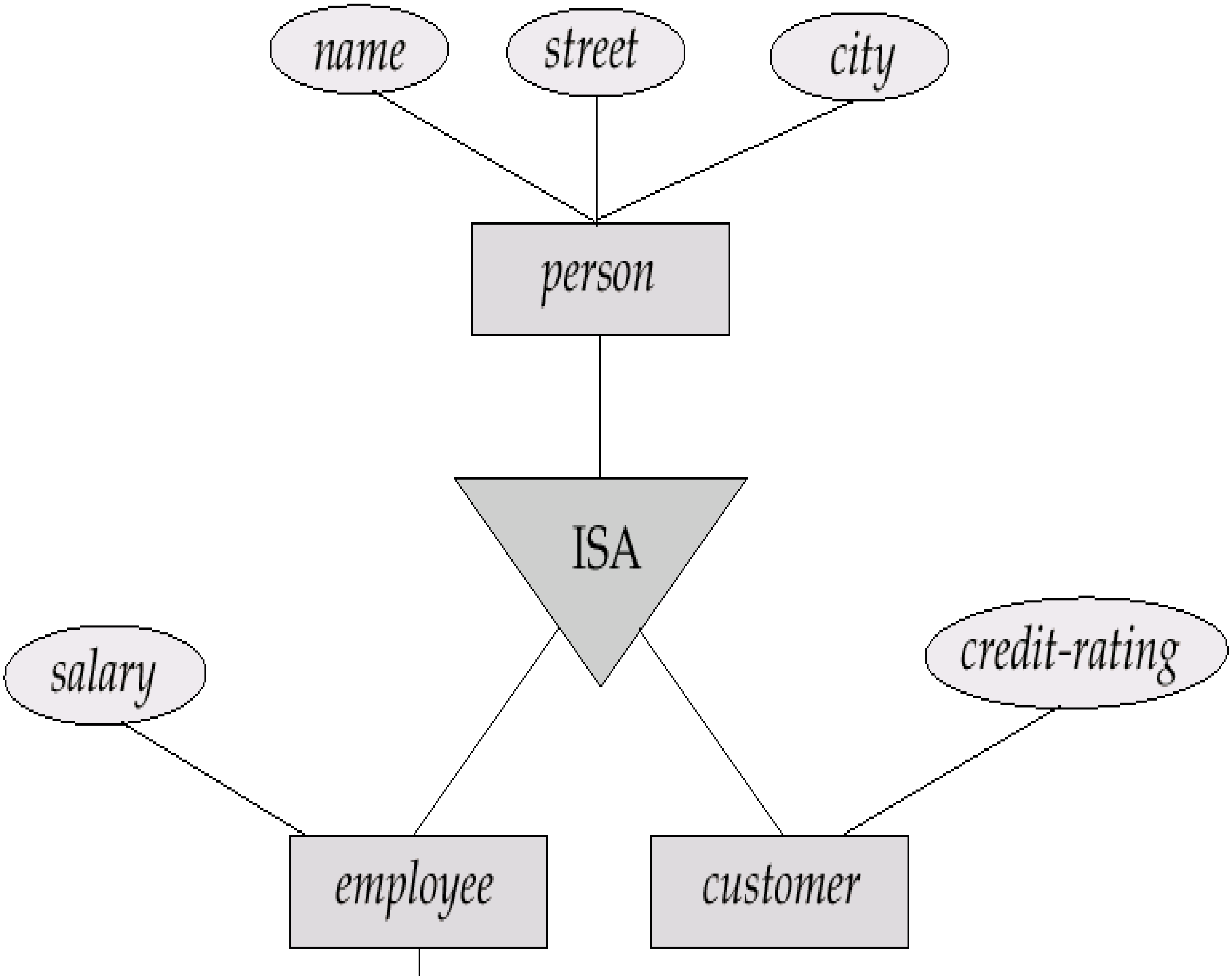


- Form a table for the higher-level entity
- Form a table for each lower level entity set, include primary key of higher-level entity set and local attributes.

For example:

In E-R diagram shown in figure4.19, PERSON entity is specialized as EMPLOYEE and CUSTOMER entity. As we discussed above it require creation of three tables one for higher level-entity set PERSON and two for each lower level entity set i.e. EMPLOYEE and CUSTOMER involving primary key of PERSON i.e. name. Thus the resultant tables are:

Table	**Table attributes**
Person	name, street, city
Customer	name, credit-rating
Employee	name, salary


name
street
city
person
ISA
salary
credit-rating
employee
customer

Representing Generalization as Tables

Form a schema for each entity set with all local and inherited attributes

schema	attributes
person	*name, street, city*
customer	*name, street, city, credit_rating*
employee	*name, street, city, salary*

ï If specialization is total, the schema for the generalized entity set (*person) not required to store information*

Drawback: *street and city may be stored redundantly for people* who are both customers and employees

Schemas Corresponding to Aggregation

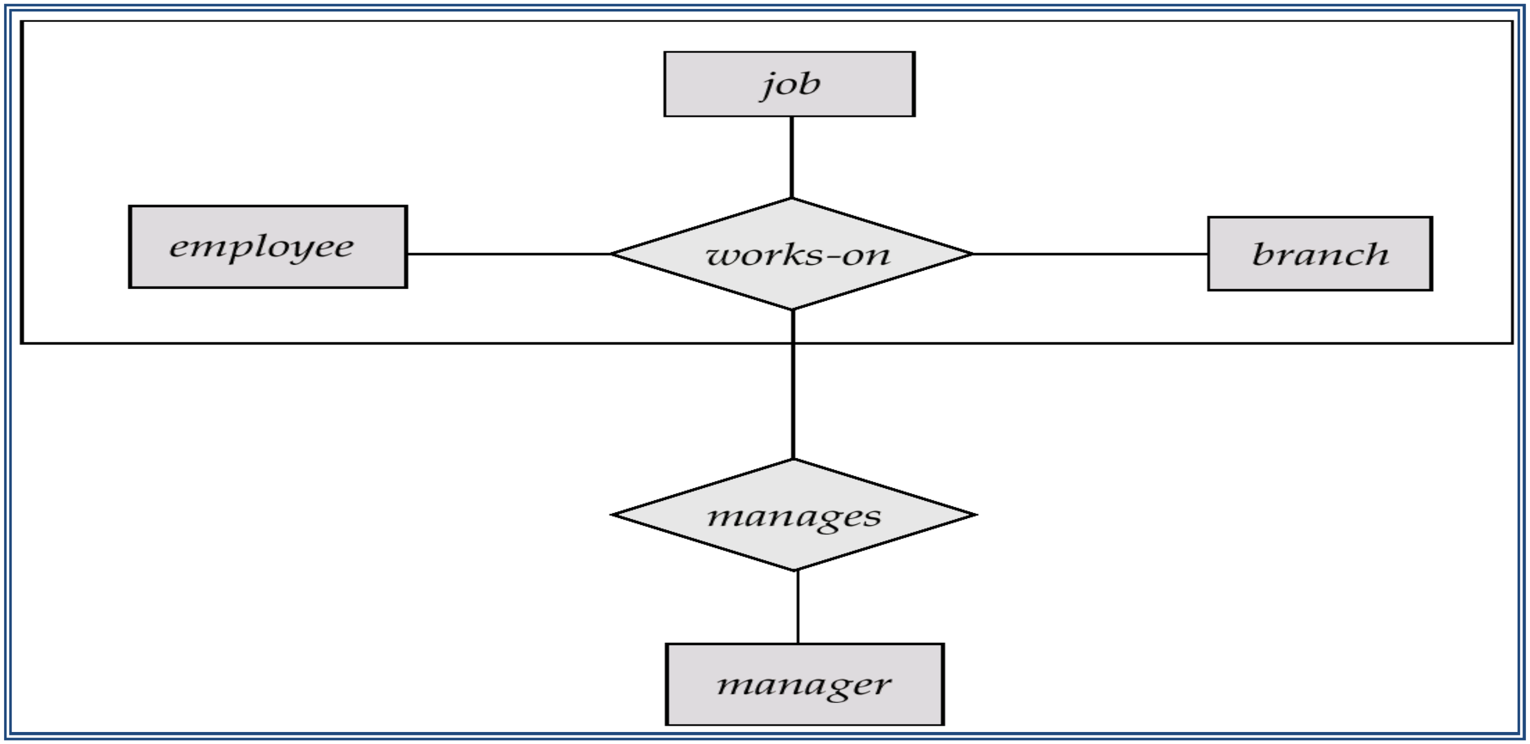
- To represent aggregation, create a schema containing
  - primary key of the aggregated relationship,
  - the primary key of the associated entity set
  - any descriptive attributes

Schemas Corresponding to Aggregation

- 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*


job
employee
works-on
branch
manages
manager