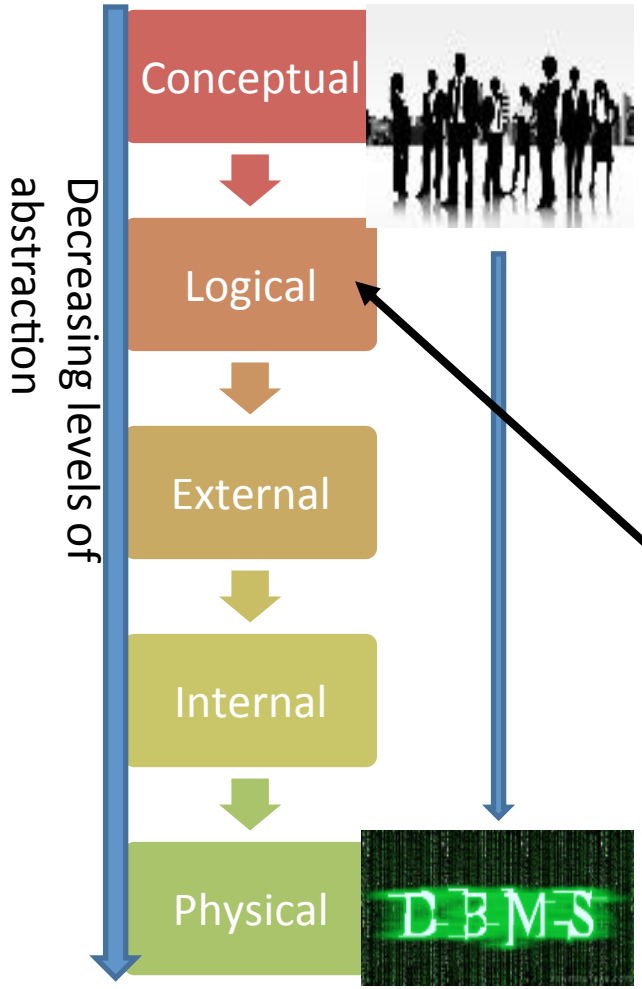
Logical Modeling

Levels of Data Model Abstraction



Logical Model

- Describes the business data and the rules for governing their use and storage
- Identifies the things that are important for an organization to store
- Provides a framework for communicating data needs to and from business users

What Exactly Is a Logical Model?

- It is a blueprint of your database.
- It is DBMS ***independent,*** and does not rely on a specific product.
- It is a technical communications tool.
- Concepts from relational theory are re-introduced (PK, FK, table, column, data type)
- Any person knowledgeable in SQL should be able to read a logical model and create a real database implementation from it.

Concept Map: Conceptual vs. Logical

CONCEPTUAL	LOGICAL
ERD	Relational diagram
Entity	Relation (Table)
Attribute	Column
Relationship	Foreign key (FK)
N/A	PK, constraints

Important: Rows (sample data) MUST be considered in the logical model!

Three Steps of Mapping

1. ***Entity mapping.*** Add PK, create table.
2. ***Attribute mapping. A***dd columns, break down composites, follow naming conventions!
3. ***Relationship mapping.*** Place FK.

Basic Entity and Attribute Mapping

Conceptual

Customer
Name Email Phone

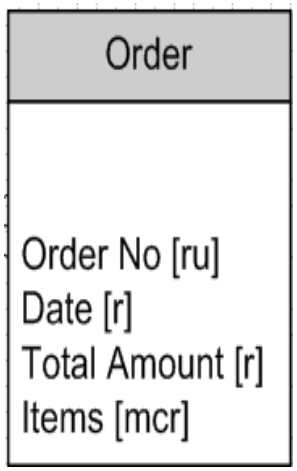
Logical

Customer		
PK	**CustomerID**	**N-Signed Integer**
	FirstName	**C-Fixed Length(50)**
	LastName	**C-Fixed Length(50)**
	CompanyName	C-Fixed Length(50)
U1	**Email**	**C-Fixed Length(75)**
	Phone	**C-Fixed Length(15)**

- Use naming conventions.
- Break down composite into simple.
- Assign data types.
- Select PK for each table.
- Set required to not null.
- Set unique and check constraints.

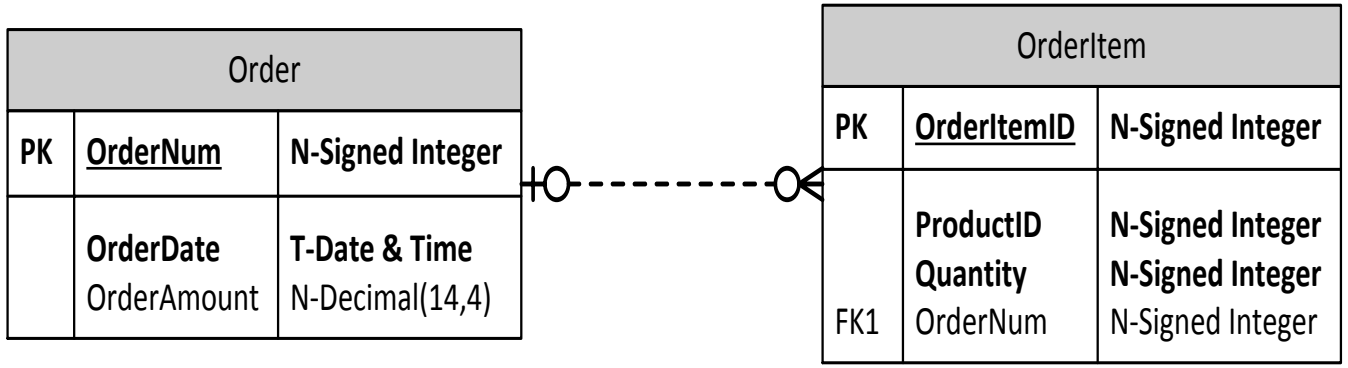
Mapping Multivalued Attributes

Conceptual

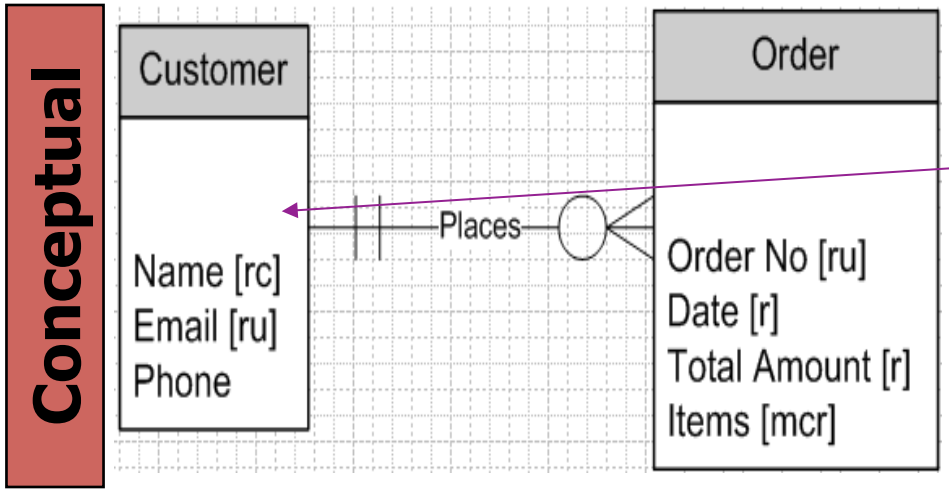


- Similar to 1-M: Place [m] attributes in new table. PK from original table is FK in the new table.

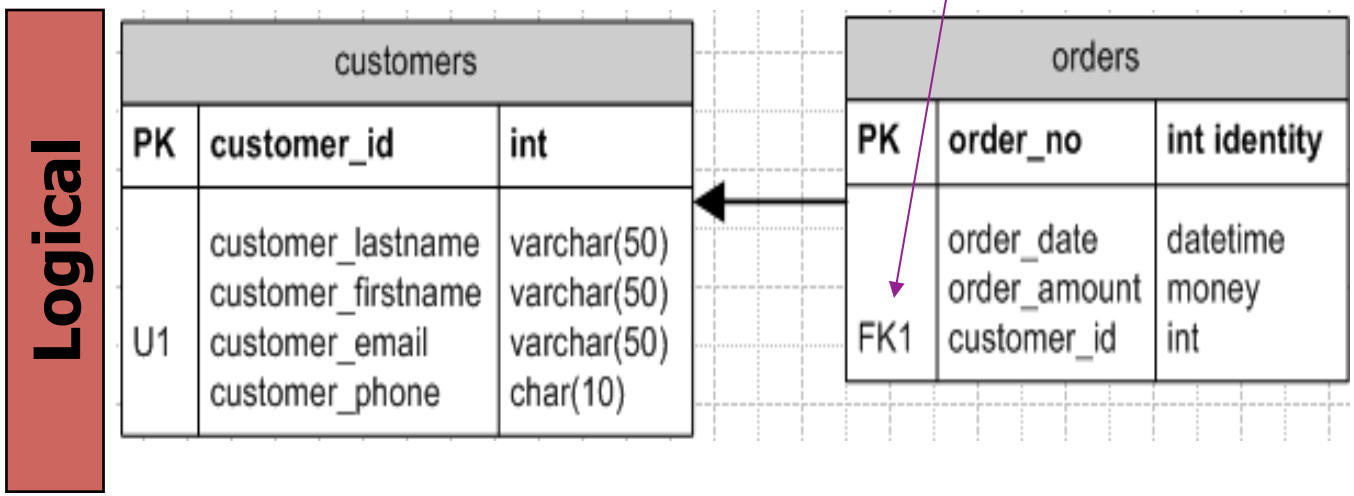
Logical



Mapping 1-M Relationships

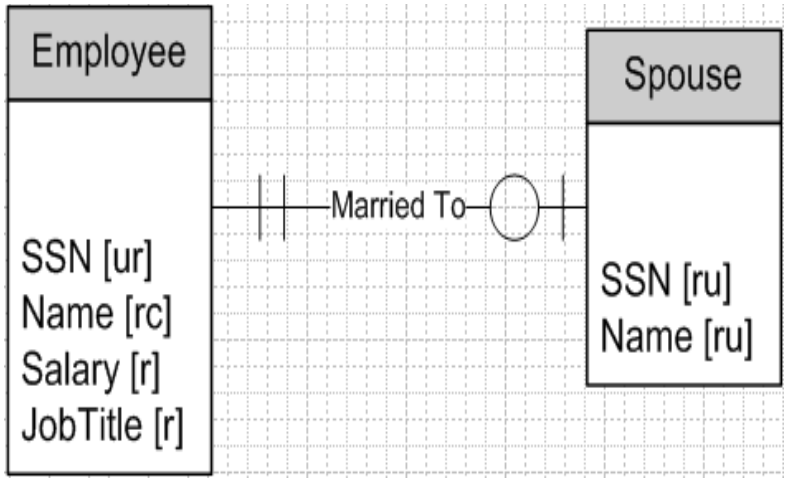


- FK goes on the many side.
- If required on the one side (a.k.a. weak entity), then set the FK to not allow null.



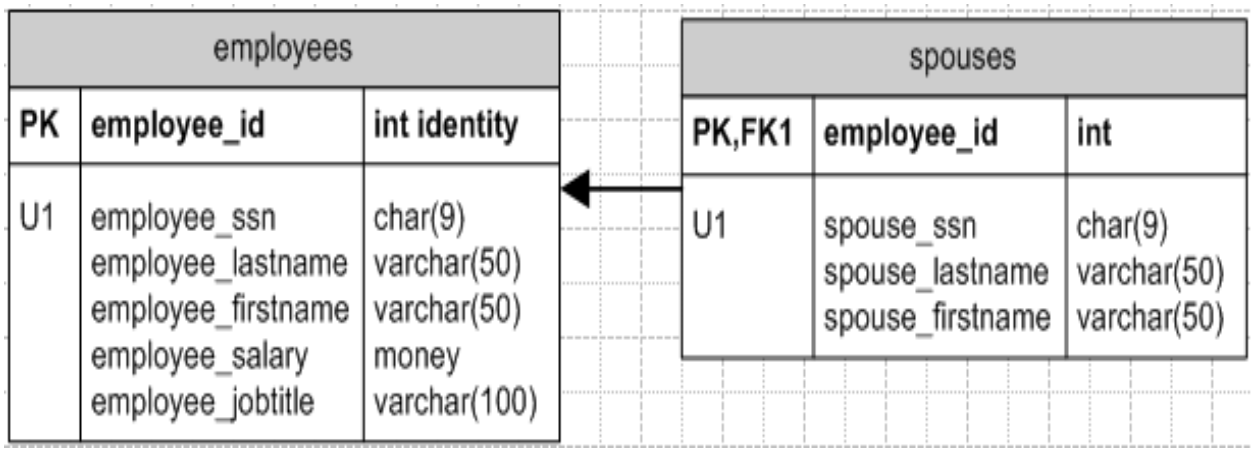
Mapping 1-1 Relationships

Conceptual



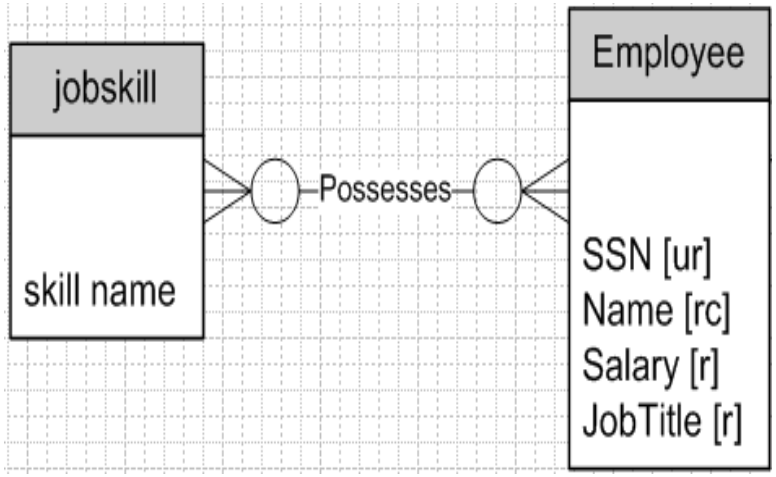
- Similar to 1-M: Since the FK goes to the optional side of the relationship, but the FK is set to PK.

Logical



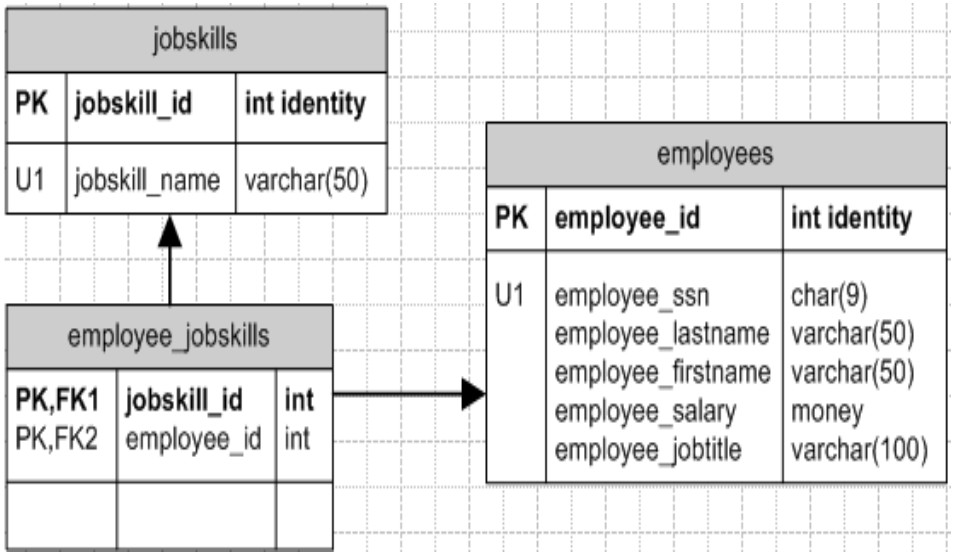
Mapping M-M Relationships

Conceptual



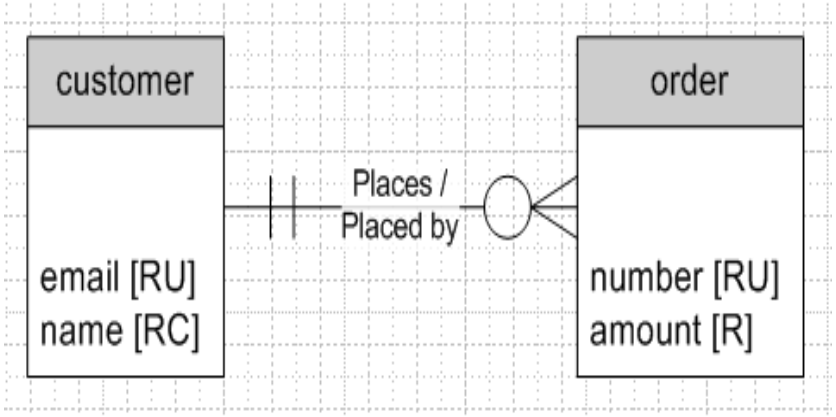
- Make a "bridge table"; place FKs in bridge table and set as composite PK.

Logical



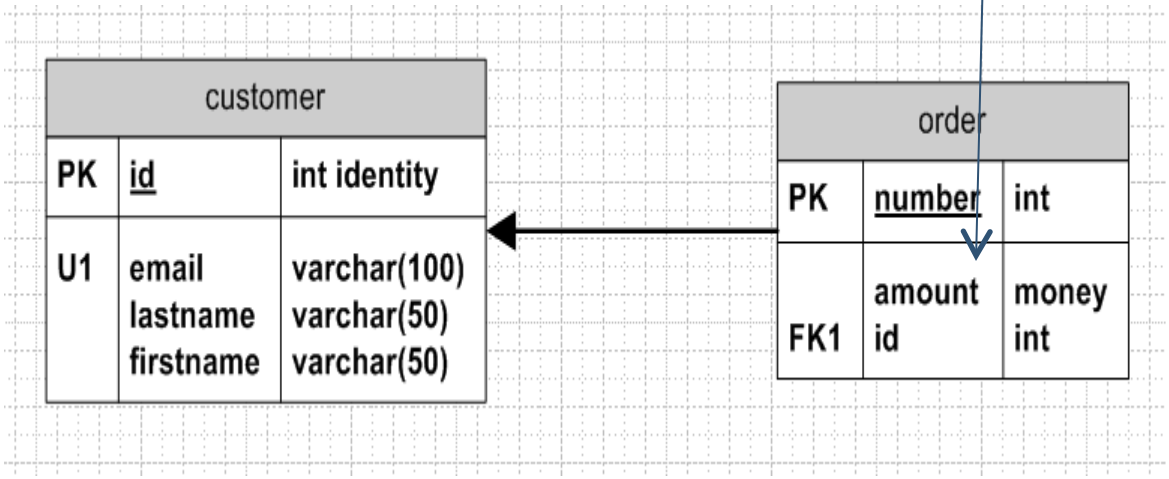
Mapping Weak Entities

Conceptual

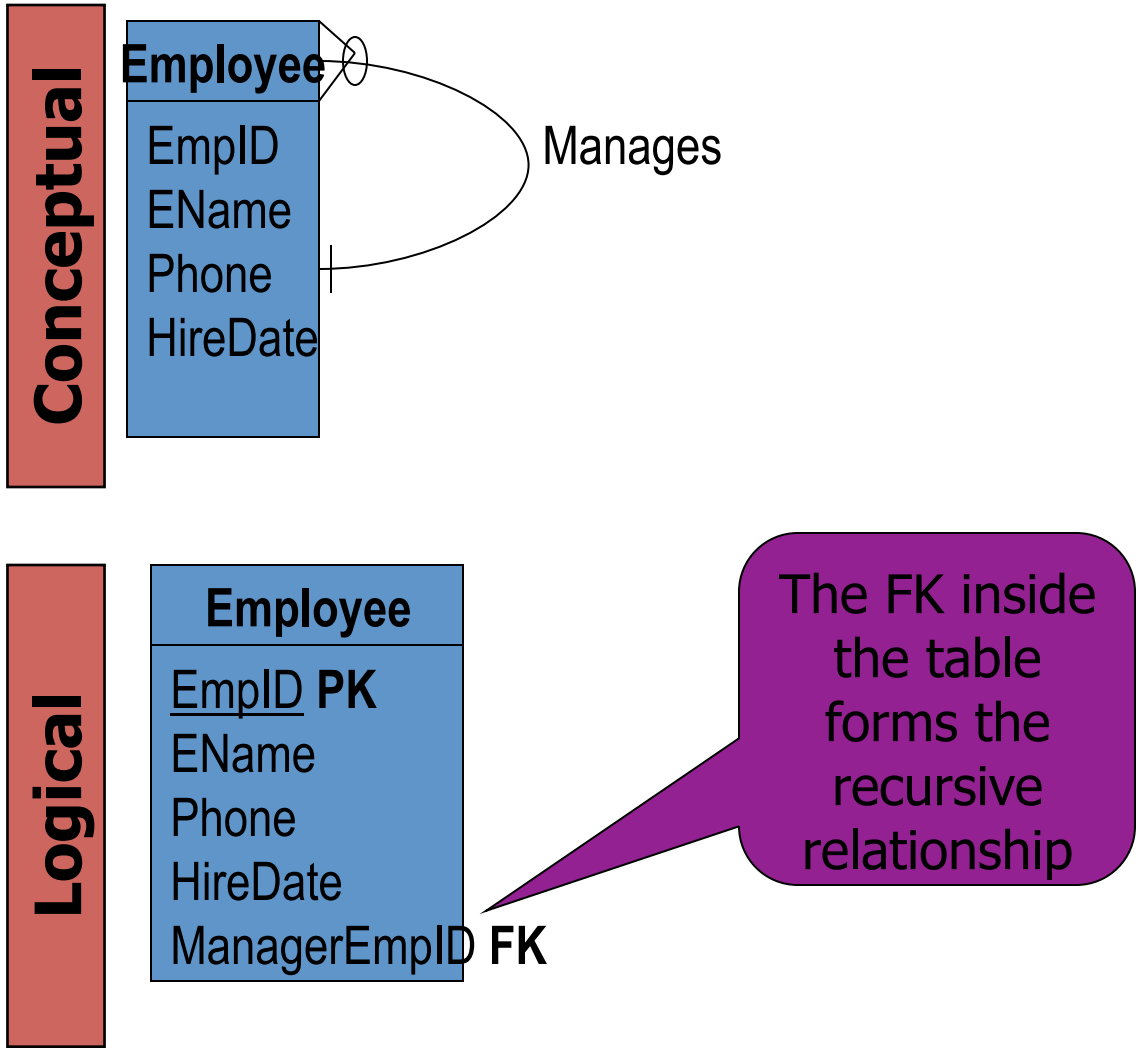


- Make sure the foreign key does not allow null.

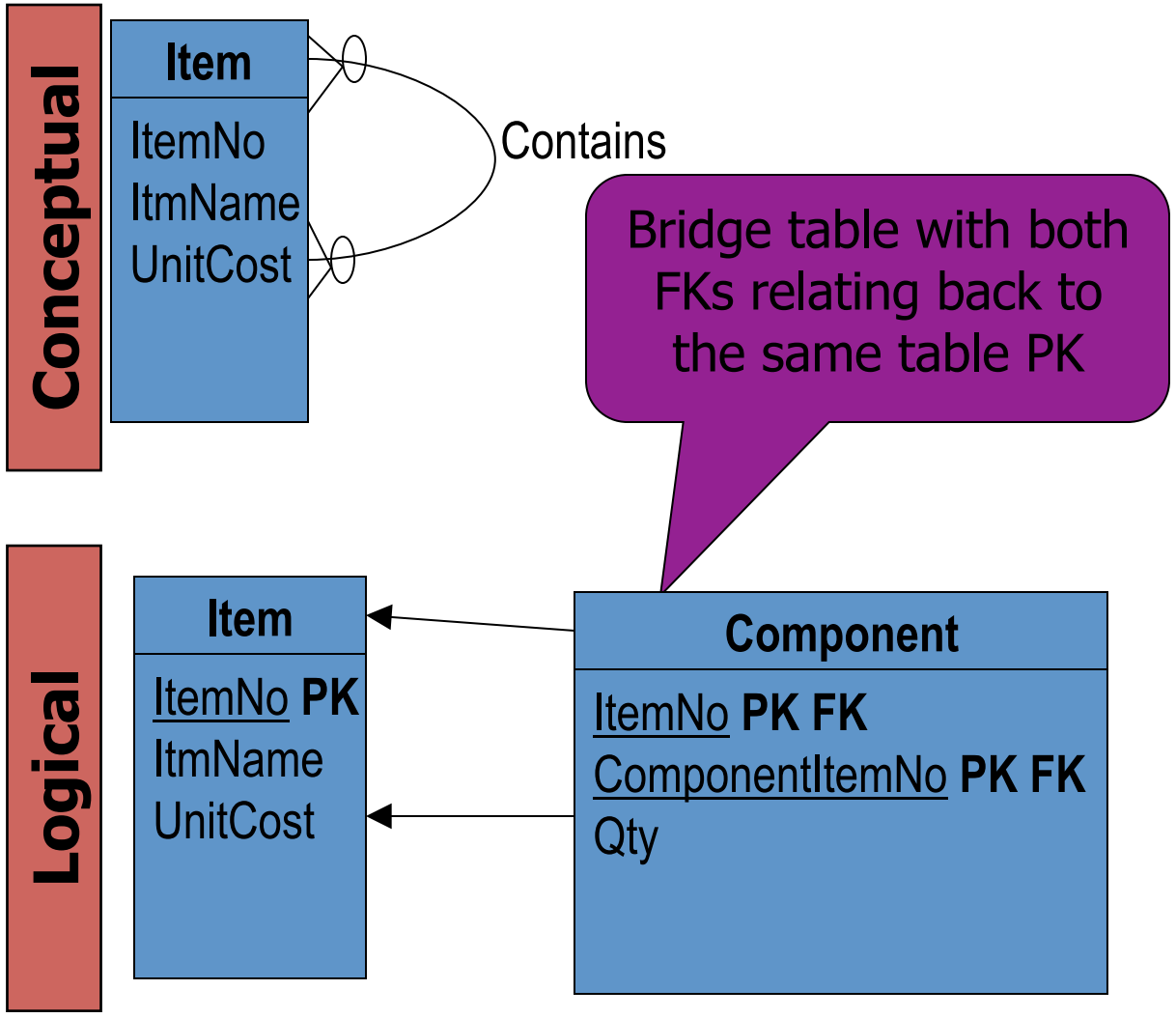
Logical



Example: Mapping a Unary 1-M Relationship



Example: Mapping a Unary N-M Relationship



The Relational Data Model

Relational Data Model

- Based on relational algebra
- Codd, E. F. (1970, June). A relational model of data for large shared data banks. *Communications of the ACM, 13*(6), 77–87.
- Three components:
  - Data structure
  - Data manipulation
  - Data integrity

Relations

- Relations are named, two-dimensional tables of data
- Attributes are named columns of relations
- Example:
  Patron(CardNumber, Name, Address, Phone, CardUsers, Email)

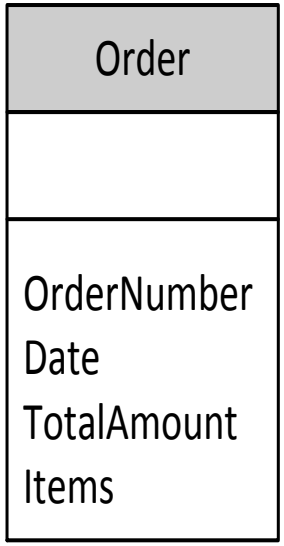
Properties of Relations

- Each relation must have a unique name.
- No multivalue columns (create new relations).
- Each row must be unique.
- Each column name must be unique within the relation.
- Sequence of rows and columns is insignificant.

Relational Keys

- Primary keys uniquely identify an entity instance in the set of all instances of an entity.

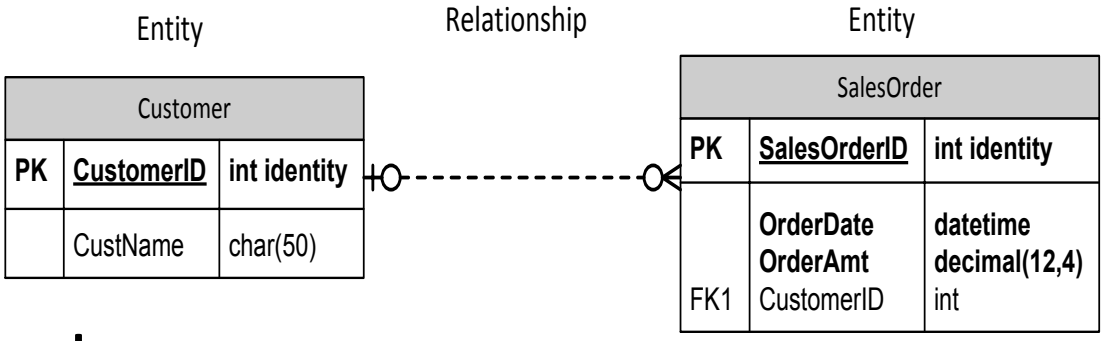
- Example:
  Order(OrderNumber, Date, TotalAmount)
  OrderItem(OrderNumber, ItemNumber, Qty)



- Note: The primary key may be more than one column. This is called a “composite key.”

Relational Keys

- Foreign keys are attributes in a relation that serve as primary keys in other relations.



- Example:
  Customer(CustomerID, CustName)
  SalesOrder(SalesOrderID, OrderDate, OrderAmt, CustomerID)

Integrity Constraints

Integrity Constraints

- Rules limiting acceptable values and actions

- Domain constraints
  - All values in a column of a relation must be from the same domain (name, meaning, data type, size, allowable values or range).
- Entity integrity
  - Every relation has a primary key and all primary key values are valid and are not NULL.

Null Values

- Null values are the absence of a value.
  - Codd's rule #3: A relational database must support a representation of missing information
  - Called NULL in SQL Server (and others)
  - Not zero (0)
  - Not an empty string
  - Literally means nothing, empty space, a vacuum
  - Cannot be compared using equivalence
  - Can cause problems with query results
  - Can be avoided using constraints (if problem domain requires it)

The Primary Key

- A column used to uniquely identify a row (tuple) in the table (relation)
- Chad's rules for primary keys
  - Must be unique and not NULL
  - Must be simple*
    - Made from only one attribute. If no single field that uniquely identifies the row exists, create one. This field is called a "surrogate key."
    - A composite key is a key constraint made from more than one field. No composite keys in the physical database. Ever.
  - Must be immutable
    - Once a primary key value is defined, it cannot be changed. If a primary key must be changed to suit a domain purpose, a new record must be created.
  - Must make no business sense*
    - Because data that make sense in the problem domain are all subject to change and the primary key is for the system's use for record identification, the primary key should map to no real-world analogue.

** These are not rules enforced by any DBMS but are instead best practices culled from years of working with terribly selected primary keys and poorly designed database models.*

Good PK . . . ***Bad*** PK

Good candidate key choices?

- Customer Name?
- Email Address?
- Phone Number?
- Name and DOB?
- Social Security Number?
- Customer Selected Value?
- Random Unique #?
- Sequential Unique #?
- Last two are examples of ***surrogate*** keys

The **Foreign Key** Constraint

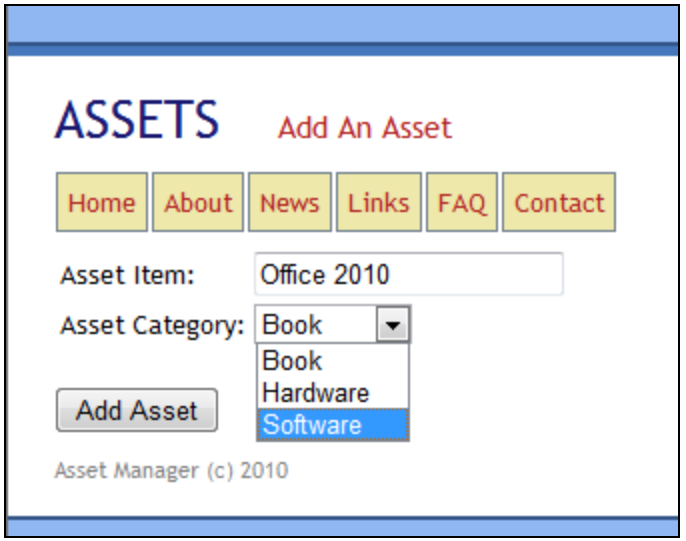
- The ***foreign key*** is a constraint on a column of one relation (table) so that it can be associated with another relation (table).
- Foreign keys must have **referential integrity**—their values must come from the corresponding PK column in the relation.
- The value entered references a primary key field in another table (think lookups), so the value entered must exist in that other table. (The customer we've chosen for our order must actually be a customer.)

Example of FK: The Lookup Table

Asset		
Item	**Category**	**Asset#**
The ABC's of Excel	Book	1
InFocus Projector	Hardware	2
Prog. for Dummies	Book	3
Learning Perl	Book	4
Dell Laptop #1	Hardware	5
Windows 2000 Server	Software	6
Office 2000 Premium	Software	7
Dell Laptop #2	Hardware	8

CategoryID
Book
Hardware
Software

Foreign Key



Let's Talk Primary and Foreign Keys

Group ID	Group Name	Num of Members	Country of Origin
123	One Republic	5	America
456	Train	3	US
222	Linkin Park	6	USA
123	Rolling Stones	4	England
333	Muse	3	England
777	One Direction	5	London
789	Bruno Mars	NA	USA
101	Neon Trees	4	USA

Lookup Tables and Foreign Keys

Group ID	Group Name	Num of Members	Country of Origin
123	One Republic	5	America
456	Train	3	US
222	Linkin Park	6	USA
123	Rolling Stones	4	England
333	Muse	3	England
777	One Direction	5	London
789	Bruno Mars	NA	USA
101	Neon Trees	4	USA

Country
US
England
France

The Natural Join of PK-FK at Work

2:Data in Table 'fudgemart_employee' in 'mafudge45' on '(local)'

emp_id	emp_lname	emp_fname	emp_dept	emp_store_num
8	Mary	Mi	Electronics	101
9	Photo	Arial	Hardware	101
10	Shores	Sonny	Customer Service	103
11	Nugget	Chris P.	Sporting Goods	103
12	Frienzergon	Alma	Sporting Goods	103
13	Furniture	Patty O.	Electronics	103
15	Enweave	Bob	Customer Service	104
16	Hvmeehom	Lee	Sporting Goods	104
17	Ladd	Sal	Sporting Goods	104
18	Donalds	Mick	Hardware	104
19	Work	Willie	Electronics	104
20	Case	Justin	Electronics	104
21	Fudge	Miquel	Customer Service	101
22	Fudge	Michelle	Customer Service	101
25	Fudge	Mike	Sporting Goods	101

3:Data in Table 'fudgemart_store' in 'mafudge...

store_num	store_zipcd	store_region
101	90211	West
102	13039	East
103	53401	South
104	33409	Midwest
105	99202	West

Natural Join

emp_id	emp_lname	emp_fname	emp_dept	emp_store_num	store_num	store_zipcd	store_region
8	Mary	Mi	Electronics	101	101	90211	West
9	Photo	Arial	Hardware	101	101	90211	West
10	Shores	Sonny	Customer Service	103	103	53401	South
11	Nugget	Chris P.	Sporting Goods	103	103	53401	South
12	Frienzergon	Alma	Sporting Goods	103	103	53401	South
13	Furniture	Patty O.	Electronics	103	103	53401	South
15	Enweave	Bob	Customer Service	104	104	33409	Midwest
16	Hvmeehom	Lee	Sporting Goods	104	104	33409	Midwest
17	Ladd	Sal	Sporting Goods	104	104	33409	Midwest
18	Donalds	Mick	Hardware	104	104	33409	Midwest
19	Work	Willie	Electronics	104	104	33409	Midwest
20	Case	Justin	Electronics	104	104	33409	Midwest
21	Fudge	Miquel	Customer Service	101	101	90211	West
22	Fudge	Michelle	Customer Service	101	101	90211	West
25	Fudge	Mike	Sporting Goods	101	101	90211	West

Other Constraints

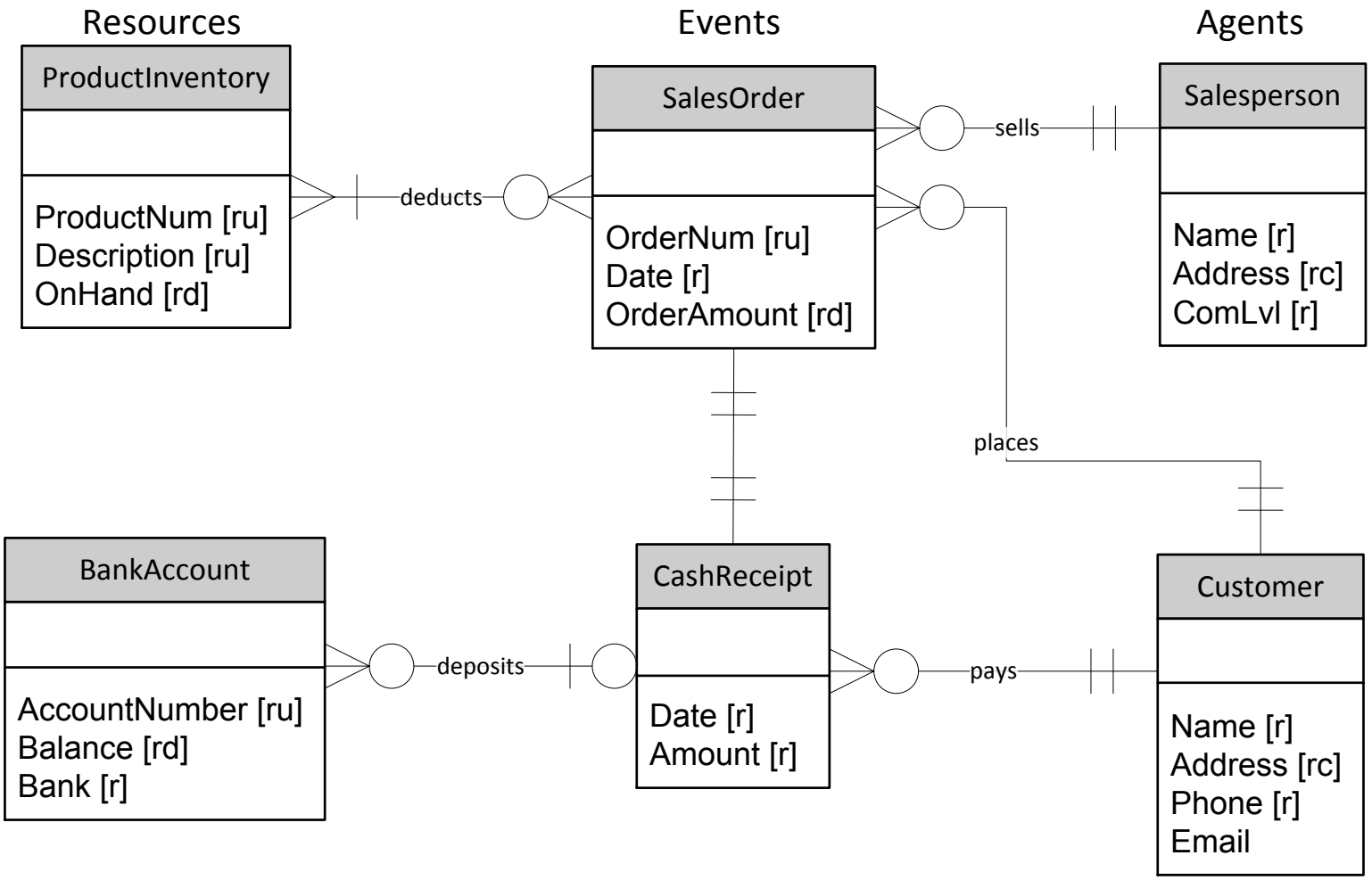
- Default value
  - Defaults an attribute to a value when one is not specified by the user
- Check
  - Evaluates whether a provided value conforms to the business rules regarding the attribute before inserting the row (e.g., number of children cannot be negative; a person is either alive, dead, or mostly dead, etc.)
- Uniqueness
  - The value entered MUST be unique for this column in this table. (There can be only one Highlander.)

Transforming ER Diagrams Into Relations

Mapping Entities

Entities and Relationships

- Another approach to thinking about entities and their relationships
- All data we track revolves around three type of entities:
  - **R**esources—These are the business *things* that are affected (e.g., inventory, bank accounts).
  - **E**vents—These are the happenings that affect the resources (e.g., sales order, purchase order).
  - **A**gents—These are the people or systems that participate in the event (e.g., customer, vendor, salesperson).
- This is known as "REA."
- Look in the data and narratives for the resources, events, and agents.


Resources
Events
Agents
ProductInventory
ProductNum [ru]
Description [ru]
OnHand [rd]
deducts
SalesOrder
OrderNum [ru]
Date [r]
OrderAmount [rd]
sells
Salesperson
Name [r]
Address [rc]
ComLvl [r]
places
BankAccount
AccountNumber [ru]
Balance [rd]
Bank [r]
deposits
CashReceipt
Date [r]
Amount [r]
pays
Customer
Name [r]
Address [rc]
Phone [r]
Email

Data Types

Data Types

- Define the kind of data to be stored
  - Numeric
  - Text
  - Dates and times

Numeric Data Types

- Integer data types store whole numbers.
- Decimal data types store fixed precision decimal values.
- Real data types store floating-point numbers with a limited number of significant digits. Values are only approximate.

Integer Data Types

Type	Bytes	Description
bigint	8	Integers from -9,223,372,036,854,775,808 through 9,223,372,036,854,775,807
int	4	Integers from -2,147,483,648 to 2,147,483,647
smallint	2	Integers from -32,768 to 32,767
tinyint	1	Unsigned integers from 0 to 255
bit	1	Integers with a value of 1 or 0. Used to represent Boolean TRUE or FALSE, respectively

Decimal Data Types

Type	Bytes	Description
decimal(p, s)	5-17	Fixed precision decimal number. p represents the total number of significant digits. s represents which position, when counted from the left, has the decimal place. i.e., decimal(5, 3) would yield numbers like 999.99
numeric(p, s)	5-17	See decimal
money	8	Same as decimal(19, 4)

Real Data Types

Type	Bytes	Description
float(n)	4 or 8	Double precision floating point numbers from -1.79 x10^{308} through 1.79x10^{308} n represents the mantissa.
real	4	Synonymous with float(34)

Text Data Types

- Store strings of characters
- Can be fixed in length or variable
- Can store just ASCII or Unicode characters

Text Data Types

Type	Bytes	Description
char(n)	n	Fixed-length ASCII character data (1 <= n >= 8000)
varchar(n)	*	Variable length ASCII character data (1 <= n >= 8000)
nchar(n)	2n	Fixed length Unicode character data (1 <= n >= 8000)
nvarchar(n)	*	Variable length Unicode character data (1 <= n >= 8000)

** Total bytes required depends on the actual contents*

Date/Time Data Types

- Stored on disk as decimal numbers
- Provided to allow for date/time-based mathematics

Date/Time Data Types

Type	Bytes	Description
datetime	8	Dates and times from Jan 1, 1753, to Dec 31, 9999 (accuracy to 3.33 milliseconds)
smalldatetime	4	Dates and times from Jan 1, 1900, to June 6, 2079 (accuracy to 1 minute)
date	3	Dates from Jan 1, 0001, through Dec 31, 9999
time(n)	3-5	Times from 00:00:00.0000000 through 23:59:59.9999999. n is used to adjust the precision from 0 to 7.
datetime2(n)	6-8	Dates from Jan 1, 0001, through Dec 31, 9999, and times from 00:00:00.0000000 through 23:59:59.9999999
datetimeoffset(n)	8-10	Extends datetime2 by including a time zone offset


SYRACUSE UNIVERSITY
SUOS
CULTORES
SCIENTIA
CORONAT