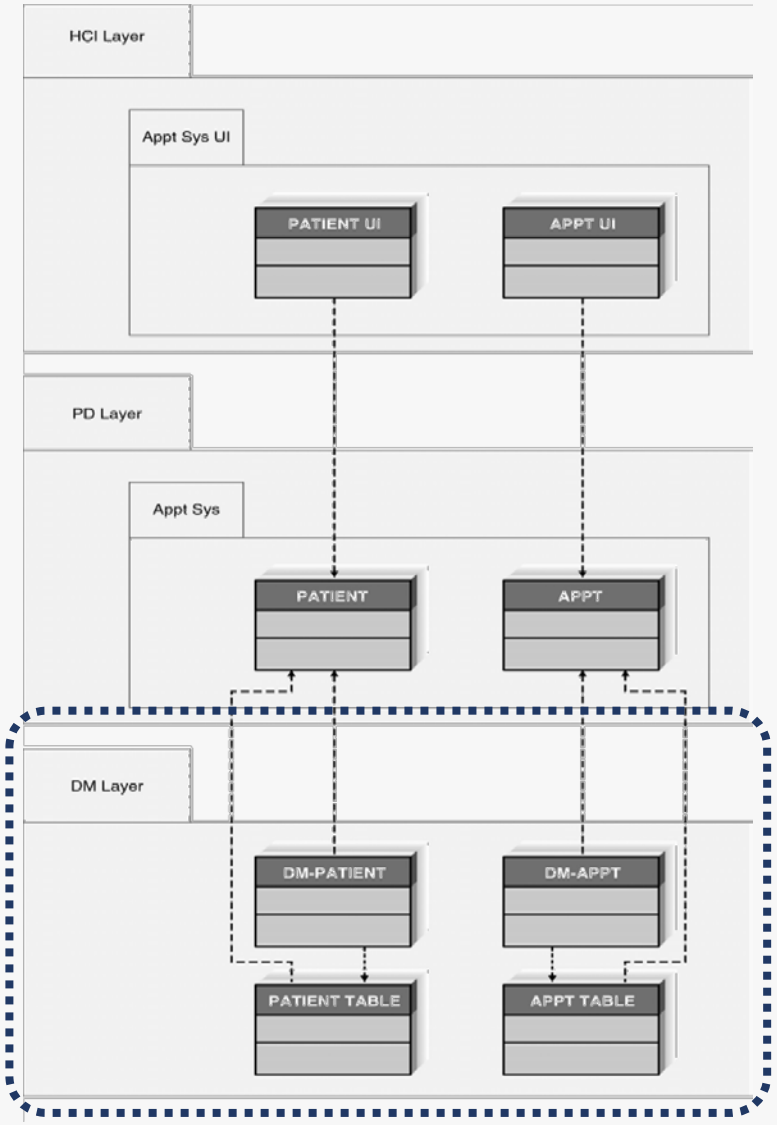
CH11.Design Activities
- Data Layer-

School of Computer Science
Prof. Euijong Lee

Other Design Activities

- **Data Layer Design – ①**
  - From domain object to data, and then to DB table schema
- **User Interface Design – ②**
  - Window navigation
  - Standard interface design
- **Physical Architecture - ③**
  - Hardware platform architecture
  - Deployment diagram



Data Layer Design

Data Management Layer

- **Choose object-persistence format to support the system**
  - **Problem domain objects drive object storage design**
  - **Object-persistence formats**
    - Files (Sequential and Random)
    - Relational databases
    - Object-relational / Object-oriented databases

- **Design of Data Storage**
  - **Transform relationships of class diagram into database table schema**
  - **Must optimize processing efficiency**

테이블의 형태 와 양식

Files

- **Sequential Access**
  - Data stored in order based on a particular attribute
  - Typically efficient for reports using all or most of the file's data
- **Random Access**
  - Data stored in unordered fashion
  - Typically efficient for finding individual records

File Example

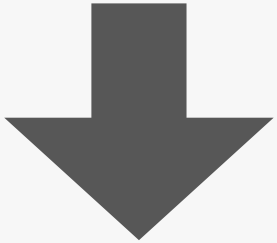
- Customer order file

Order Number	Date	Cust ID	Last Name	First Name	Amount	Tax	Total	Prior Customer	Payment Type
234	11/23/00	2242	DeBerry	Ann	$ 90.00	$5.85	$ 95.85	Y	MC
235	11/23/00	9500	Chin	April	$ 12.00	$0.60	$ 12.60	Y	VISA
236	11/23/00	1556	Fracken	Chris	$ 50.00	$2.50	$ 52.50	N	VISA
237	11/23/00	2242	DeBerry	Ann	$ 75.00	$4.88	$ 79.88	Y	AMEX
238	11/23/00	2242	DeBerry	Ann	$ 60.00	$3.90	$ 63.90	Y	MC
239	11/23/00	1035	Black	John	$ 90.00	$4.50	$ 94.50	Y	AMEX
240	11/23/00	9501	Kaplan	Bruce	$ 50.00	$2.50	$ 52.50	N	VISA
241	11/23/00	1123	Williams	Mary	$120.00	$9.60	$129.60	N	MC
242	11/24/00	9500	Chin	April	$ 60.00	$3.00	$ 63.00	Y	VISA
243	11/24/00	4254	Bailey	Ryan	$ 90.00	$4.50	$ 94.50	Y	VISA
244	11/24/00	9500	Chin	April	$ 24.00	$1.20	$ 25.20	Y	VISA
245	11/24/00	2242	DeBerry	Ann	$ 12.00	$0.78	$ 12.78	Y	AMEX
246	11/24/00	4254	Bailey	Ryan	$ 20.00	$1.00	$ 21.00	Y	MC
247	11/24/00	2241	Jones	Chris	$ 50.00	$2.50	$ 52.50	N	VISA
248	11/24/00	4254	Bailey	Ryan	$ 12.00	$0.60	$ 12.60	Y	AMEX
249	11/24/00	5927	Lee	Diane	$ 50.00	$2.50	$ 52.50	N	AMEX
250	11/24/00	2242	DeBerry	Ann	$ 12.00	$0.78	$ 12.78	Y	MC
251	11/24/00	9500	Chin	April	$ 15.00	$0.75	$ 15.75	Y	MC
252	11/24/00	2242	DeBerry	Ann	$132.00	$8.58	$140.58	Y	MC
253	11/24/00	2242	DeBerry	Ann	$ 72.00	$4.68	$ 76.68	Y	AMEX

(source: Dennis, Alan, Barbara Haley Wixom, and Roberta M. Roth. Systems analysis and design. John wiley & sons.)

Object-Relational Databases (ORDBMS)

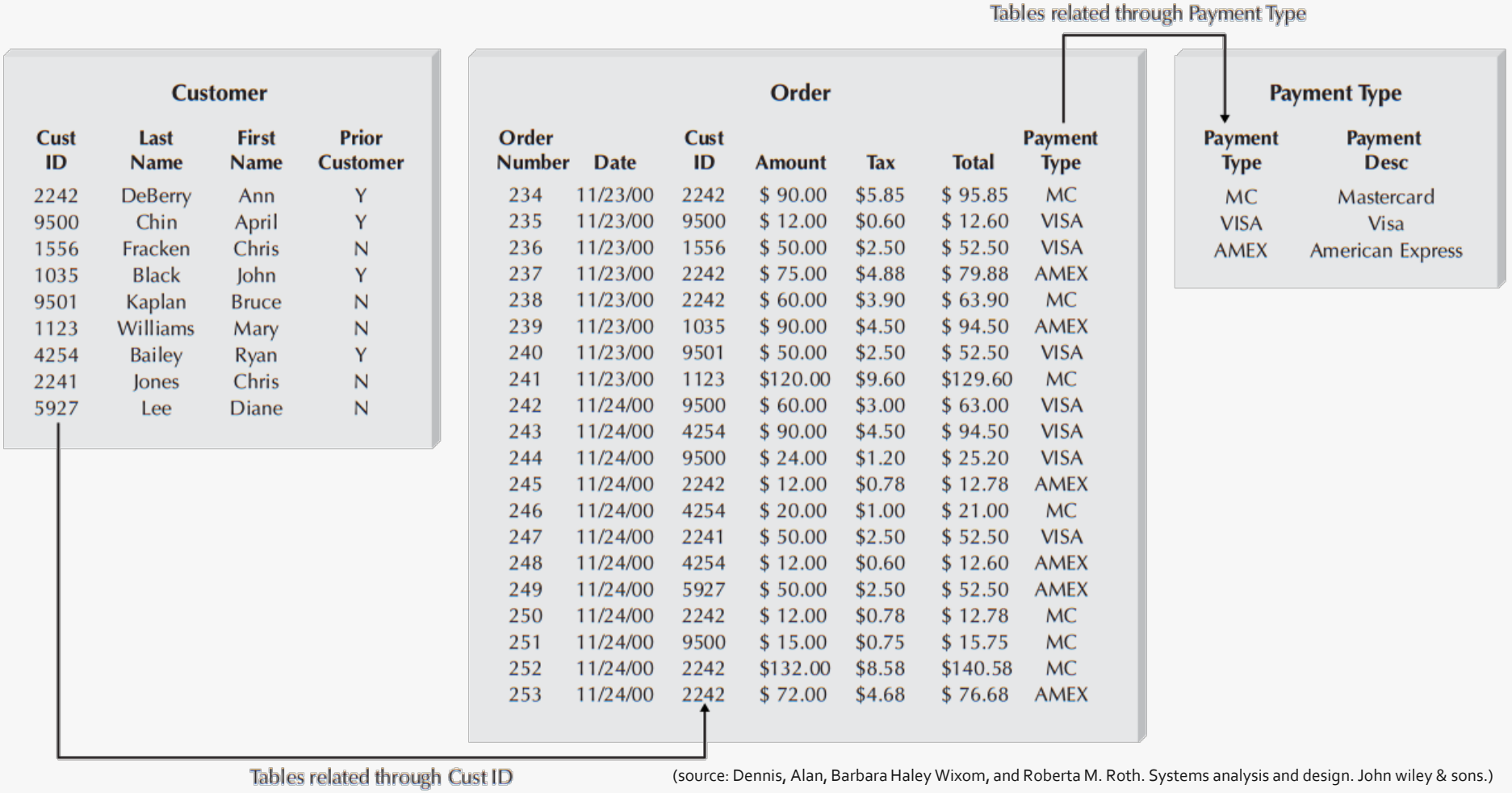
- **Characteristics of RDB**
  - Primary key, Foreign key
  - Referential integrity
  - Structured Query Language (SQL)
  - Tables and joining tables



- **Object-relational databases**
  - Relational databases extended to handle the storage of objects
  - Use of user-defined data types
  - Extended SQL
  - Inheritance tends to be language dependent

Relational Database Example

- Objects must be converted so they can be stored in a table



Customer

Cust ID	Last Name	First Name	Prior Customer
2242	DeBerry	Ann	Y
9500	Chin	April	Y
1556	Fracken	Chris	N
1035	Black	John	Y
9501	Kaplan	Bruce	N
1123	Williams	Mary	N
4254	Bailey	Ryan	Y
2241	Jones	Chris	N
5927	Lee	Diane	N

Order

Order Number	Date	Cust ID	Amount	Tax	Total	Payment Type
234	11/23/00	2242	$ 90.00	$5.85	$ 95.85	MC
235	11/23/00	9500	$ 12.00	$0.60	$ 12.60	VISA
236	11/23/00	1556	$ 50.00	$2.50	$ 52.50	VISA
237	11/23/00	2242	$ 75.00	$4.88	$ 79.88	AMEX
238	11/23/00	2242	$ 60.00	$3.90	$ 63.90	MC
239	11/23/00	1035	$ 90.00	$4.50	$ 94.50	AMEX
240	11/23/00	9501	$ 50.00	$2.50	$ 52.50	VISA
241	11/23/00	1123	$120.00	$9.60	$129.60	MC
242	11/24/00	9500	$ 60.00	$3.00	$ 63.00	VISA
243	11/24/00	4254	$ 90.00	$4.50	$ 94.50	VISA
244	11/24/00	9500	$ 24.00	$1.20	$ 25.20	VISA
245	11/24/00	2242	$ 12.00	$0.78	$ 12.78	AMEX
246	11/24/00	4254	$ 20.00	$1.00	$ 21.00	MC
247	11/24/00	2241	$ 50.00	$2.50	$ 52.50	VISA
248	11/24/00	4254	$ 12.00	$0.60	$ 12.60	AMEX
249	11/24/00	5927	$ 50.00	$2.50	$ 52.50	AMEX
250	11/24/00	2242	$ 12.00	$0.78	$ 12.78	MC
251	11/24/00	9500	$ 15.00	$0.75	$ 15.75	MC
252	11/24/00	2242	$132.00	$8.58	$140.58	MC
253	11/24/00	2242	$ 72.00	$4.68	$ 76.68	AMEX

Payment Type

Payment Type	Payment Desc
MC	Mastercard
VISA	Visa
AMEX	American Express

(source: Dennis, Alan, Barbara Haley Wixom, and Roberta M. Roth. Systems analysis and design. John wiley & sons.)

Object-Oriented Databases (OODMBS)

- **Two approaches**
  - **Adding persistence extensions to OO languages**
  - **Separate database management systems**
    - Extents: set of instance associated with a particular class
    - Object ID: unique identifier assigned to it by OODBMS
    - Some inheritance (supporting different languages)
    - Repeating groups or multivalued attributes
    - Mainly support multimedia applications
    - Steep learning curve

Selecting an Object Persistence Format

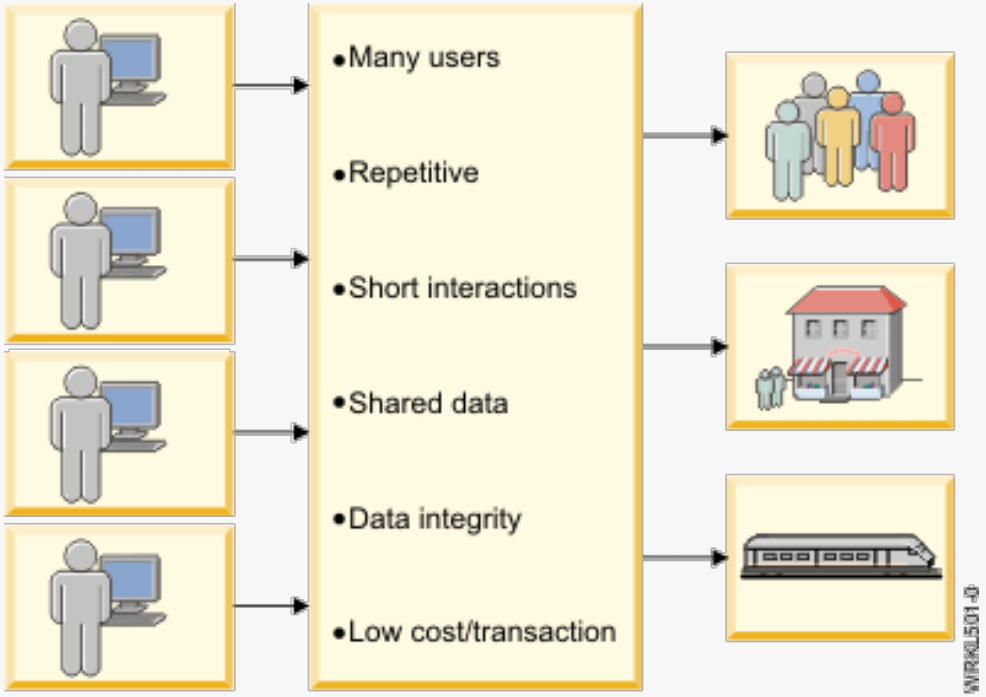
	Sequential and Random Access Files	Relational DBMS	Object Relational DBMS	Object-Oriented DBMS	NoSQL data store
Major Strengths	Usually part of an object-oriented programming language Files can be designed for fast performance Good for short-term data storage	Leader in the database market Can handle diverse data needs	Based on established, proven technology, e.g., SQL Able to handle complex data	Able to handle complex data Direct support for object orientation	Able to handle complex data
Major Weaknesses	Redundant data Data must be updated using programs, i.e., no manipulation or query language No access control	Cannot handle complex data No support for object orientation Impedance mismatch between tables and objects	Limited support for object orientation Impedance mismatch between tables and objects	Technology is still maturing Skills are hard to find	Technology is still maturing Skills are hard to find
Data Types Supported	Simple and Complex	Simple	Simple and Complex	Simple and Complex	Simple and Complex
Types of Application Systems Supported	Transaction processing	Transaction processing and decision making	Transaction processing and decision making	Transaction processing and decision making	Primarily decision making
Existing Storage Formats	Organization dependent	Organization dependent	Organization dependent	Organization dependent	Organization dependent
Future Needs	Poor future prospects	Good future prospects	Good future prospects	Good future prospects	Good future prospects

(source: Dennis, Alan, Barbara Haley Wixom, and Roberta M. Roth. Systems analysis and design. John wiley & sons.)

Additional Information

❖ Transaction System

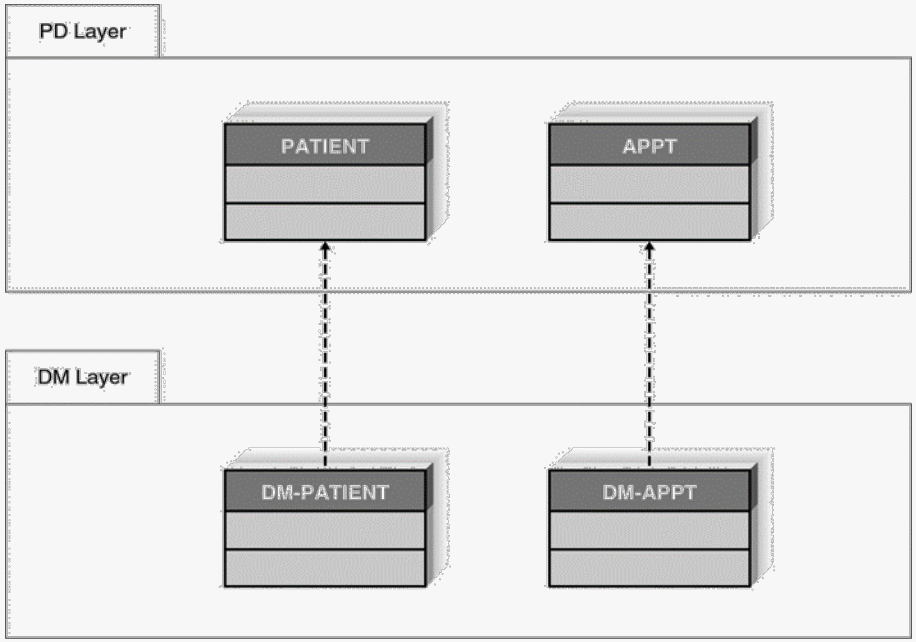
- Transaction systems must be able to support a high number of concurrent users and transaction types
- Transaction: a single action such as adding a sale or a new customer, which changes data in a computer database



Characteristics of
a transaction system (IBM)

Mapping domain object to Object persistence format

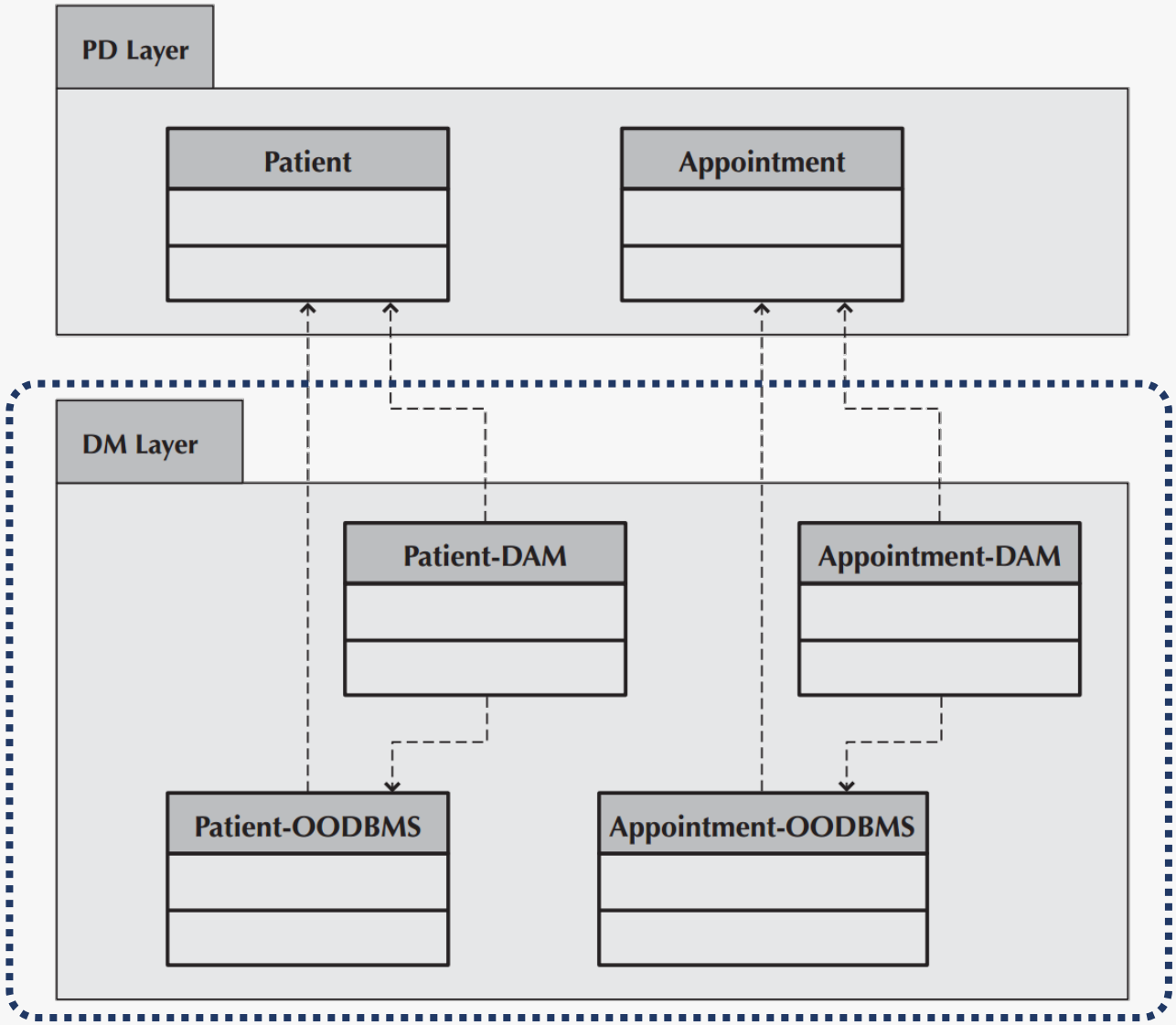
- **Initial Points to Consider**
  - **Adding primary and foreign keys**
    - Unless they add too much overhead
  - **Data management functionality only in classes at data management layer**
    - May add overhead, but aids in portability and reuse



(source: Dennis, Alan, Barbara Haley Wixom, and Roberta M. Roth. Systems analysis and design. John wiley & sons.)

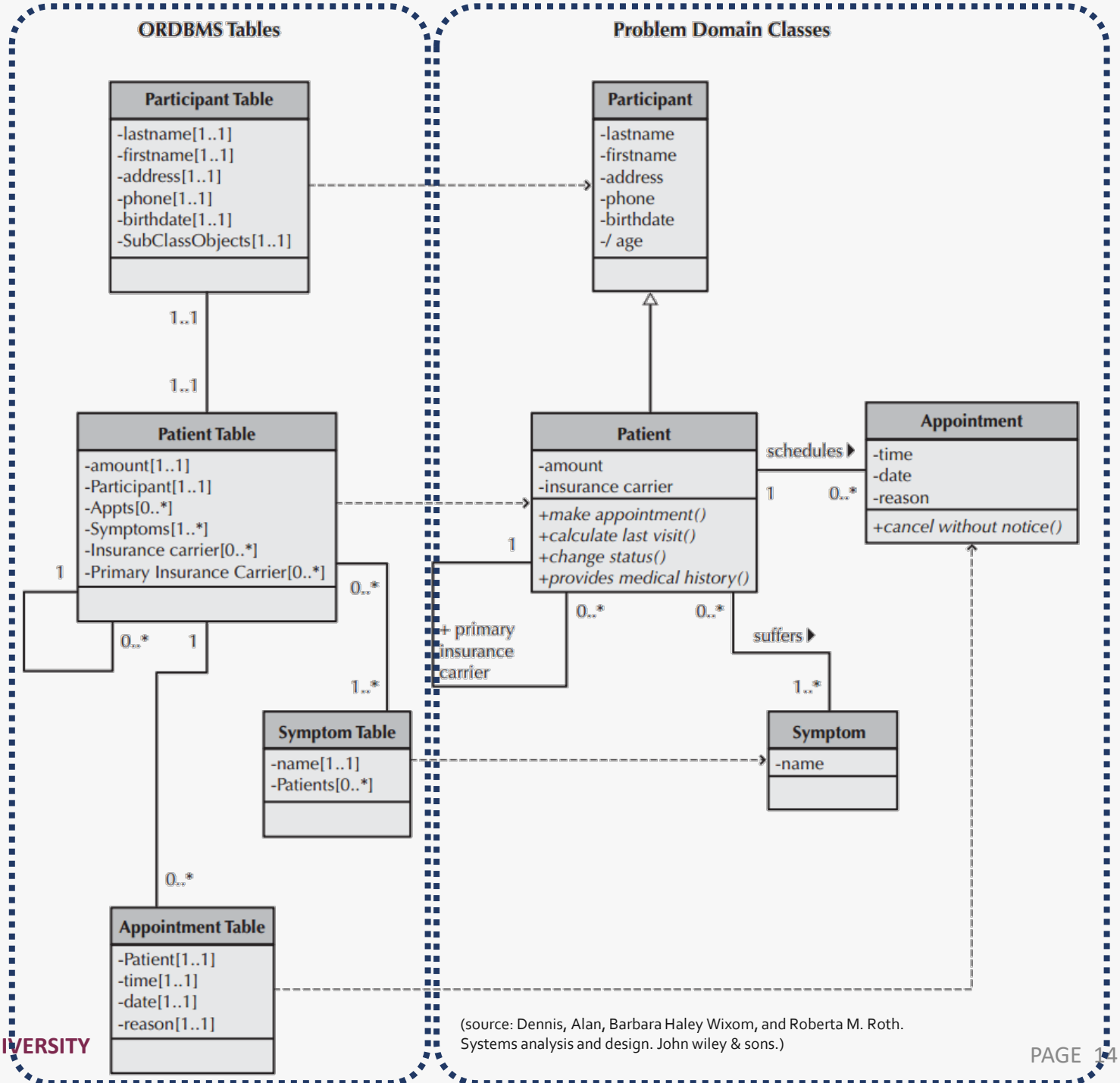
Mapping Objects to Object-Persistence Formats

- Example



(source: Dennis, Alan, Barbara Haley Wixom, and Roberta M. Roth. Systems analysis and design. John wiley & sons.)

Mapping (1)



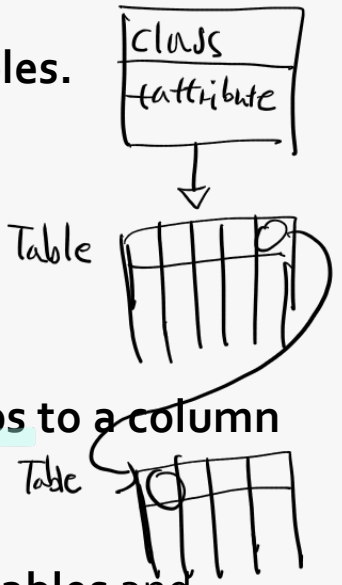
(source: Dennis, Alan, Barbara Haley Wixom, and Roberta M. Roth. Systems analysis and design. John wiley & sons.)

Mapping PD Objects to RDBMS Schema

Rule 1: Map all concrete problem **domain classes to the RDBMS tables.**

Rule 2: Map single valued **attributes to columns of the tables.**

Rule 3: Map **methods** to stored procedures or **to program modules.**



Rule 4: Map single-valued **aggregation and association relationships to a column** that can store the key of the related table

Rule 5: Map **multi-valued attributes and repeating groups to new tables** and create a one-to-many association from the original table to the new ones.

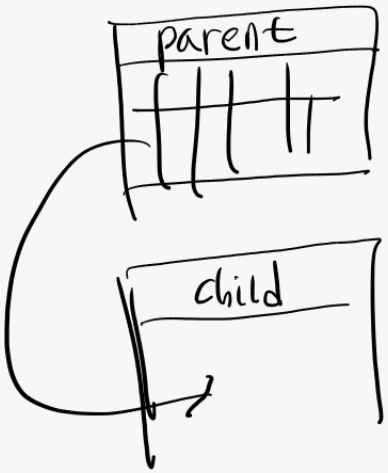
Rule 6: Map **multi-valued aggregation and association relationships to a new associative table** that relates the two original tables together.
Copy the primary key from both original tables to the new associative table

Mapping PD Objects to RDBMS Schema

Rule 7: For aggregation and association **relationships of mixed type,**
copy the primary key from the single-valued side (1..1 or 0..1) of the relationship to a new column in the table on the multi-valued side (1..* or 0..*) of the relationship that can store the key of the related table

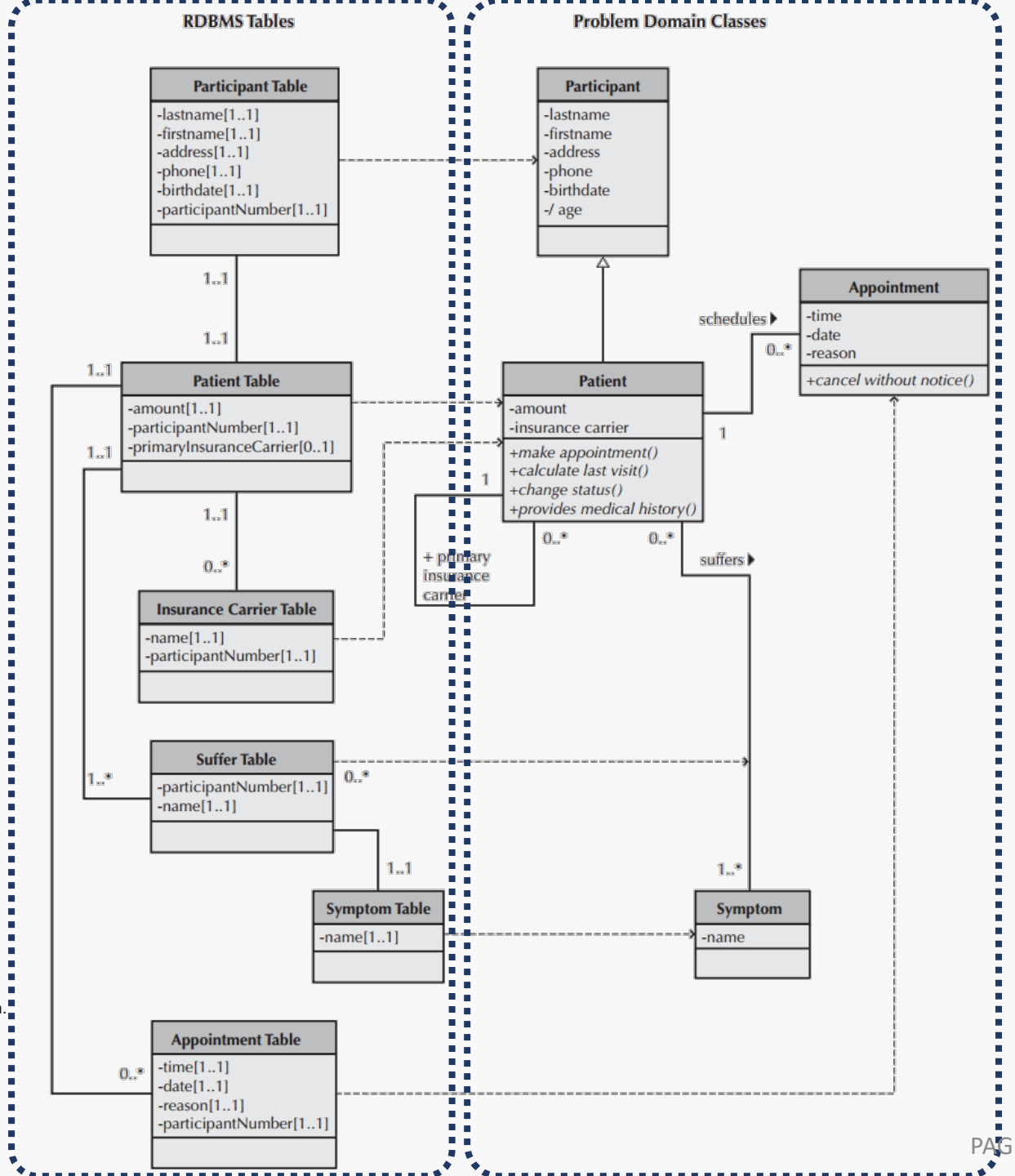
Rule 8a: Ensure that the **primary key of the subclass** instance is **the same as the primary key of the superclass**.. OR

Rule 8b: Flatten the inheritance



Flatten : 평탄화.
부모 클래스의 속성을 그대로 자식 클래스에 복붙(정리).

Mapping (2)

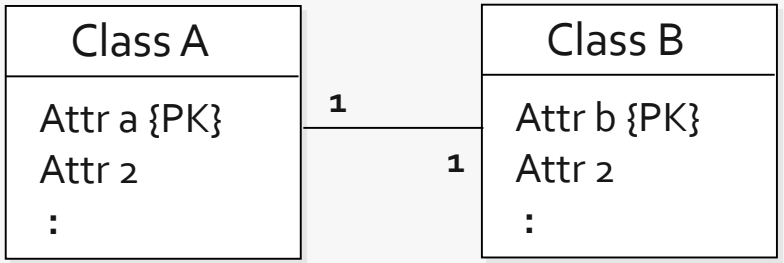


(source: Dennis, Alan, Barbara Haley Wixom, and Roberta M. Roth. Systems analysis and design. John wiley & sons.)

From Domain Class to RDB Table

- **Designing Relational Database Schema**
  - **Transforms the domain class model to RDB schema**
    - by Entity-Relationship Diagram (ERD)
    - by DDL(Data Definition Language) from some Tools
  - **Transformation Rules**
    - Association relationship
      : one-to-one, one-to-many, many-to-many
    - Inheritance relationship

One-to-One : All Mandatory (Rule 4)



- a table per a class
- add secondary key into one table with the primary key of the other table (secondary key is NOT NULL)

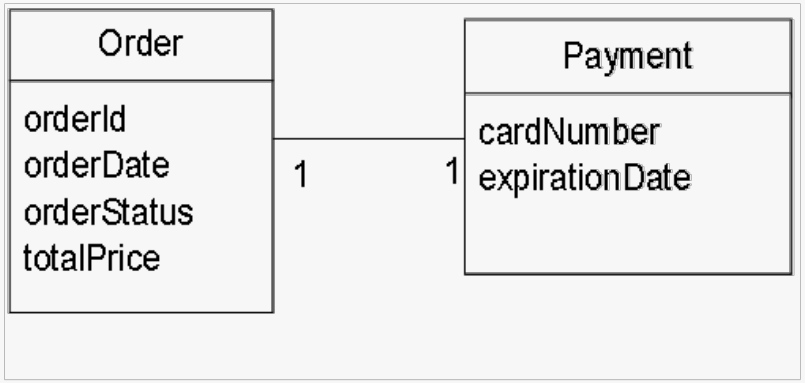
T_A

Attr a {PK}	Attr 2	....

T_B

Attr b {PK}	Attr 2	Attr a {SK}	....

Example : One-to-One All Mandatory



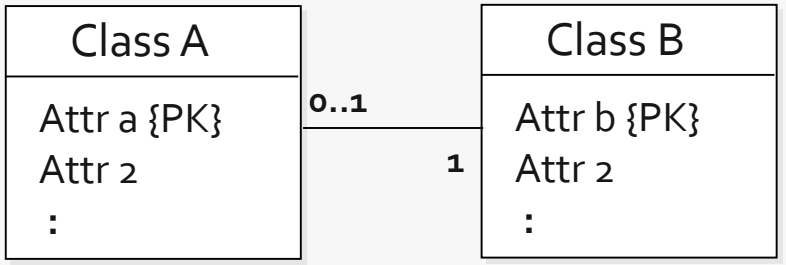
T_Order

Attributes	Types
orderId	**VARCHAR(16)**
orderDate	**Date**
orderStatus	**Integer**
totalPrice	**Integer**
cardNumber {SK}	**VARCHAR(16)**

T_Payment

Attributes	Types
cardNumber	**VARCHAR(16)**
expirationDate	**Date**

One-to-One : One Optional (Rule 4)



- a table per a class
- add secondary key into one table (with o multiplicity) with the primary key of the other table (secondary key is NOT NULL)

A ← B

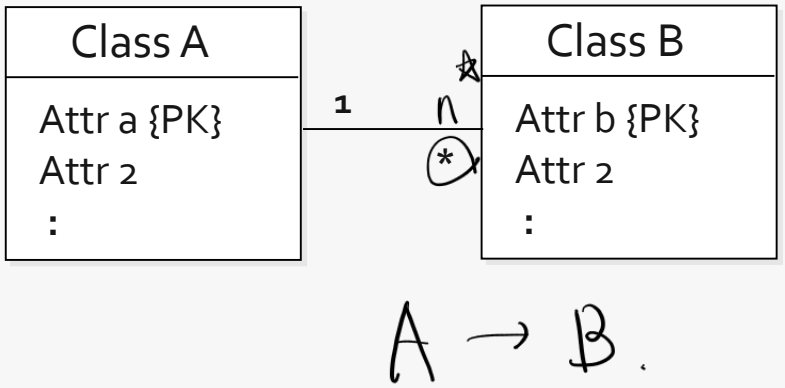
T_A

Attr a {PK}	Attr 2	Attr b {SK}	....

T_B

Attr b {PK}	Attr 2	....

One-to-Many : Mandatory One



- a table per a class
- add secondary key into table having many multiplicity with primary key of the other table
  (secondary key is NOT NULL)

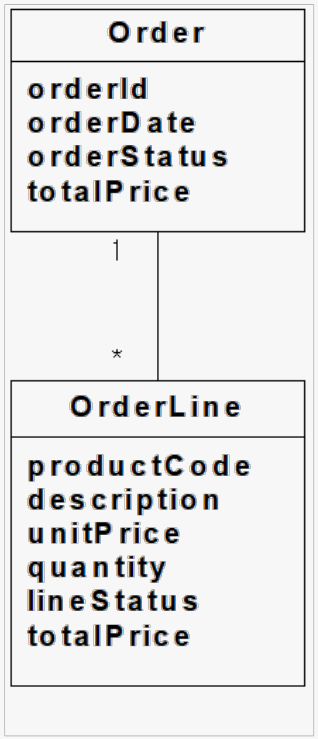
T_A

Attr a { PK }	Attr 2	....

T_B

Attr b { PK }	Attr 2	Attr a {SK}	....

Example : One-to-Many



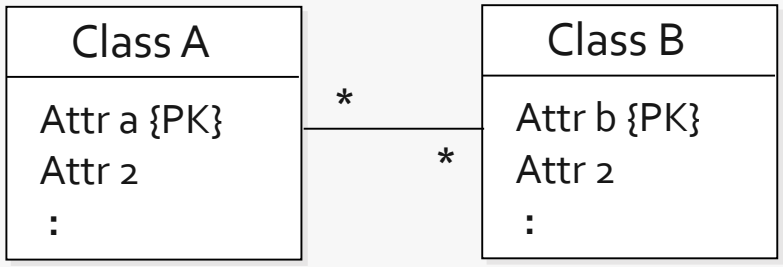
T_Order

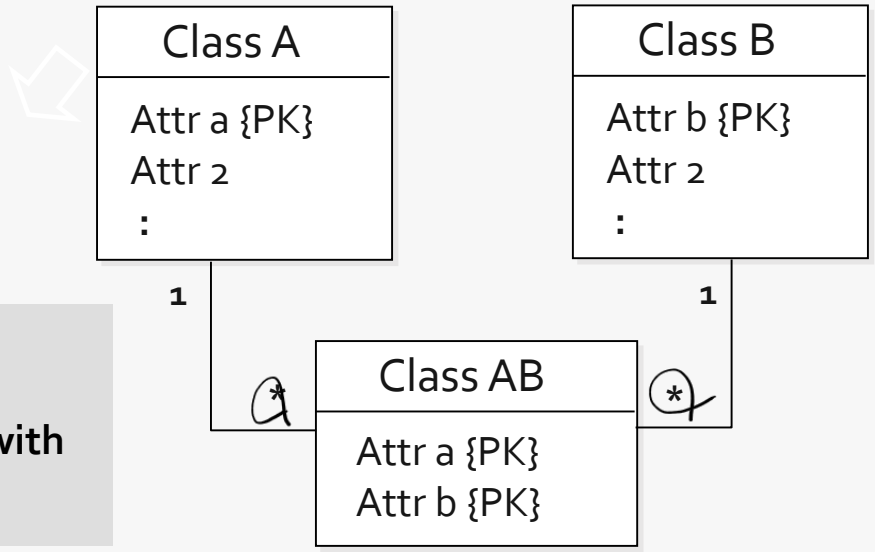
Attributes	Types
orderId	**VARCHAR(16)**
orderDate	**Date**
orderStatus	**Integer**
totalPrice	**Integer**

T_OrderLine

Attributes	Types
productCode	**VARCHAR(10)**
description	**VARVHAR(24)**
unitPrice	**Integer**
quantity	**Ineteger**
lineStatus	**Integer**
totalPrice	**Integer**
orderId {SK}	**VARCHAR(16)**

Many-to-Many : Intersecting Class (Rule 6)





- make intersecting Class
- transform many-to-many to one-to-many
- add key attributes into the intersecting class with primary key of both origin classes

- a table per a class
- attributes in the intersecting class : primary and also secondary key

Many-to-Many : Intersecting Class

T_A

Attr a { PK }	Attr 2	….

T_B

Attr b {PK }	Attr 2	….

T_AB

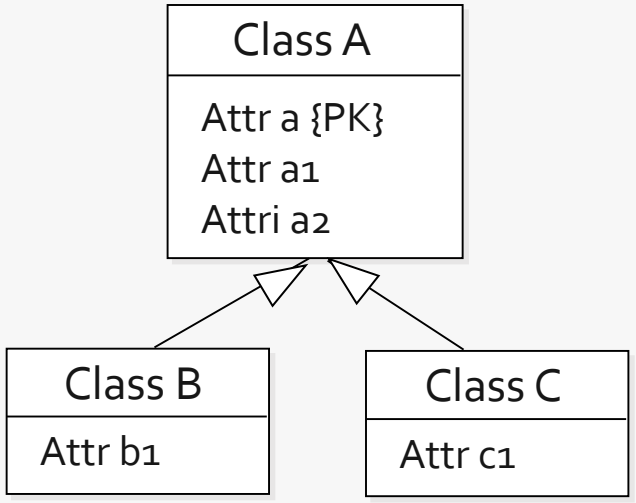
Attr a {PK, SK }	Attr b {PK, SK}	….

1:1 A ← B

1:n A → B

n:n AB ← A
↑ B

Inheritance Relationship I (Rule 8a)



- a table per a class
- add a flag attribute into super class table corresponding to sub classes
- add primary key into subclasses' table with primary key of super class (Key inheritance)

T_A

Attr a {PK}	Attr a1	Attr a2	Flag

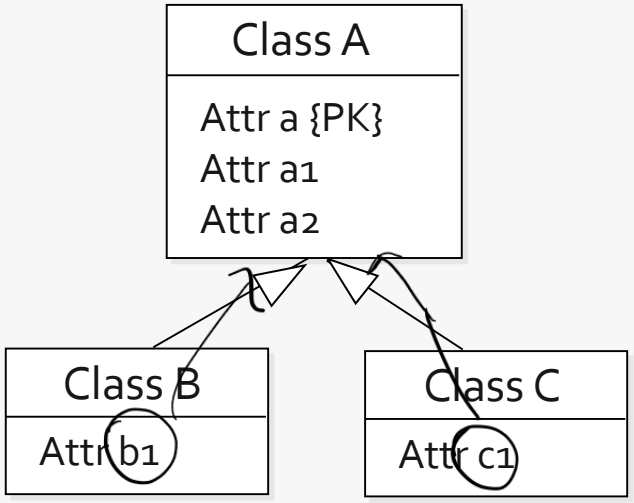
T_B

Attr a {PK, SK}	Attr b1	….

T_C

Attr a {PK, SK}	Attr c1	….

Inheritance Relationship II (Rule 8b)



- a table per a subclass
- add all attributes of super class into subclass table corresponding to sub classes
- primary key of subclasses' table is the primary key of super class (Key inheritance)

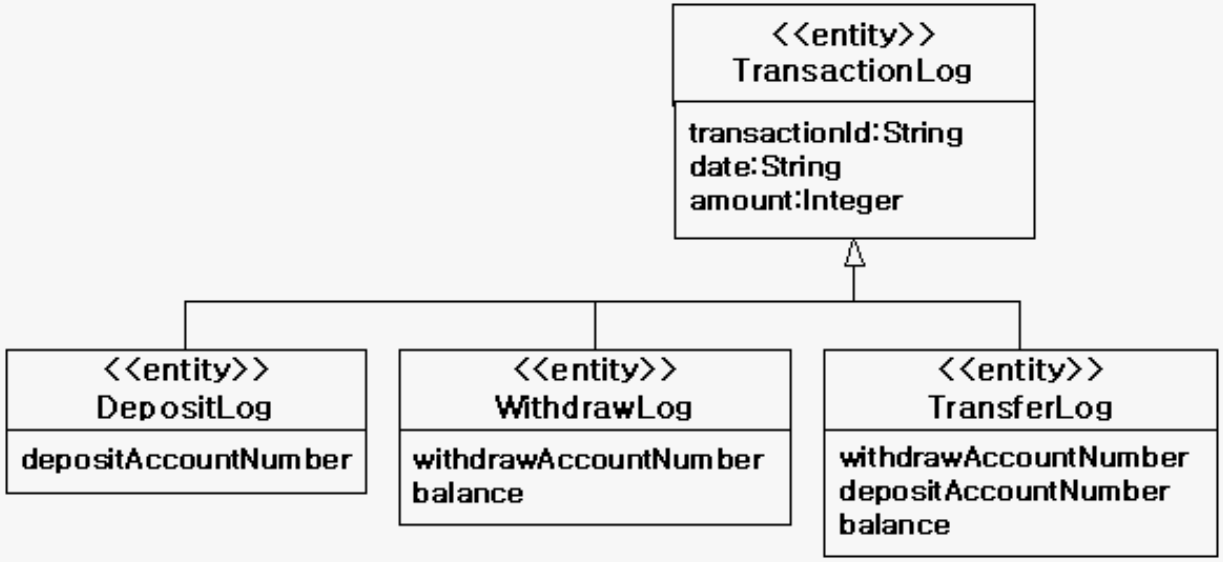
T_B

Attr a {PK}	Attr a1	Attr a2	Attr b1	….

T_C

Attr a {PK}	Attr a1	Attr a2	Attr c1	….

Example : Inheritance



T_DepositLog

Attributes	Types
transactionId {PK}	**VARCHAR(10)**
depositAccountNumber	**VARCHAR(20)**
date	**Date**
amount	**Integer**

T_WithdrawLog

Attributes	Types
transactionId {PK}	**VARCHAR(10)**
withdrawAccountNumber	**VARCHAR(20)**
date	**Date**
amount	**Integer**
balance	**Integer**

Case Study : CD Selection Company

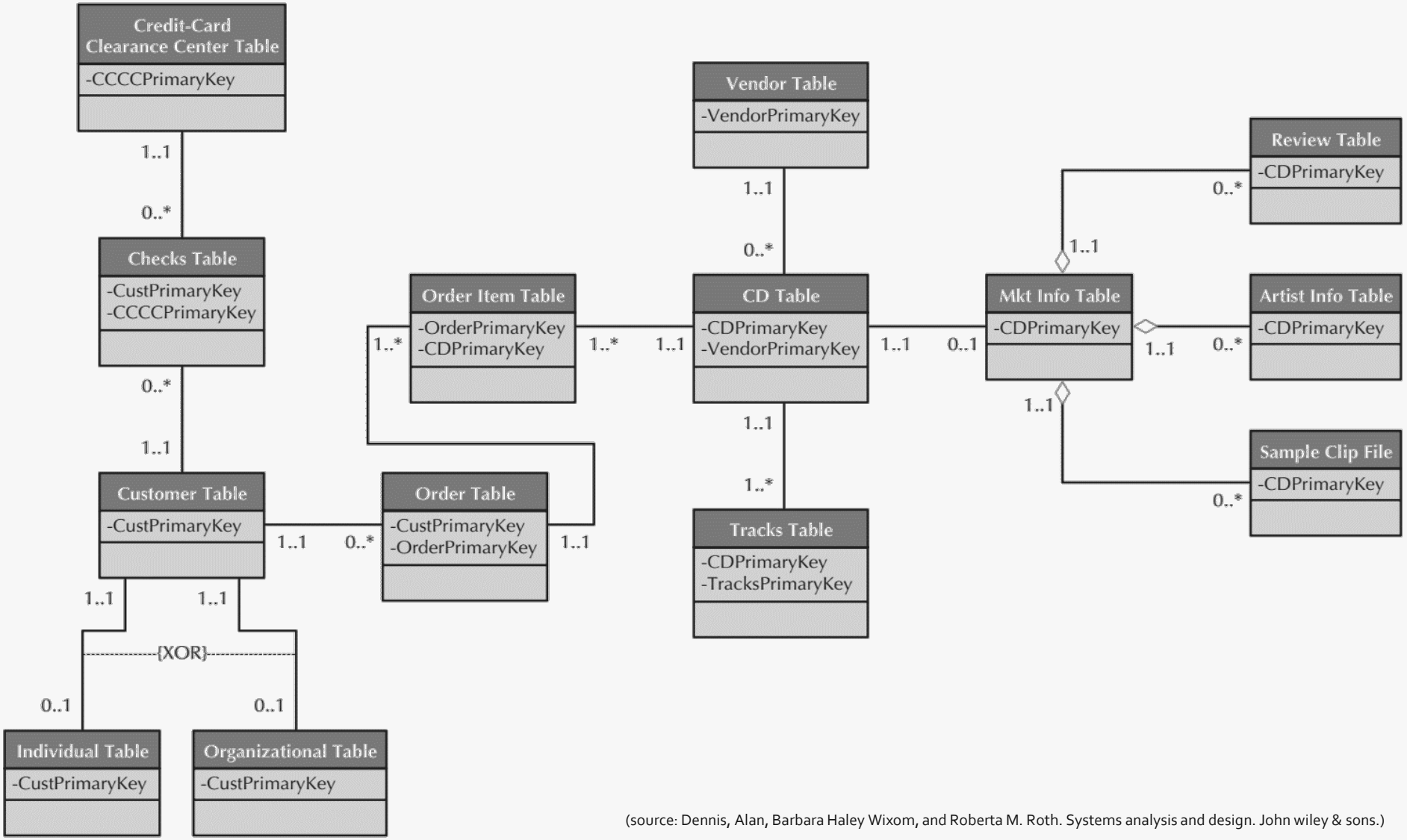
- Most of the data would be text and numbers
- Thus a relational database would be able to handle the data effectively
- However, images for the catalog require complex data objects for sound and video

Looking at the Data Needs (CD Selection Company)

Data	Type	Use	Suggested Format
Customer information	Simple (mostly text)	Transactions	Relational
Order Information	Simple (text and numbers)	Transactions	Relational
Marketing Information	Both simple and complex (eventually the system will contain audio clips, video, etc.)	Transactions	Object add-on?
Information that will be exchanged with the Distribution System	Simple text, formatted specifically for importing into the Distribution System	Transactions	Transaction file
Temporary Information	The Web component will likely need to hold information for temporary periods of time. (e.g., the shopping card will store order information before the order is actually placed)	Transactions	Transaction file

(source: Dennis, Alan, Barbara Haley Wixom, and Roberta M. Roth. Systems analysis and design. John wiley & sons.)

Object Persistent Design



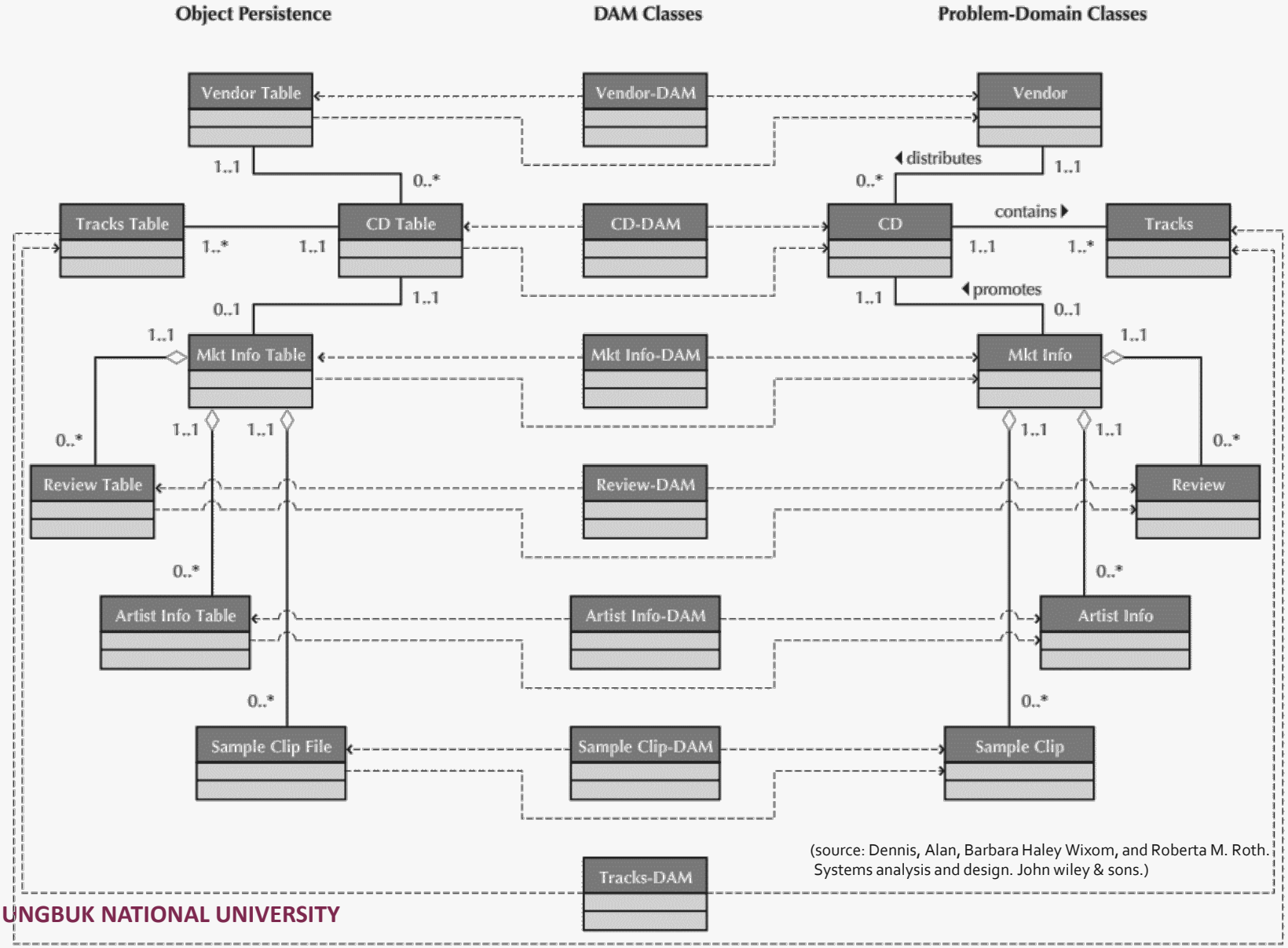
(source: Dennis, Alan, Barbara Haley Wixom, and Roberta M. Roth. Systems analysis and design. John wiley & sons.)

Optimizing Application

Target	Comments	Suggestions to Improve Data Access Speed
All tables	Basic table manipulation	• Investigate if records should be clustered physically by primary key • Create indexes for primary keys • Create indexes for foreign key fields
All tables	Sorts and Grouping	• Create indexes for fields that are frequently sorted or grouped
CD information	Users will need to search CD information by title, artist, and category	• Create indexes for CD title, artist, and category
Order Information	Operators should be able to locate information about a particular customer's order	• Create an index in the Order table for orders by customer name
Entire Physical Model	Investigate denormalization opportunities for all fields that are not updated very often	• Investigate one-to-one relationships • Investigate look-up tables • Investigate one-to-many relationships

(source: Dennis, Alan, Barbara Haley Wixom, and Roberta M. Roth. Systems analysis and design. John wiley & sons.)

Problem Domain Layer



(source: Dennis, Alan, Barbara Haley Wixom, and Roberta M. Roth. Systems analysis and design. John wiley & sons.)

Optimizing Access Speed (Additional Slides)

- **Dimensions of Data Storage Optimization**
  - **Storage efficiency (minimizing storage space)**
  - **Speed of access (minimizing time to retrieve desired information)**

- **Optimizing Storage Efficiency**
  - **Reduce redundant data**
  - **Limit null values**
    - Multiple possible interpretations can lead to mistakes
  - **A well-formed logical data model does not contain redundancy or many null values**

The Steps of Normalization (Additional Slides)

0 Normal Form

Do any tables have repeating fields? Do some records have a different number of columns from other records?	Yes: Remove the repeating fields. Add a new table that contains the fields that repeat.
	No: The data model is in 1NF

First Normal Form

Is the primary key made up of more than one field? If so, do any fields depend on only a part of the primary key?	Yes: Remove the partial dependency. Add a new table that contains the fields that are partially dependent.
	No: The data model is in 2NF

Second Normal Form

Do any fields depend on another nonprimary key field?	Yes: Remove the transitive dependency. Add a new table that contains the fields that are transitively dependent.
	No: The data model is in 3NF

Third Normal Form

(source: Dennis, Alan, Barbara Haley Wixom, and Roberta M. Roth. Systems analysis and design. John wiley & sons.)

Guidelines for Creating Indexes (Additional Slides)

- **Use indexes sparingly for transaction systems**
- **Use many indexes to increase response times in decision support systems**
- **For each table**
  - Create a unique index based on the primary key
  - Create an index based on the foreign key
- **Create an index for fields used frequently for grouping, sorting, or criteria**

Summary and Discussion

- **Data Layer Design**
  - Map problem domain objects to Data
  - From Data objects to DB table schema by transformation rules