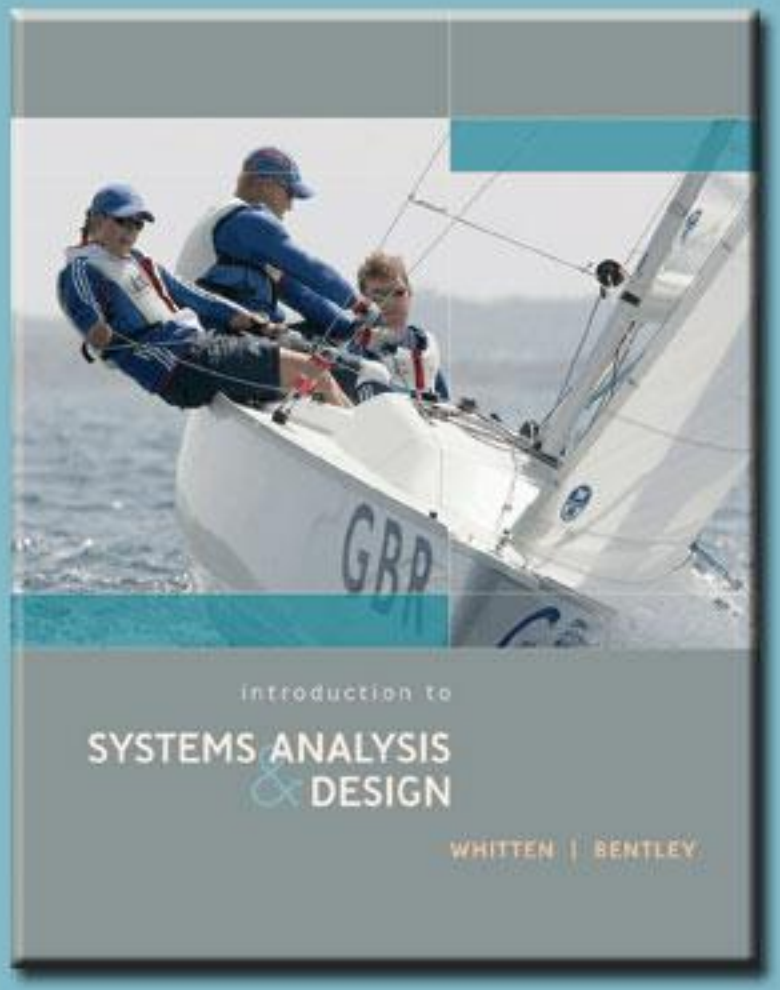


Chapter 13

Database Design

Objectives

- Define and give examples of fields, records, files, and databases.
- Describe modern data architecture of files, operational databases, data warehouses, personal databases, and work group databases.
- Compare roles of systems analyst, database administrator, and data administrator.
- Describe architecture of database management system
- Describe how a relational database implements entities, attributes, and relationships from a logical data model.
- Transform a logical data model into a physical, relational database schema.
- Generate SQL to create the database structure in a schema.

Fields

Field – the smallest unit of meaningful data to be stored in a database

- the physical implementation of a data attribute

Fields (continued)

Primary key – a field that uniquely identifies a record.

Secondary key – a field that identifies a single record or a subset of related records.

Foreign key – a field that points to records in a different file.

Descriptive field – any nonkey field.

Records

Record – a collection of fields arranged in a predetermined format.

- Fixed-length record structures
- Variable-length record structures

Blocking factor – the number of logical records included in a single read or write operation (from the computer's perspective).

Files and Tables

File – the set of all occurrences of a given record structure.

Table – the relational database equivalent of a file.

Types of conventional files and tables

- **Master files** – Records relatively permanent though values may change
- **Transaction files** – Records describe business events
- **Document files** – Historical data for review without overhead of regenerating document
- **Archival files** – Master and transaction records that have been deleted
- **Table lookup files** – Relatively static data that can be shared to maintain consistency
- **Audit files** – Special records of updates to other files

File and Table Design

- Older file design methods required analyst to specify precisely how records should be:
  - Sequenced (File organization)
  - Accessed (File access)
- Database technology usually predetermines and/or limits this
  - Trained database administrator may be given some control over organization, storage location, and access methods for performance tuning.

Data Architecture

Data architecture – a definition of how:

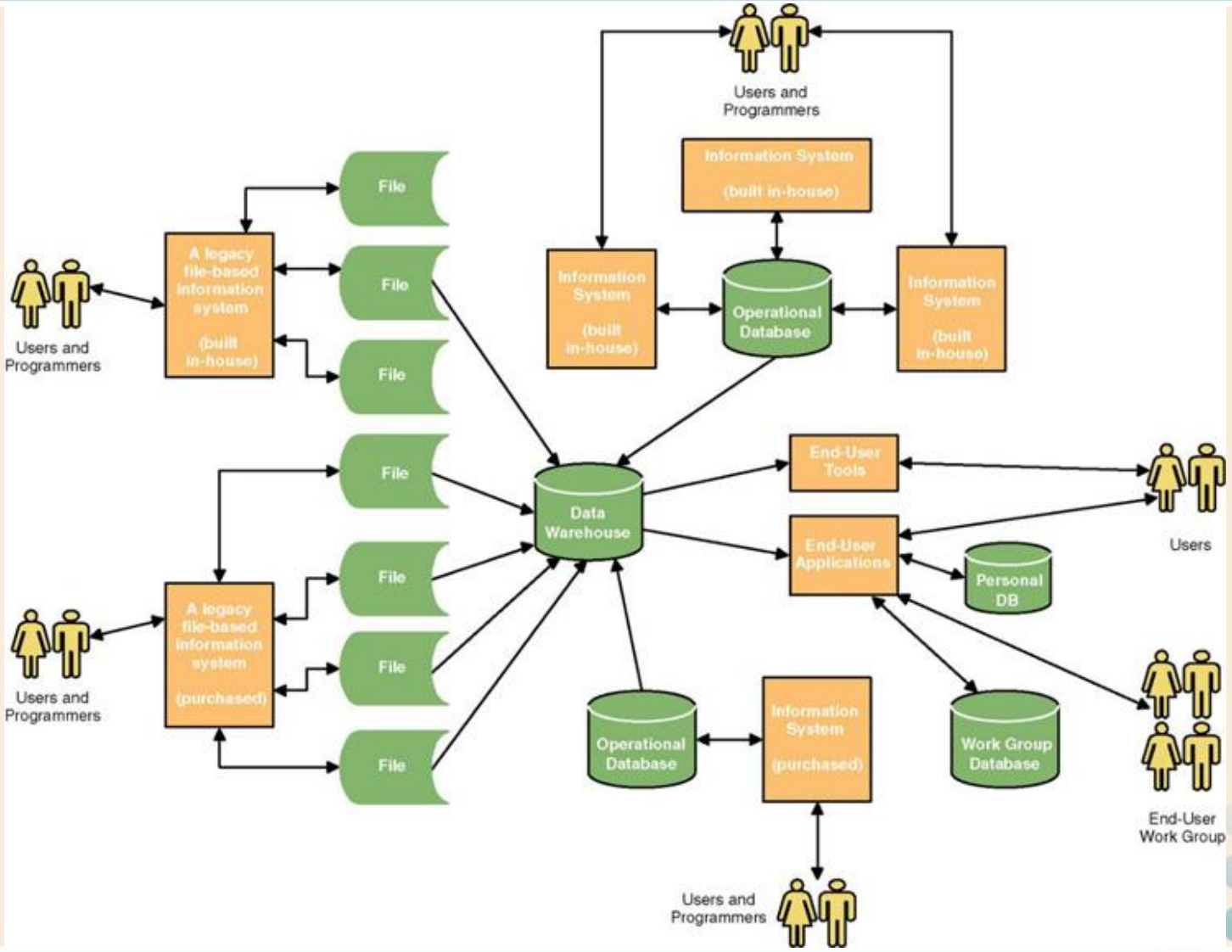
- Files and databases are to be developed and used to store data
- The file and/or database technology to be used
- The administrative structure set up to manage the data resource

Data Architecture (continued)

Data is stored in some combination of:

- **Conventional files**
- **Operational databases** – databases that support day-to-day operations and transactions for an information system. Also called transactional databases.
- **Data warehouses** – databases that store data extracted from operational databases.
  - To support data mining
- **Personal databases**
- **Work group databases**

A Modern Data Architecture



Administrators

Data administrator – a database specialist responsible for data planning, definition, architecture, and management.

Database administrator – a specialist responsible for database technology, database design,construction, security, backup and recovery, and performance tuning.

- A database administrator will administer one or more databases

Database Architecture

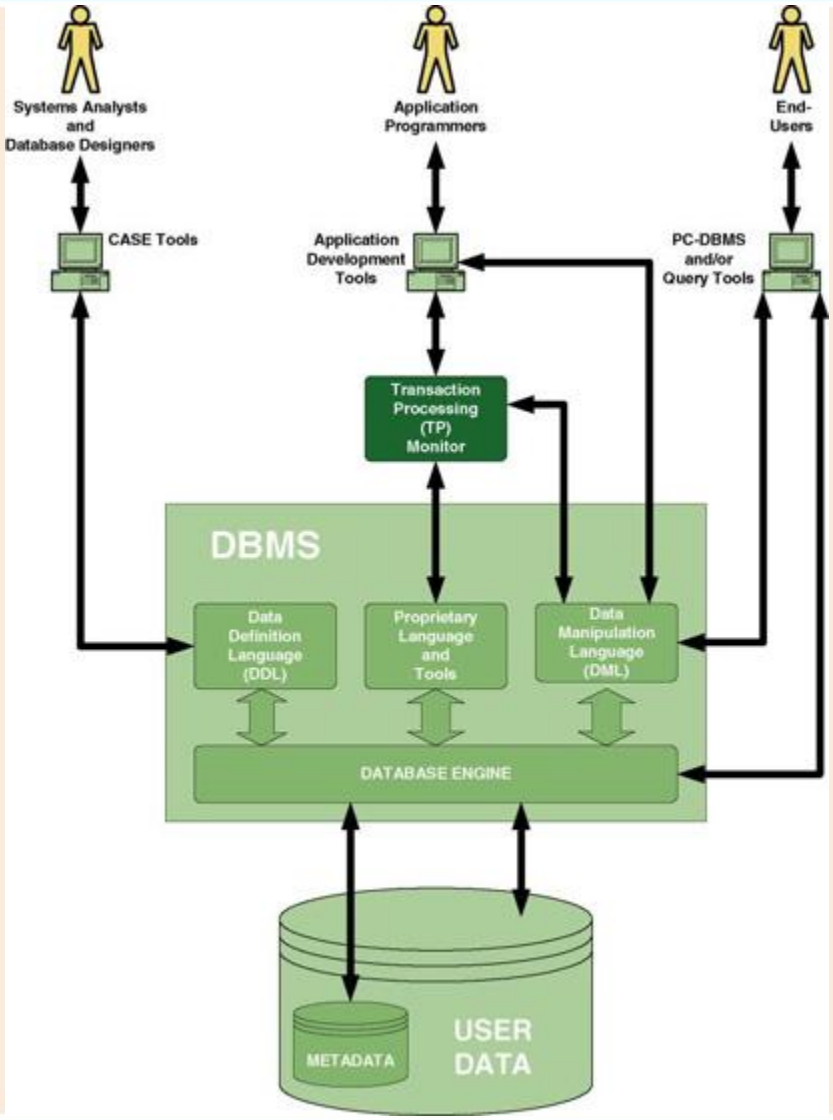
Database architecture – the database technology used to support data architecture

- Including the database engine, database utilities, CASE tools, and database development tools.

Database management system (DBMS) – special software used to create, access, control, and manage a database.

- The core of the DBMS is its **database engine**.
- A data definition language (DDL) is used to physically define tables, fields, and structural relationships.
- A data manipulation language (DML) is used to create, read, update, and delete records in database and navigate between records.

Typical DBMS Architecture

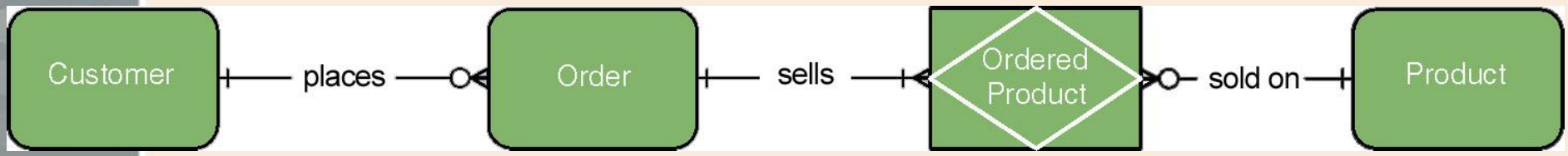


Relational Databases

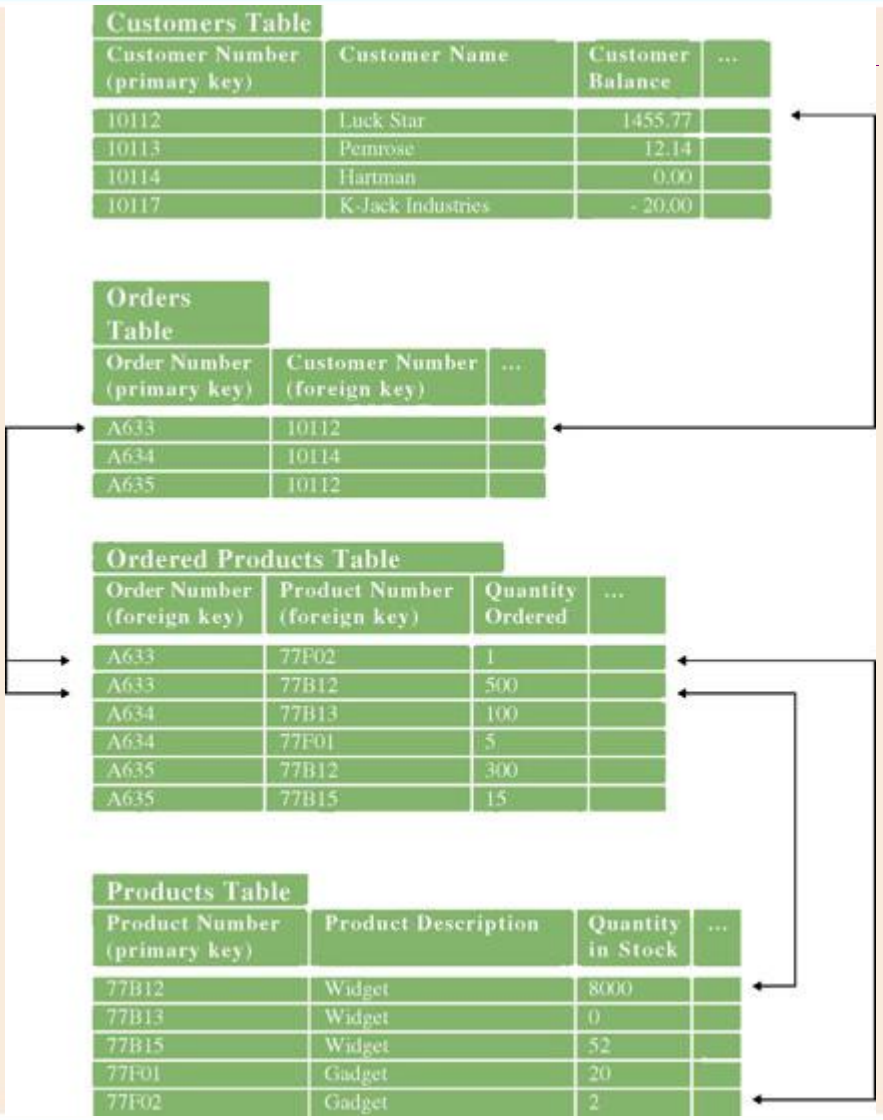
Relational database – a database that implements stored data in a series of two-dimensional tables that are "related" to one another via foreign keys.

- The physical data model is called a **schema**.
- The DDL and DML for a relational database is called **SQL** (Structured Query Language).
- **Triggers** – programs embedded within a database that are automatically invoked by updates.
- **Stored procedures** – programs embedded within a database that can be called from an application program.

From Logical Data Model …



… To Physical Data Model (Relational Schema)



Customers Table

Customer Number (primary key)	Customer Name	Customer Balance	…
10112	Luck Star	1455.77	
10113	Penrose	12.14	
10114	Hartman	0.00	
10117	K-Jack Industries	- 20.00	

Orders Table

Order Number (primary key)	Customer Number (foreign key)	…
A633	10112	
A634	10114	
A635	10112	

Ordered Products Table

Order Number (foreign key)	Product Number (foreign key)	Quantity Ordered	…
A633	77F02	1	
A633	77B12	500	
A634	77B13	100	
A634	77F01	5	
A635	77B12	300	
A635	77B15	15	

Products Table

Product Number (primary key)	Product Description	Quantity in Stock	…
77B12	Widget	8000	
77B13	Widget	0	
77B15	Widget	52	
77F01	Gadget	20	
77F02	Gadget	2	

User Interface for a Relational PC DBMS

What is a Good Data Model?

- A good data model is simple
  - The data attributes that describe an entity should describe only that entity
- A good data model is essentially nonredundant
  - Each data attribute exists in at most one entity (except for foreign keys)
- A good data model should be flexible and adaptable to future needs

These goals are achieved through database normalization.

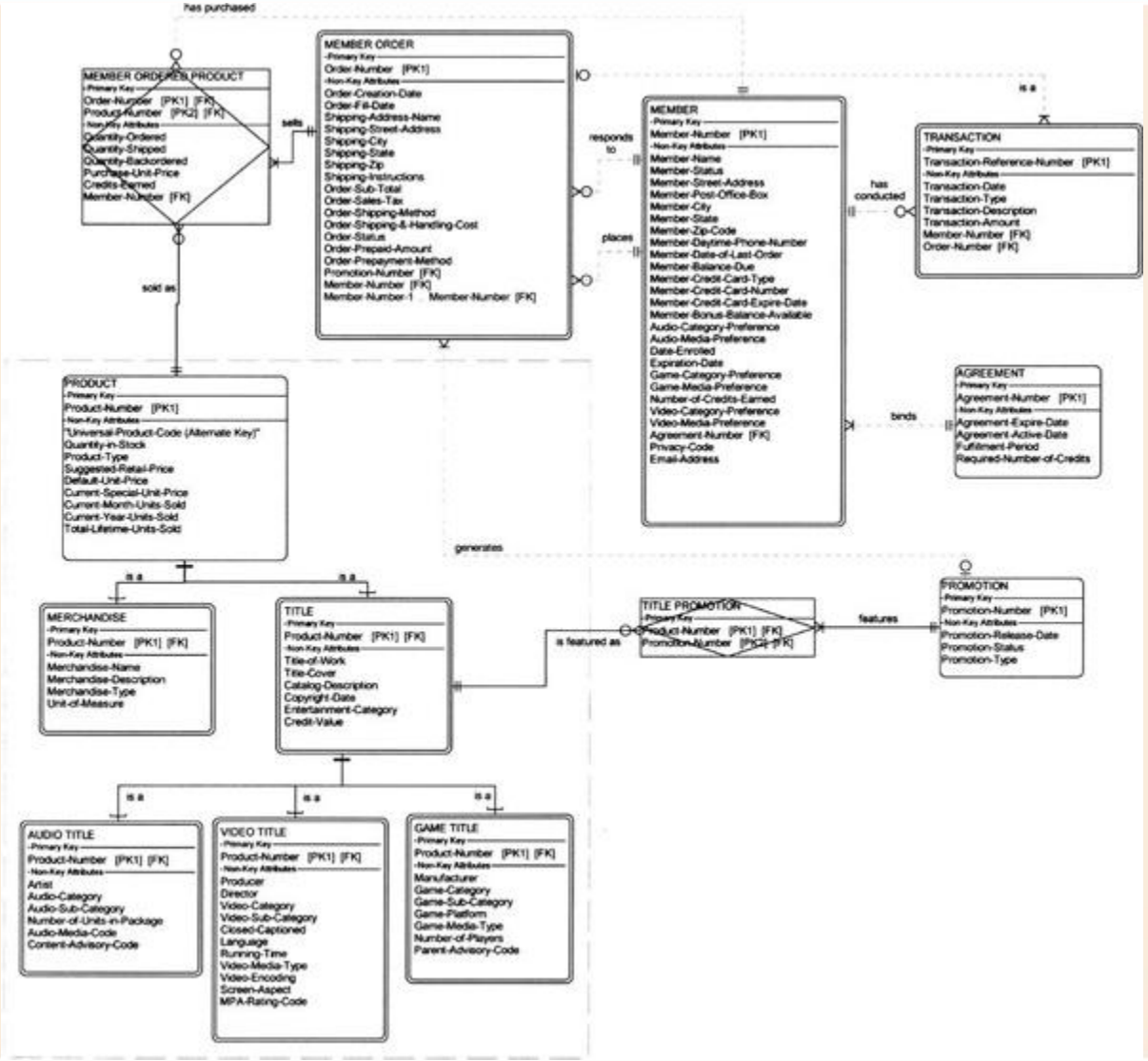
Database Normalization (also see Chapter 7)

- A logical entity (or physical table) is in **first normal form** if there are no attributes (fields) that can have more than one value for a single instance (record).
- A logical entity (or physical table) is in **second normal form** if it is in first normal form and if the values of all nonprimary key attributes are dependent on the full primary key.
- A logical entity (or physical table) is in **third normal form** if it is in second normal form and if the values of all nonprimary key attributes are not dependent on other nonprimary key attributes.

Goals of Database Design

- A database should provide for efficient storage, update, and retrieval of data.
- A database should be reliable—the stored data should have high integrity and promote user trust in that data.
- A database should be adaptable and scalable to new and unforeseen requirements and applications.
- A database should support the business requirements of the information system.

Logical data Model in Third Normal Form



Database Schema

- Database schema – a model or blueprint representing the technical implementation of the database.
  - Also called a physical data model

A Method for Database Design

1. Review the logical data model.
2. Create a table for each entity.
3. Create fields for each attribute.
4. Create index for each primary & secondary key.
5. Create index for each subsetting criterion.
6. Designate foreign keys for relationships.
7. Define data types, sizes, null settings, domains, and defaults for each attribute.
8. Create or combine tables to implement supertype/subtype structures.
9. Evaluate/specify referential integrity constraints.

Database Integrity

- **Key integrity** – Every table should have a primary key.
- **Domain integrity** – Appropriate controls must be designed to ensure that no field takes on an inappropriate value
- **Referential integrity** – the assurance that a foreign key value in one table has a matching primary key value in the related table.
  - No restriction
  - Delete: cascade
  - Delete: restrict
  - Delete: set null

Data Types for Different Database Technologies

Logical Data Type to be stored in field)	**Physical Data Type MS *Access***	**Physical Data Type MS *SQL Server***	**Physical Data Type Oracle**
Fixed length character data *(use for fields with relatively fixed length character data)*	TEXT	CHAR (size) *or* character (size)	CHAR (size)
Variable length character data *(use for fields that require character data but for which size varies greatly--such as ADDRESS)*	TEXT	VARCHAR (max size) *or* character varying (max size)	VARCHAR (max size)
Very long character data *(use for long descriptions and notes--usually no more than one such field per record)*	MEMO	TEXT	LONG VARCHAR *or* LONG VARCHAR2

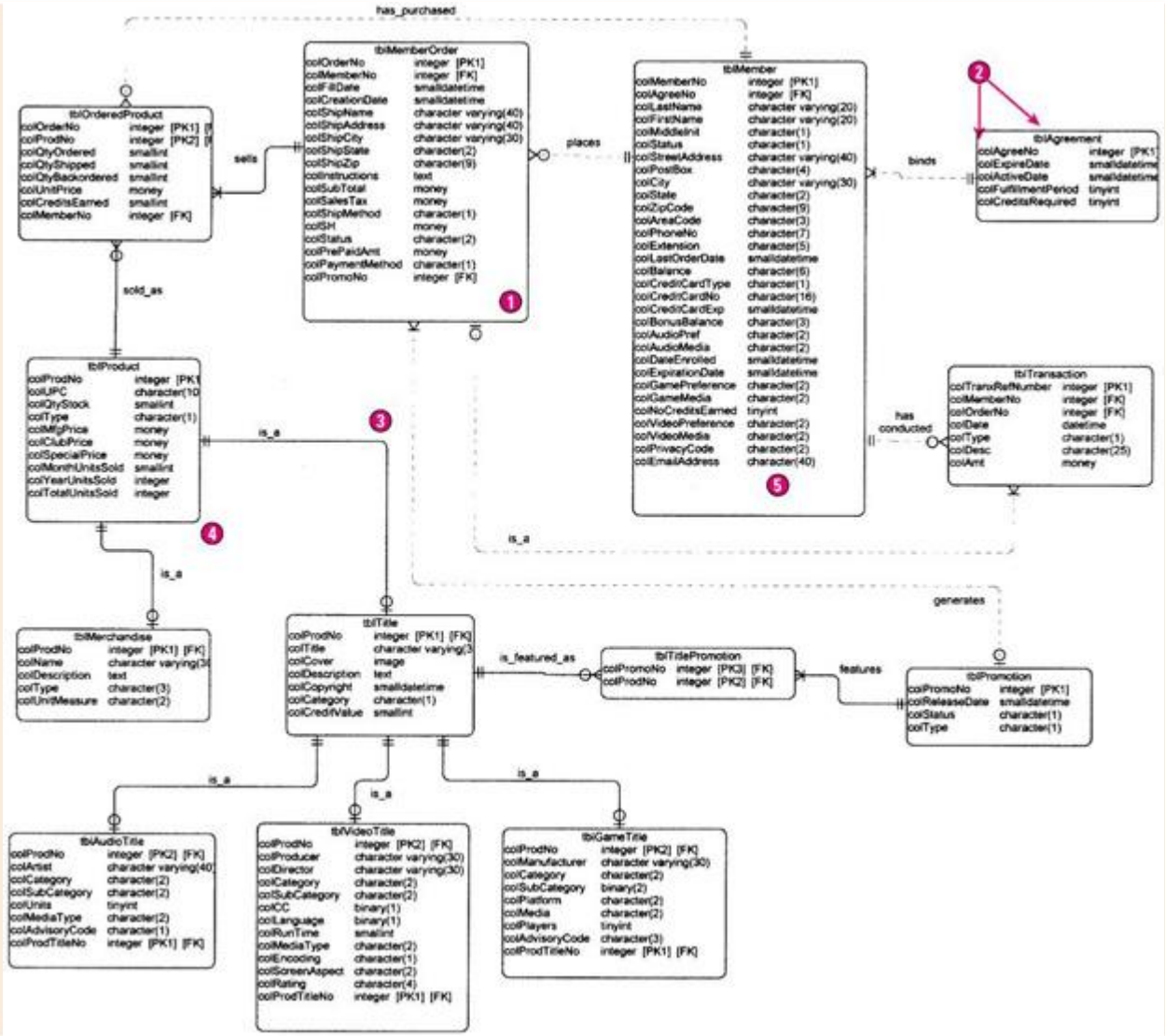
Data Types for Different Database Technologies (cont.)

Logical Data Type to be stored in field)	**Physical Data Type MS *Access***	**Physical Data Type MS *SQL Server***	**Physical Data Type Oracle**
Integer number	NUMBER	INT (size) *or* integer *or* smallinteger *or* tinuinteger	INTEGER (size) *or* NUMBER (size)
Decimal number	NUMBER	DECIMAL (size, decimal places) *or* NUMERIC (size, decimal places)	DECIMAL (size, decimal places) *or* NUMERIC (size, decimal places) *or* NUMBER
Financial Number	CURRENCY	MONEY	*see decimal number*
Date (with time)	DATE/TIME	DATETIME *or* SMALLDATETIME *Depending on precision needed*	DATE
Current time *(use to store the data and time from the computer's system clock)*	*not supported*	TIMESTAMP	*not supported*

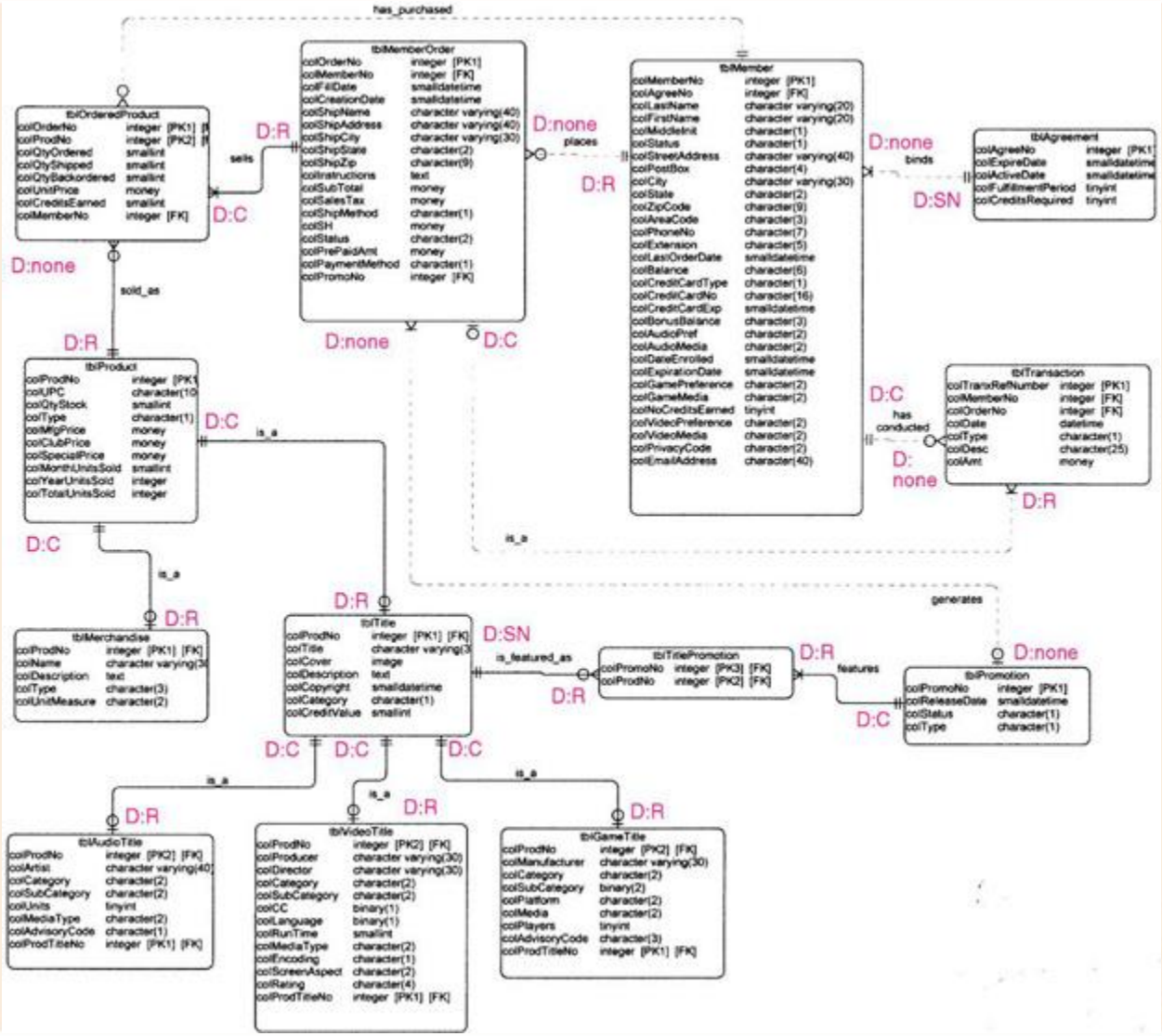
Data Types for Different Database Technologies (cont.)

Logical Data Type to be stored in field)	**Physical Data Type MS *Access***	**Physical Data Type MS *SQL Server***	**Physical Data Type Oracle**
Yes or No; *or* True or False	YES/NO	BIT	*use CHAR(1) and set a yes or no domain*
Image	OLE OBJECT	IMAGE	LONGRAW
Hyperlink	HYPERLINK	VARBINARY	RAW
Can designer define new data types?	NO	YES	YES

Physical Database Schema



Database Schema with Referential Integrity Constraints



Database Distribution and Replication

Data distribution analysis establishes which business locations need access to which logical data entities and attributes.

Database Distribution and Replication (continued)

- **Centralization**
  - Entire database on a single server in one physical location
- **Horizontal distribution (also called partitioning)**
  - Tables or row assigned to different database servers/locations.
  - Efficient access and security
  - Cannot always be easily recombined for management analysis
- **Vertical distribution (also called partitioning)**
  - Specific table columns assigned to specific databases/servers
  - Similar advantages and disadvantages of Horizontal
- **Replication**
  - Data duplicated in multiple locations
  - DBMS coordinates updates and synchronization
  - Performance and accessibility advantages
  - Increases complexity

Database Capacity Planning

- For each table sum the *field sizes*. This is the *record size*.
- For each table, multiply the *record size* times the number of entity instances to be included in the table (planning for growth). This is the *table size*.
- Sum the *table sizes*. This is the *database size*.
- Optionally, add a slack capacity buffer (e.g. 10percent) to account for unanticipated factors. This is the *anticipated database capacity*.

SQL DDL Code

```
CREATE TABLE [dbo].[ClassCodes] (
        [ClassID] [Integer] Identity(1,1) NOT NULL,
        [DepartmentCodeID] [varchar] (3) NOT NULL ,
        [SectionCodeID] [varchar] (2) NOT NULL ,
        [ClassCodeID] [varchar] (5) NOT NULL ,
        [GroupCodeID] [varchar] (1) NOT NULL ,
        [ClassDescription] [varchar] (50) NOT NULL ,
        [ValidOnLine] bit NULL ,
        [LastUpdated] [smalldatetime] NULL
) ON [PRIMARY]
GO

Alter Table [dbo].[ClassCodes] Add Constraint pk_classcodes
        Primary Key (ClassID)
Alter Table [dbo].[ClassCodes] Add Constraint df_classcodes_groupcodeid
        Default 'A' for GroupCodeID
Alter Table [dbo].[ClassCodes] Add Constraint fk_classcodes_sectioncodes
        Foreign Key (DepartmentCodeID,SectionCodeID)
        References SectionCodes(DepartmentCodeID,SectionCodeID)
Alter Table [dbo].[ClassCodes] Add Constraint
un_classcodes_Dept_Section_Class
        Unique (DepartmentCodeID,SectionCodeID,ClassCodeID)
GO
```