

Database Development

Exam 70-461 Querying Microsoft SQL Server 2012

Database Development

Course Objectives

- Normalization
- ER Diagram
- DBMS
- SQL Commands
- Constraints
- Data integrity rules
- Subqueries
- Indexes
- Functions
- Triggers
- Stored Procedures

Database Development

Material:

Book:

**Training Kit (Exam 70-461) Querying
Microsoft SQL Server 2012**

Days Reading:

Database Development

Tools:

MS VISIO 2013

MS SQL SERVER 2012 or 2014 Evaluation Edition

Database Development

Evaluation:

Oral or MCQ Test

Database Development

Outline:

- Database Introduction
- Data Modeling
 - Entity Relationship modeling
 - Normalization

Database Development

General Terms

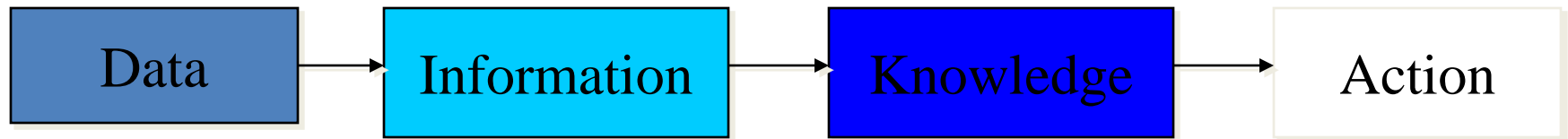
Data

Database

Database Management System

What is the ultimate purpose of a database management system?

Is to transform



Data Modeling

The three level of data modeling,

1. Conceptual data model
2. Logical data model
3. Physical data model

Abstraction Layers

- Conceptual
 - What data is held
 - An Image and its meta-data
 - Entity-Relationship model (ERM)
- Logical
 - How data is organised in storage
 - Block and Directory structure
 - Tables, keys
- Physical
 - How data is stored in bits
 - JPEG as a stream of bytes
 - A Database as files and records stored in a DBMS-specific format

WHAT

Realisation

(Refinement

Reification)

(Engineering,
Model-Driven
development

Abstraction

(Reverse
Engineering)

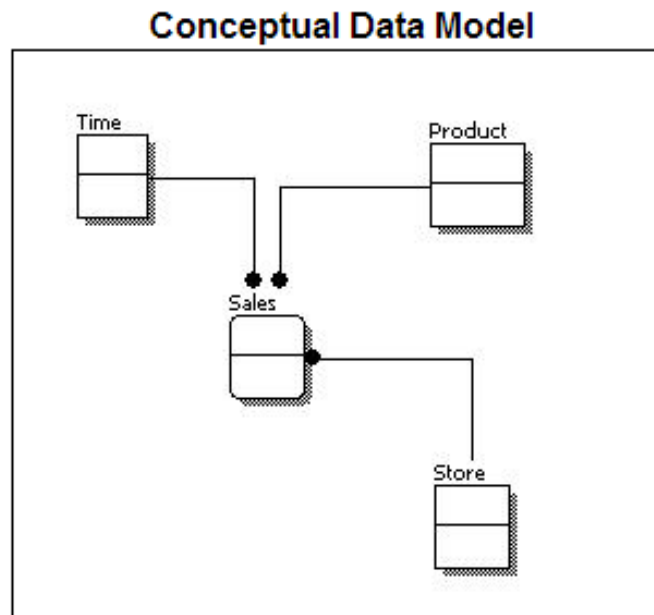
HOW

Conceptual data model

A conceptual data model identifies the highest-level relationships between the different entities. Features of conceptual data model include:

- Includes the important entities and the relationships among them.
- No attribute is specified.
- No primary key is specified.

The figure below is an example of a conceptual data model.



From the figure above, we can see that the only information shown via the conceptual data model is the entities that describe the data and the relationships between those entities. No other information is shown through the conceptual data model.

Logical data model

A logical data model describes the data in as much detail as possible, without regard to how they will be physical implemented in the database. Features of a logical data model include:

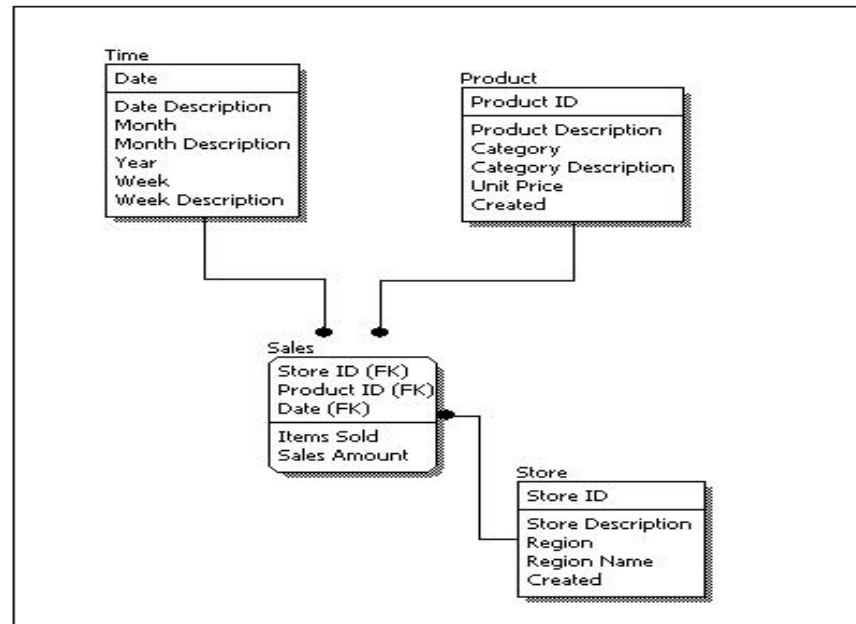
- Includes all entities and relationships among them.
- All attributes for each entity are specified.
- The primary key for each entity is specified.
- Foreign keys (keys identifying the relationship between different entities) are specified.
- Normalization occurs at this level.

The steps for designing the logical data model are as follows:

1. Specify primary keys for all entities.
2. Find the relationships between different entities.
3. Find all attributes for each entity.
4. Resolve many-to-many relationships.
5. Normalization.

The figure below is an example of a logical data model.

Logical Data Model



Logical data model

Comparing the logical data model shown above with the [conceptual data model](#) diagram, we see the main differences between the two:

- In a logical data model, primary keys are present, whereas in a conceptual data model, no primary key is present.
- In a logical data model, all attributes are specified within an entity. No attributes are specified in a conceptual data model.
- Relationships between entities are specified using primary keys and foreign keys in a logical data model. In a conceptual data model, the relationships are simply stated, not specified, so we simply know that two entities are related, but we do not specify what attributes are used for this relationship.

Physical data model

Physical data model represents how the model will be built in the database. A physical database model shows all table structures, including column name, column data type, column constraints, primary key, foreign key, and relationships between tables. Features of a physical data model include:

- Specification all tables and columns.
- Foreign keys are used to identify relationships between tables.
- Denormalization may occur based on user requirements.
- Physical considerations may cause the physical data model to be quite different from the logical data model.
- Physical data model will be different for different RDBMS. For example, data type for a column may be different between MySQL and SQL Server.

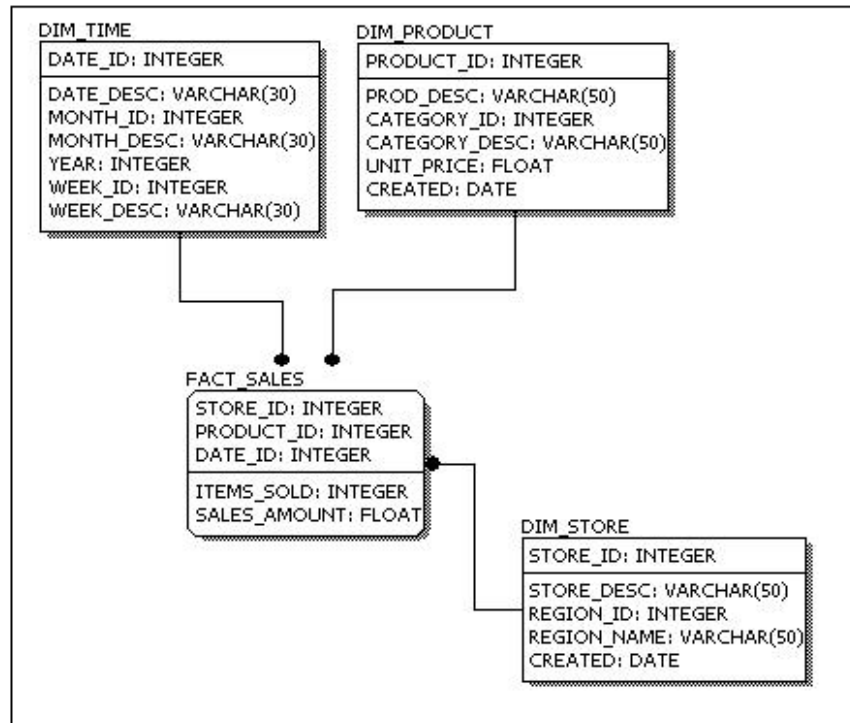
The steps for physical data model design are as follows:

1. Convert entities into tables.
2. Convert relationships into foreign keys.
3. Convert attributes into columns.
4. Modify the physical data model based on physical constraints / requirements.

Physical data model

The figure below is an example of a physical data model.

Physical Data Model



Comparing the logical data model shown above with the [logical data model](#) diagram, we see the main differences between the two:

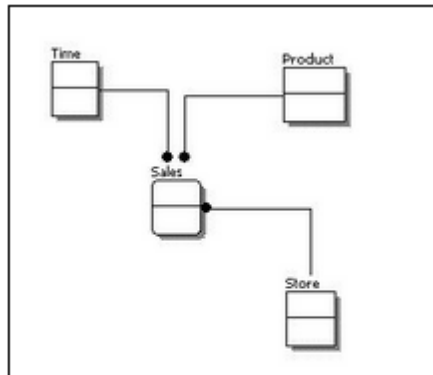
- Entity names are now table names.
- Attributes are now column names.
- Data type for each column is specified. Data types can be different depending on the actual database being used.

Comparison

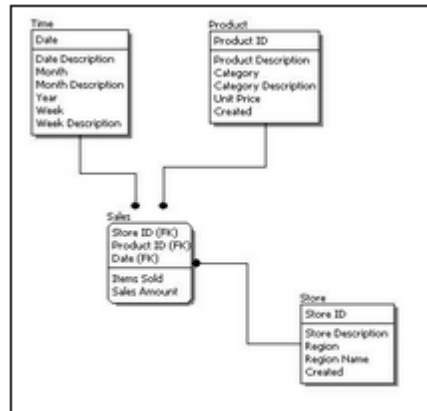
Feature	Conceptual	Logical	Physical
Entity Names	✓	✓	
Entity Relationships	✓	✓	
Attributes		✓	
Primary Keys		✓	✓
Foreign Keys		✓	✓
Table Names			✓
Column Names			✓
Column Data Types			✓

Below we show the conceptual, logical, and physical versions of a single data model.

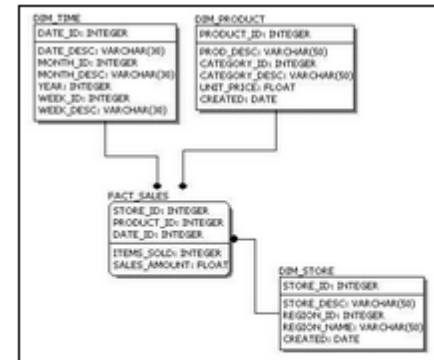
Conceptual Model Design



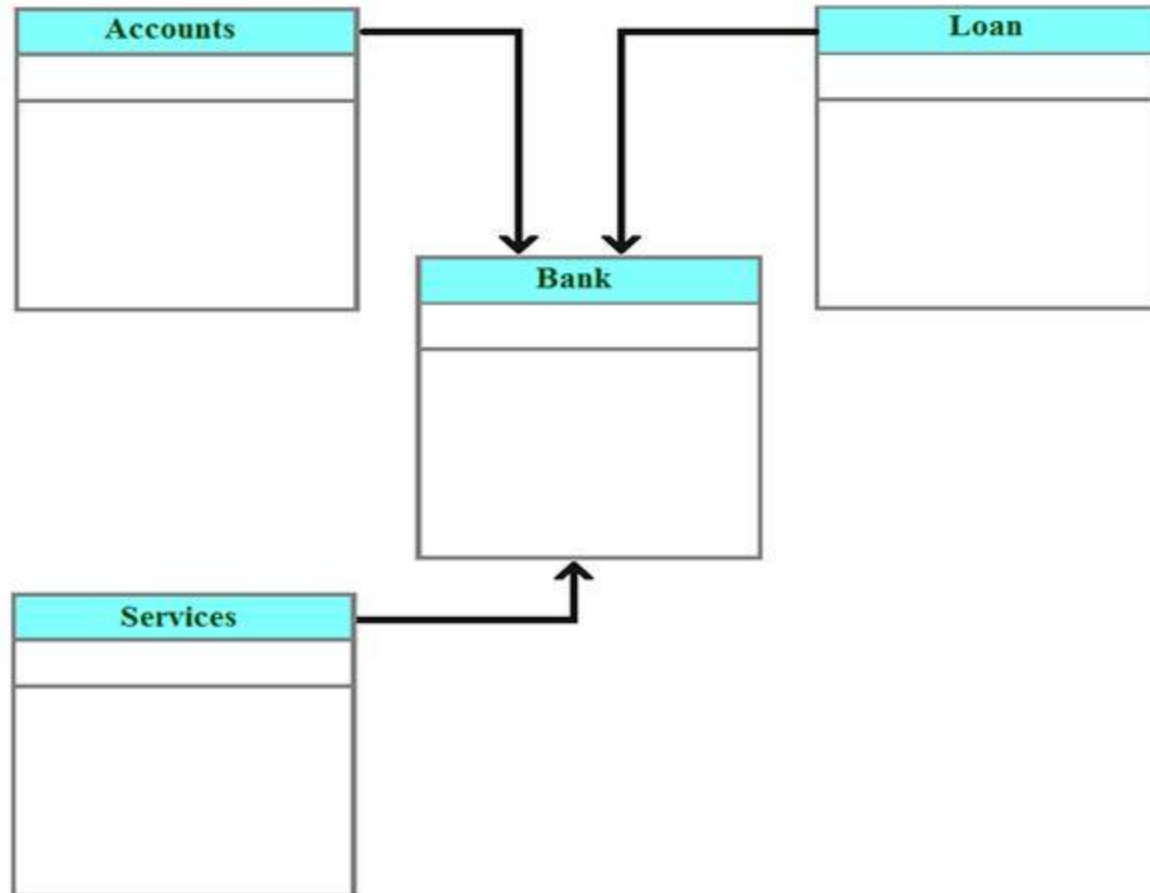
Logical Model Design



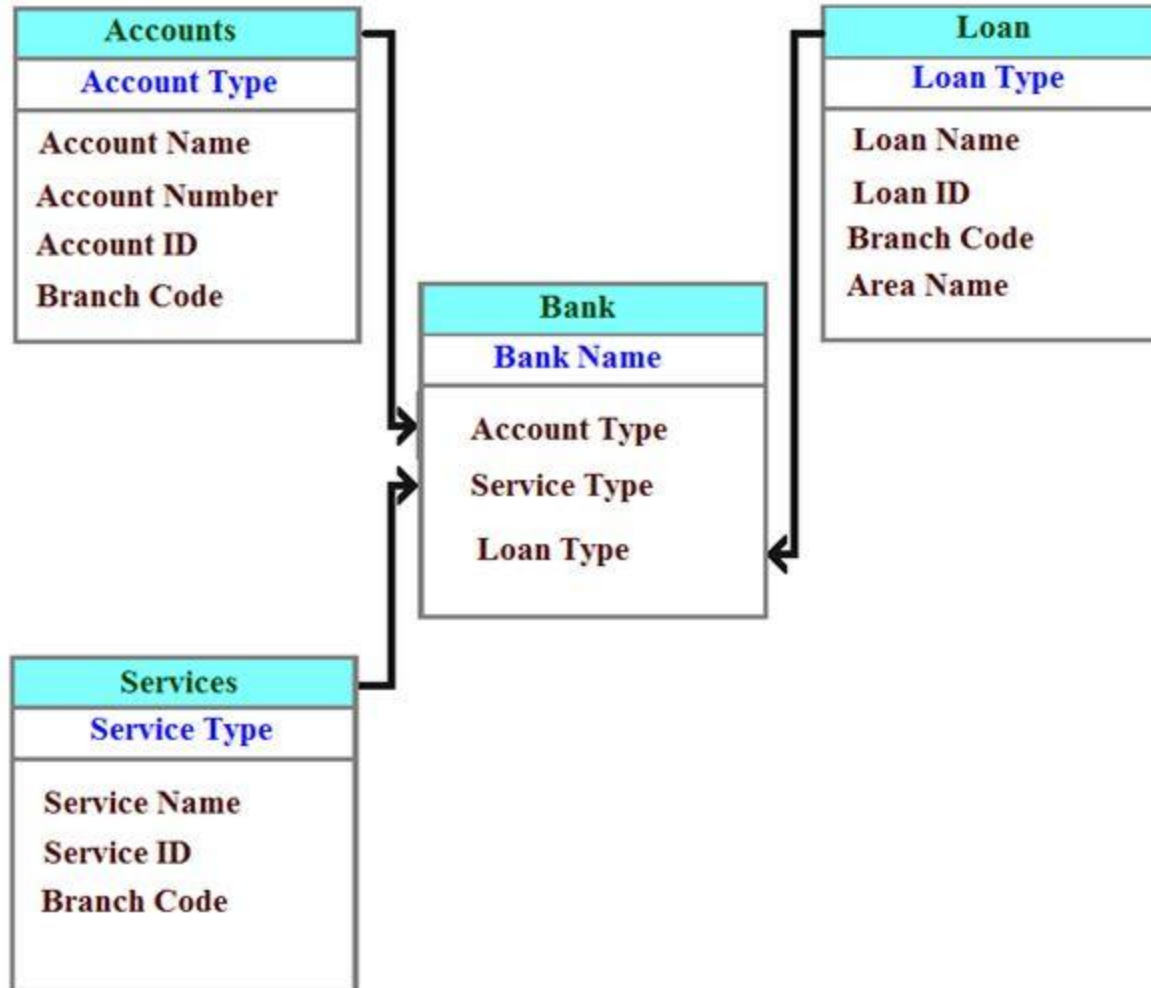
Physical Model Design



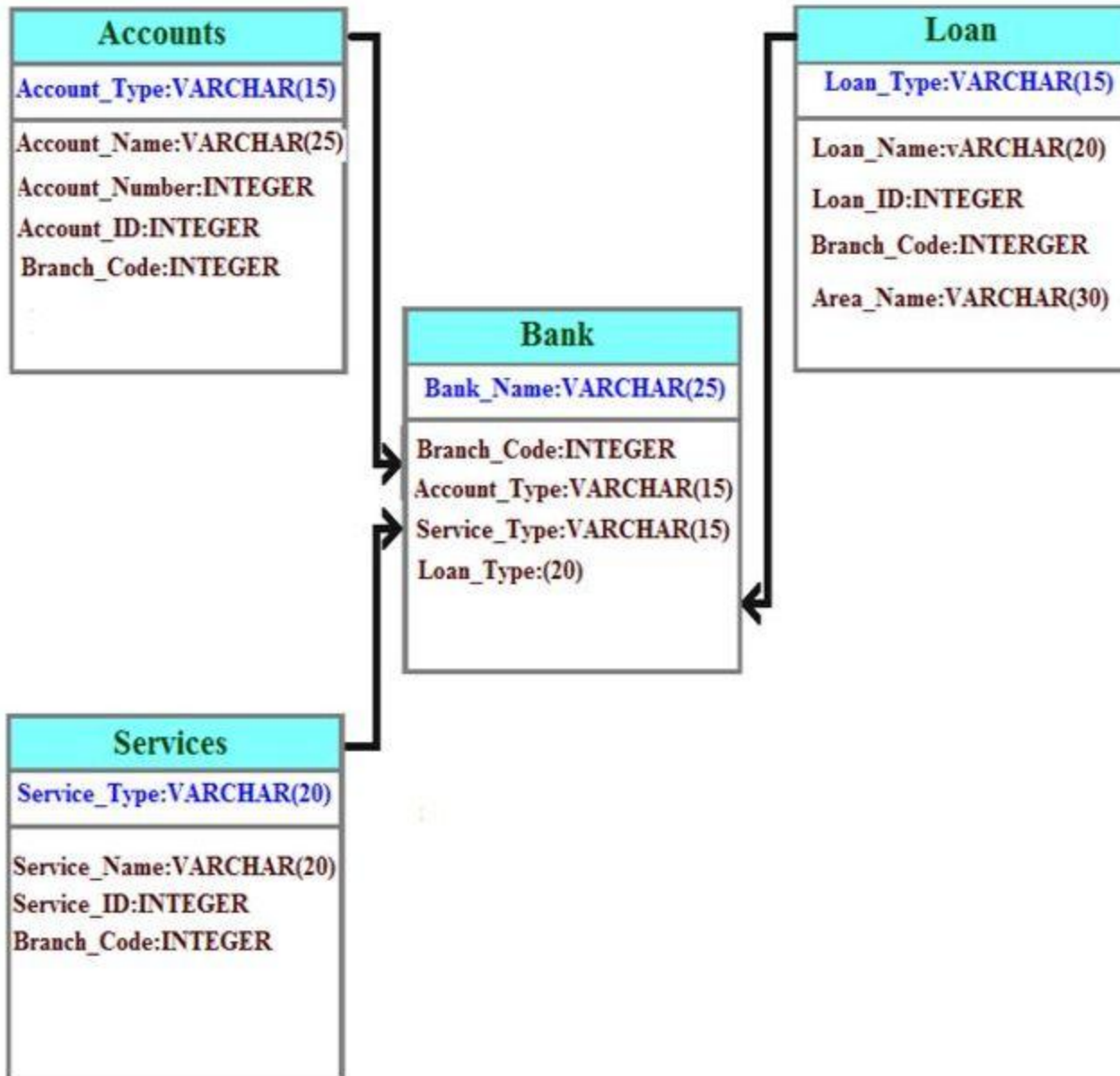
Conceptual Design



Logical Design



Physical Design



Comparison

Feature	Conceptual	Logical	Physical
Table Names	No	No	Yes
Column Names	No	No	Yes
Column Data Types	No	No	Yes
Entity Names	Yes	Yes	No
Entity Relationships	Yes	Yes	No
Attributes	No	Yes	No
Primary Keys	No	Yes	Yes
Foreign Keys	No	Yes	Yes

E R Diagram

Entity relationship Model

Specialized graphics

Inter-relationship between Entities

Use symbols

E R Diagram (Chen Notation)

Three types of information

Boxes = Entity

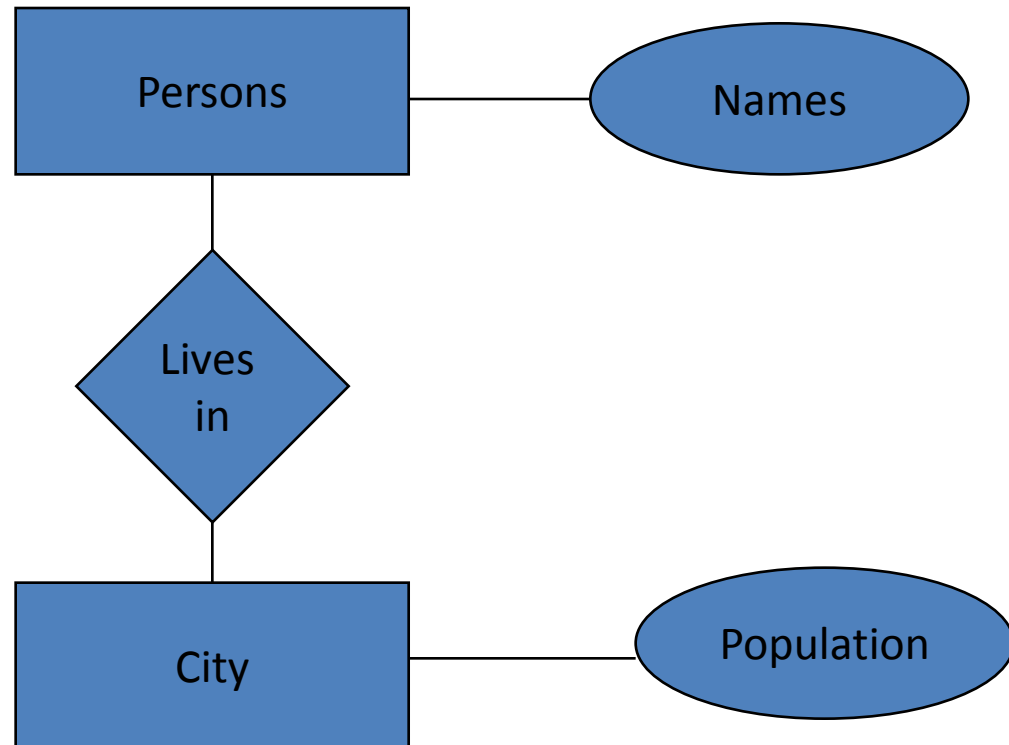
Diamonds = Relationship

Oval = Attributes

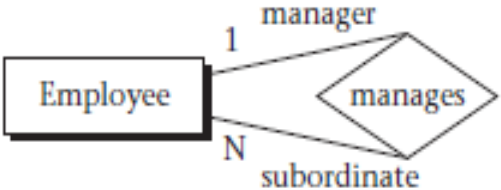
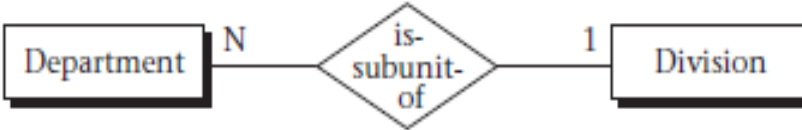
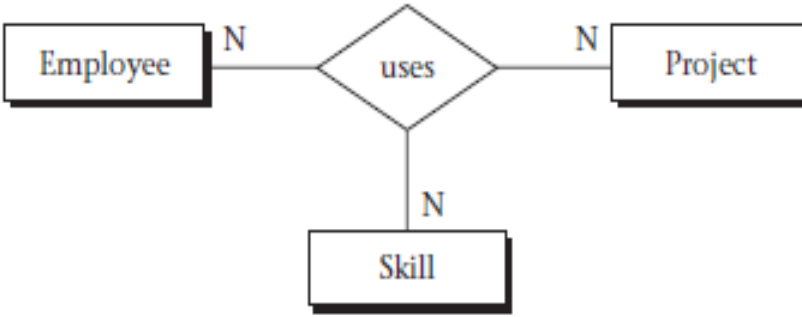
E R Diagram (Chen Notation)

Example:

Information about residents



E R Diagram

<i>Concept</i>	<i>Representation & Example</i>
<p><i>Degree</i></p> <p>recursive binary</p>	
<p>binary</p>	
<p>ternary</p>	

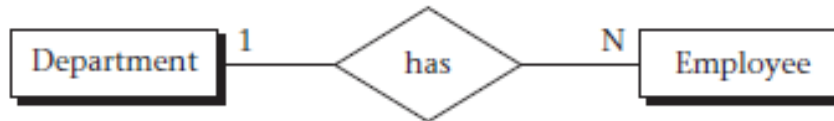
Cardinality and Existence

Connectivity

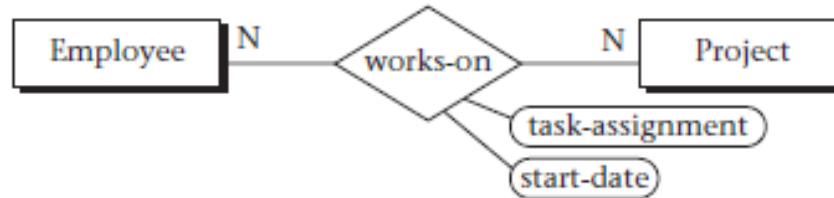
one-to-one



one-to-many



many-to-many

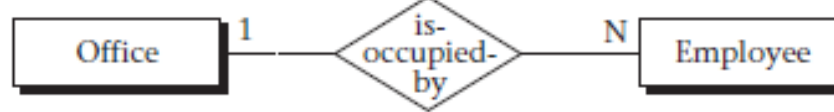


Existence

optional



mandatory



Cardinality

1:1 --- one-to-one --- One entity of type X can be associated with, at most, one entity of type Y . One entity of type Y can be associated with, at most, one entity of type X .

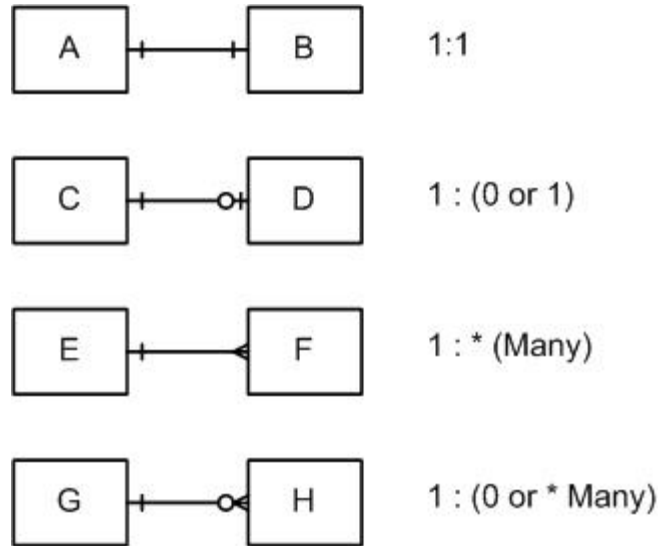
An example: the relationship between *car* and *steering wheel*. A car has only one steering wheel and a steering wheel can only be installed in one car.

1:M --- one-to-many --- One entity of type X can be associated with many entities of type Y . One entity of type Y can be associated with, at most, one entity of type X .

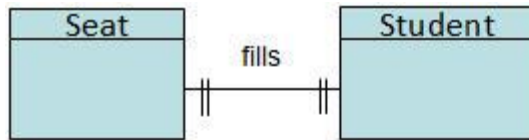
An example: the relationship between *building* and *rooms*. A building can have many rooms but a room can be in, at most, one building.

M:M --- many-to-many --- One entity of type X can be associated with many entities of type Y . One entity of type Y can be associated with many entities of type X . An example: the relationship between a car and its options (such as air conditioning, ABS brakes). A car can have many options and an option can be installed on many cars.

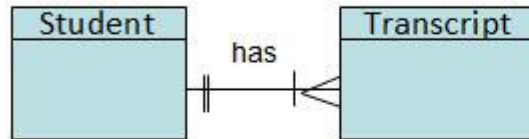
E R Diagram (Crow Foot)



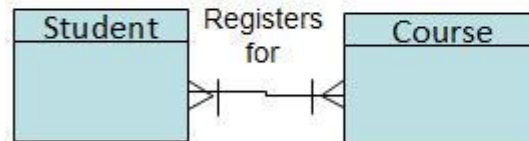
E R Diagram (Crow Foot)



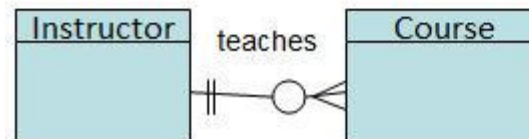
Left to right: one to one, 1:1
Right to left: one to one, 1:1



Left to right: one to many, 1:M
Right to left: many to one, M:1

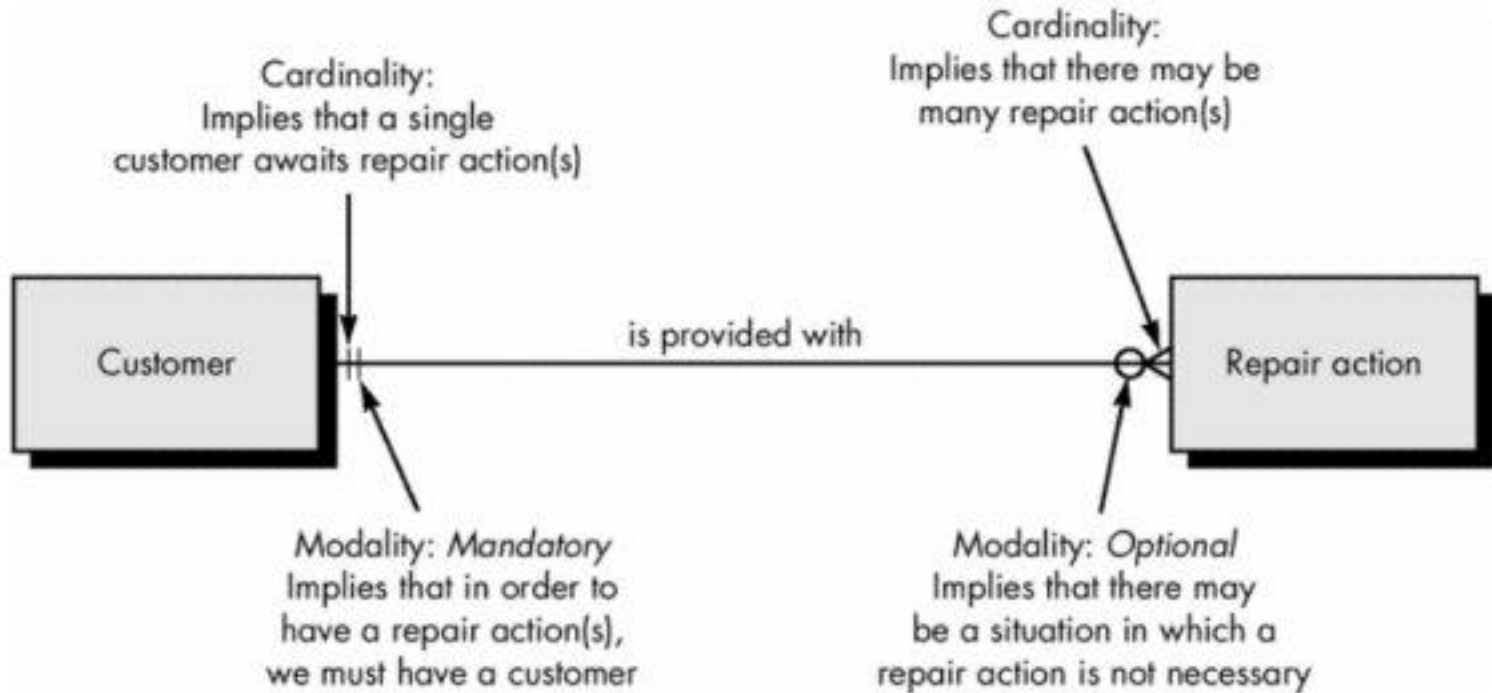


Left to right: many to many, M:M
Right to left: many to many, M:M

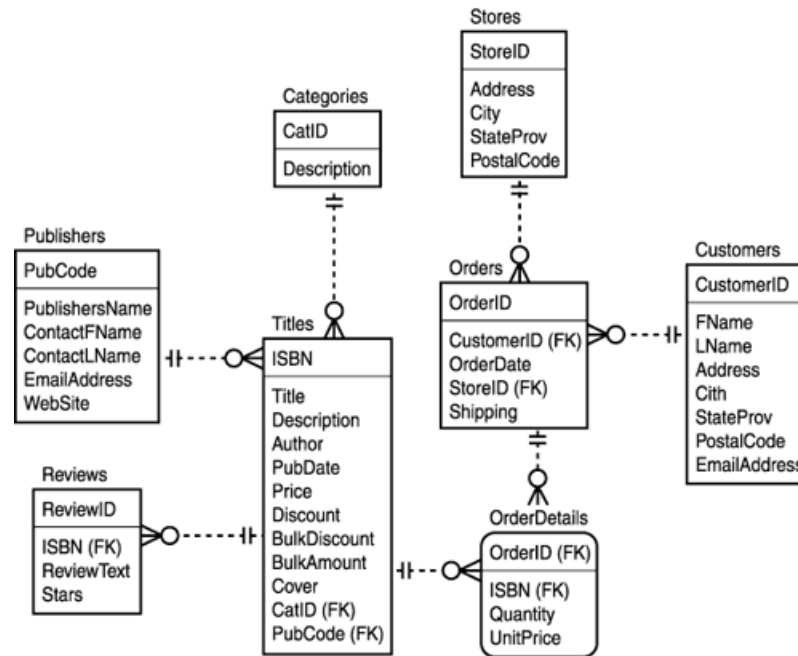


Left to right: one to many, 1:M
Right to left: many to one, M:1

E R Diagram (Crow Foot)



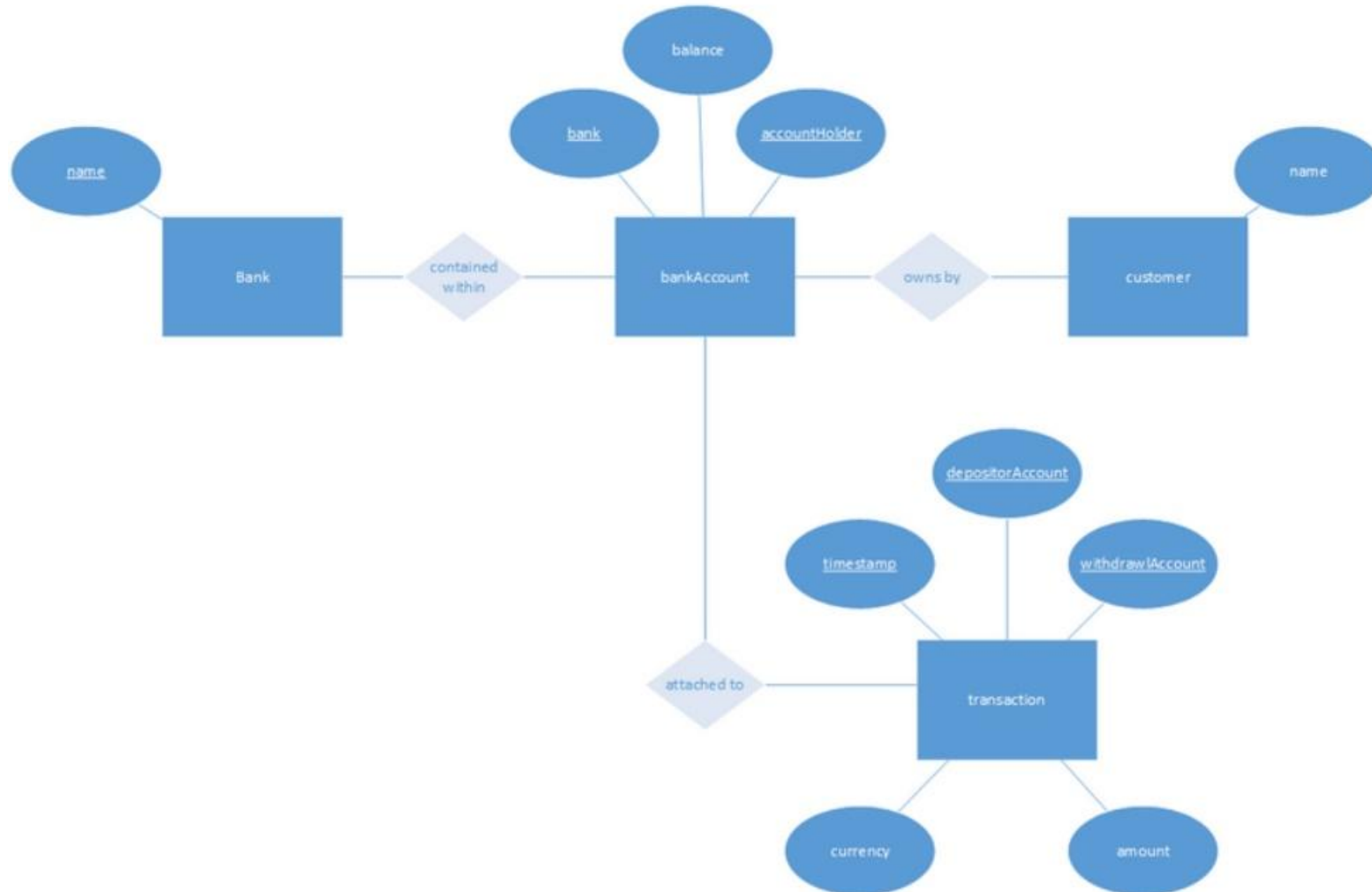
E R Diagram (Crow Foot Example)



TASK 1:

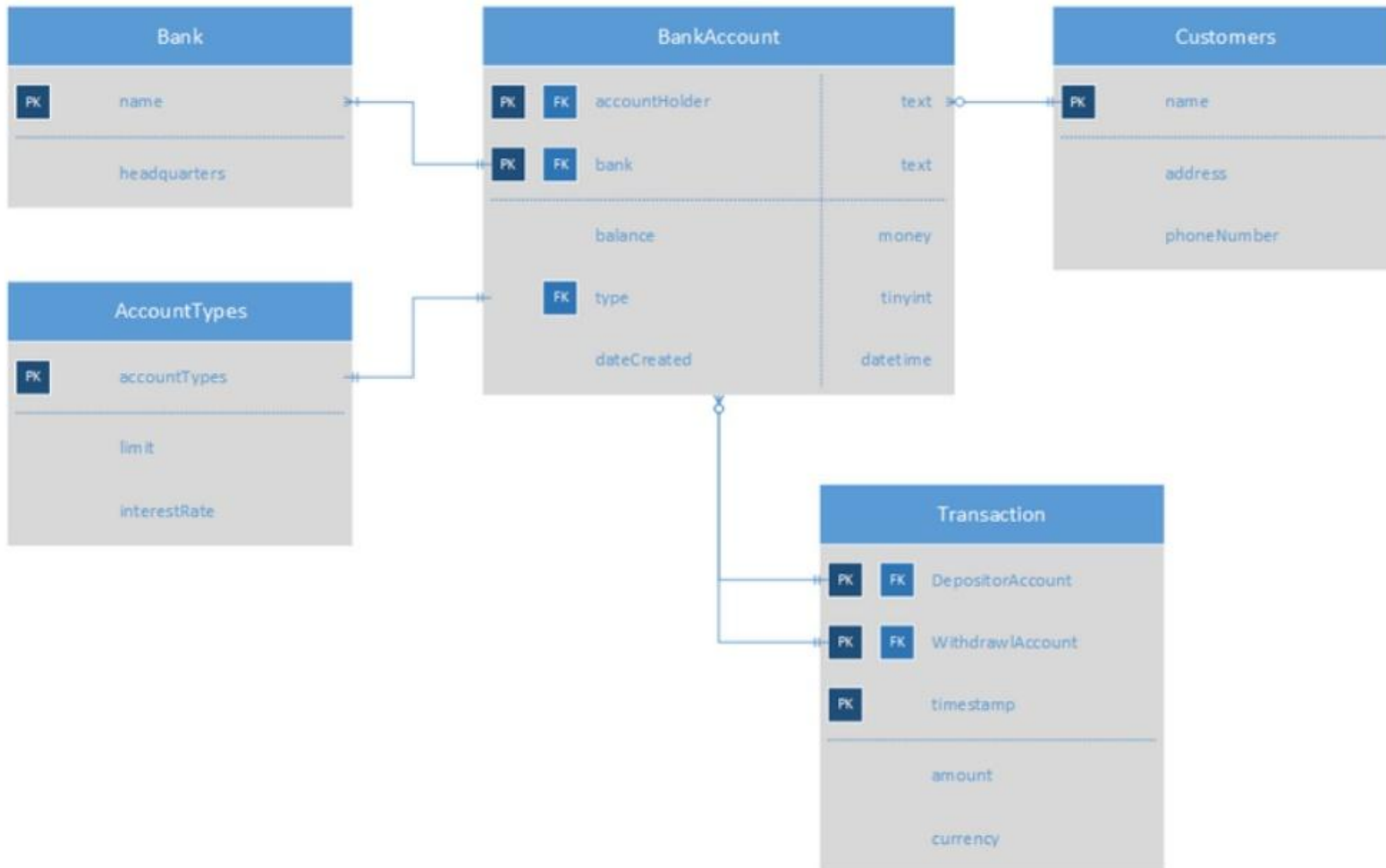
❑ Install MS VISO 2013

❑ Construct following ER Diagram by using chen database notations. Watch video **Create an ER Diagram**.



TASK 2:

- ❑ Watch video **Data Modeling in Visio 2013.**
- ❑ Construct following ER Diagram by using crow foot database notations.



TASK 3:

□ Construct following ER Diagram by using chen database notations. Recreate the diagram into **crow foot notation**.

