Basic Data Modeling

Slides adapted from Prof. Heimann CMU-P

Why Data Modeling?

Why Data Modeling?

- A bridge to convert requirements into a database

Why Data Modeling?

- A bridge to convert requirements into a database
- Can be done early in the process

Why Data Modeling?

- A bridge to convert requirements into a database
- Can be done early in the process
- Cheaper to fix errors at this stage

Why Data Modeling?

- A bridge to convert requirements into a database
- Can be done early in the process
- Cheaper to fix errors at this stage
- Understandable to users and developers

Why Data Modeling?

- A bridge to convert requirements into a database
- Can be done early in the process
- Cheaper to fix errors at this stage
- Understandable to users and developers
- Data is critical!

Why Data Modeling?

- A bridge to convert requirements into a database
- Can be done early in the process
- Cheaper to fix errors at this stage
- Understandable to users and developers
- Data is critical!
- Entity-Relationship modeling is fairly easy to do

What Makes a Good Data Model?

- Completeness - Does the model support all the necessary data?
- Nonredundancy - Does the model specify a database in which the same fact can be recorded more than once?
- Enforcement of Business Rules - How accurately does the model reflect and enforce rules that apply to the business data?
- Data Reusability - Will the data stored in the database be reusable for purposes beyond those initially anticipated?
- Stability and Flexibility - How well will the model cope with possible changes to business requirements?
- Elegance - Does the data model provide a reasonably neat and simple classification of the data?

What kind of Datamodels?

- Conceptual Data Model - Technology independent specification of data to be held in the database. Focus is on communication between the data modeler and business stakeholders.
- Logical Data Model - Translation of the conceptual models into structures useable by DBMS (tables and columns).
- Physical Data Model - Deal with performance, physical storage, and access mechanisms.

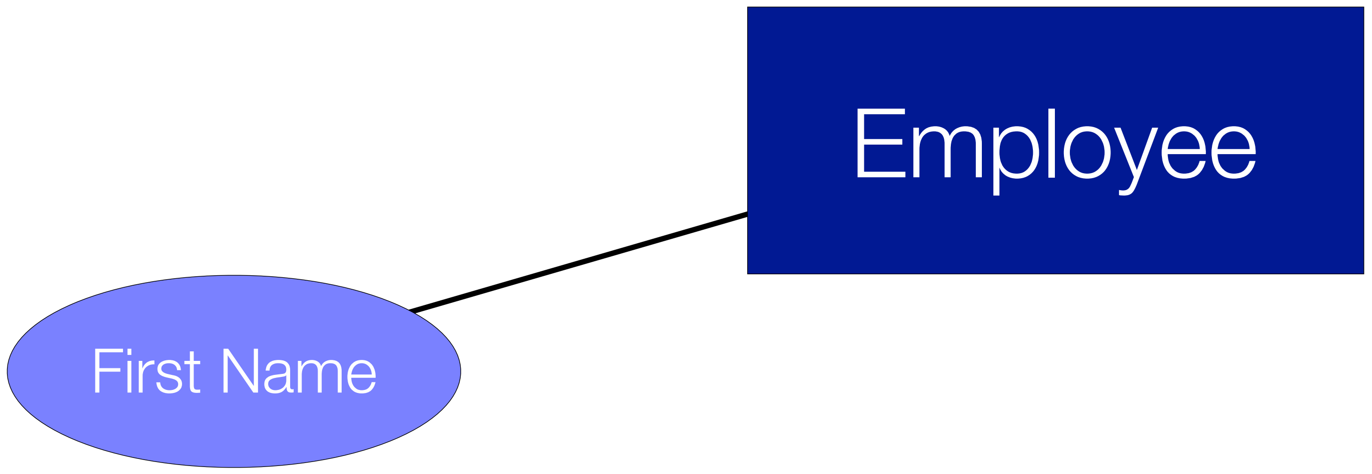
What is an entity?

What is an entity?

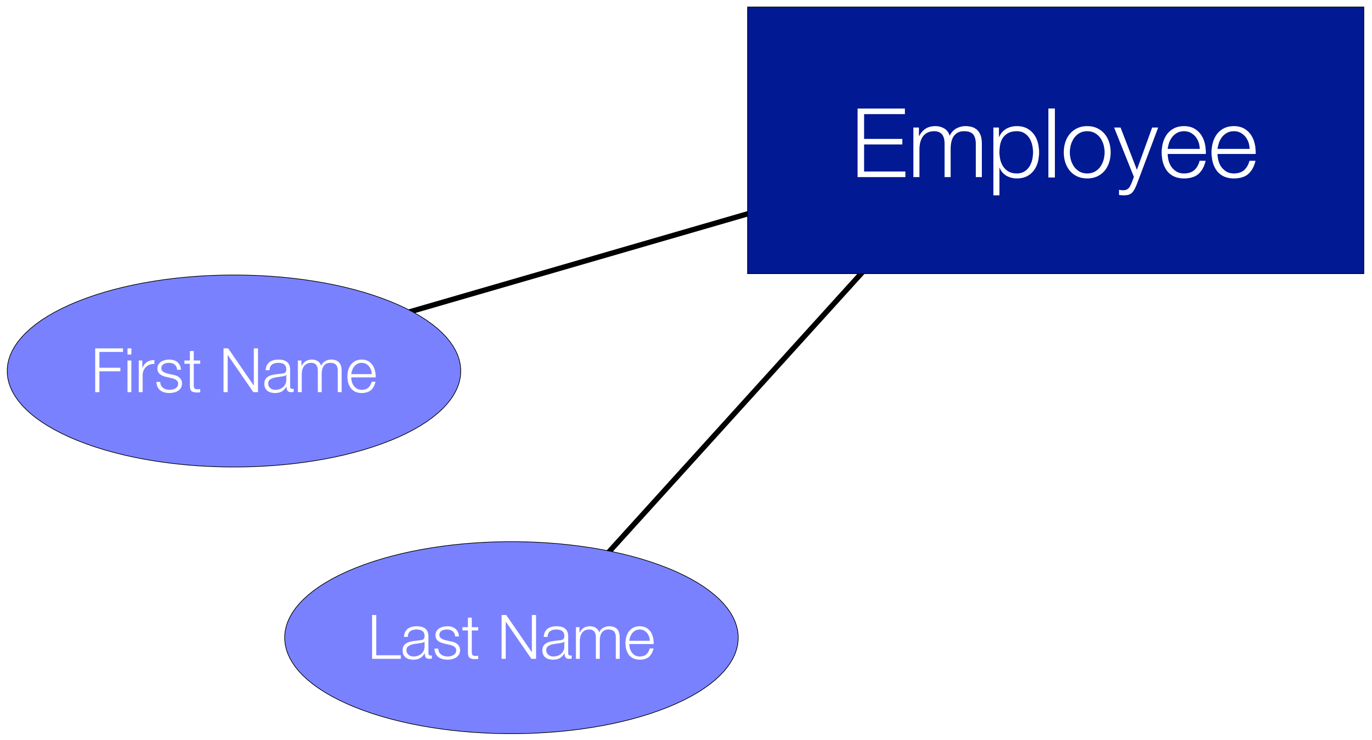
Applied to Creamery

Applied to Creamery

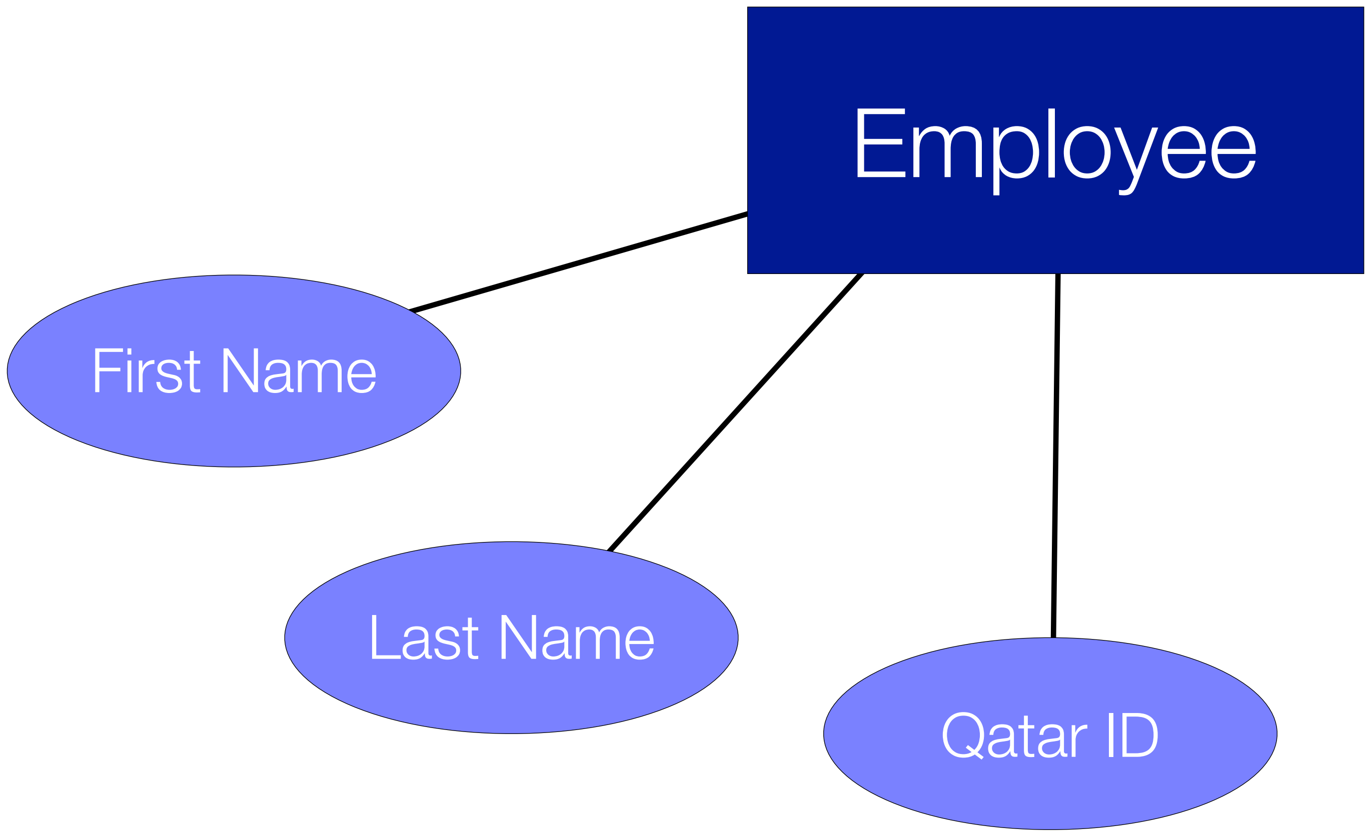
Applied to Creamery



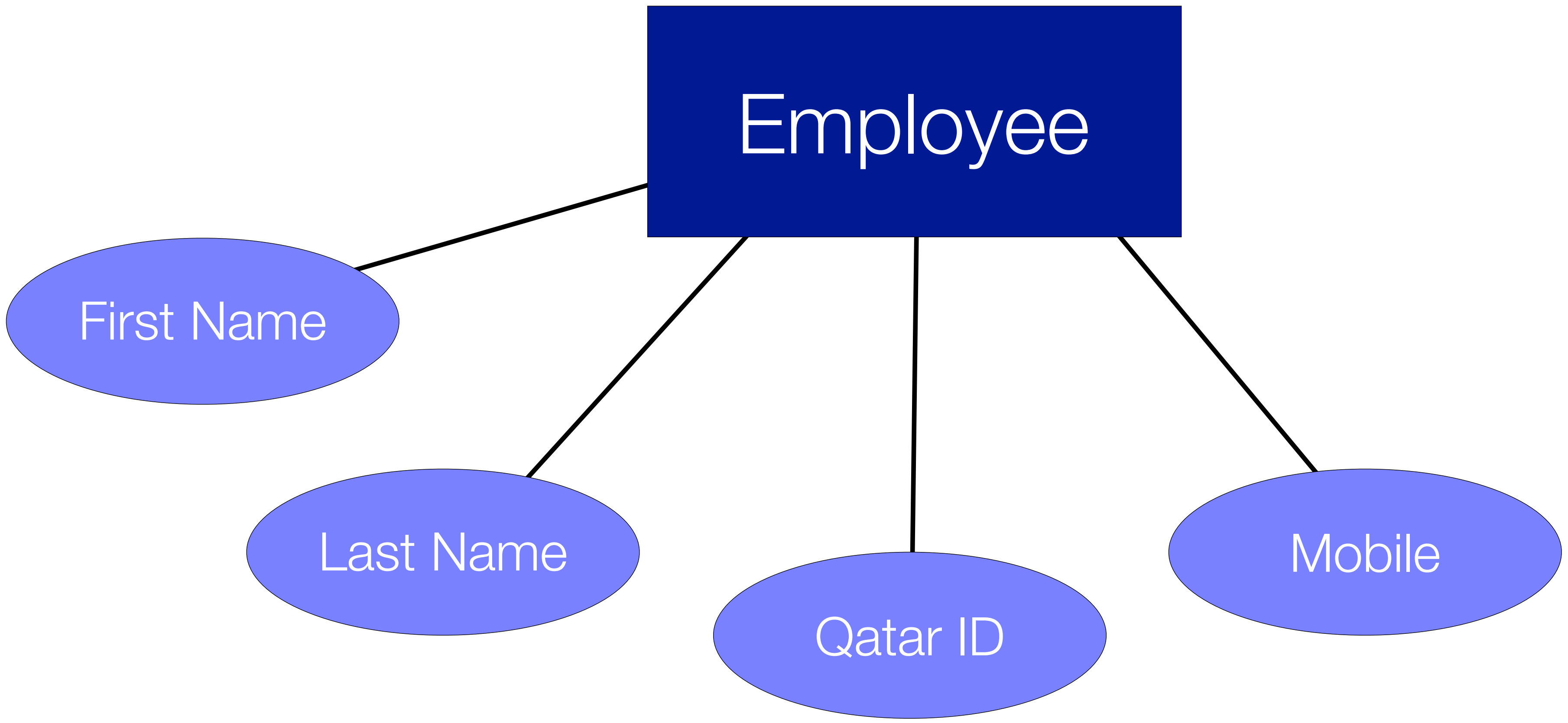
Applied to Creamery



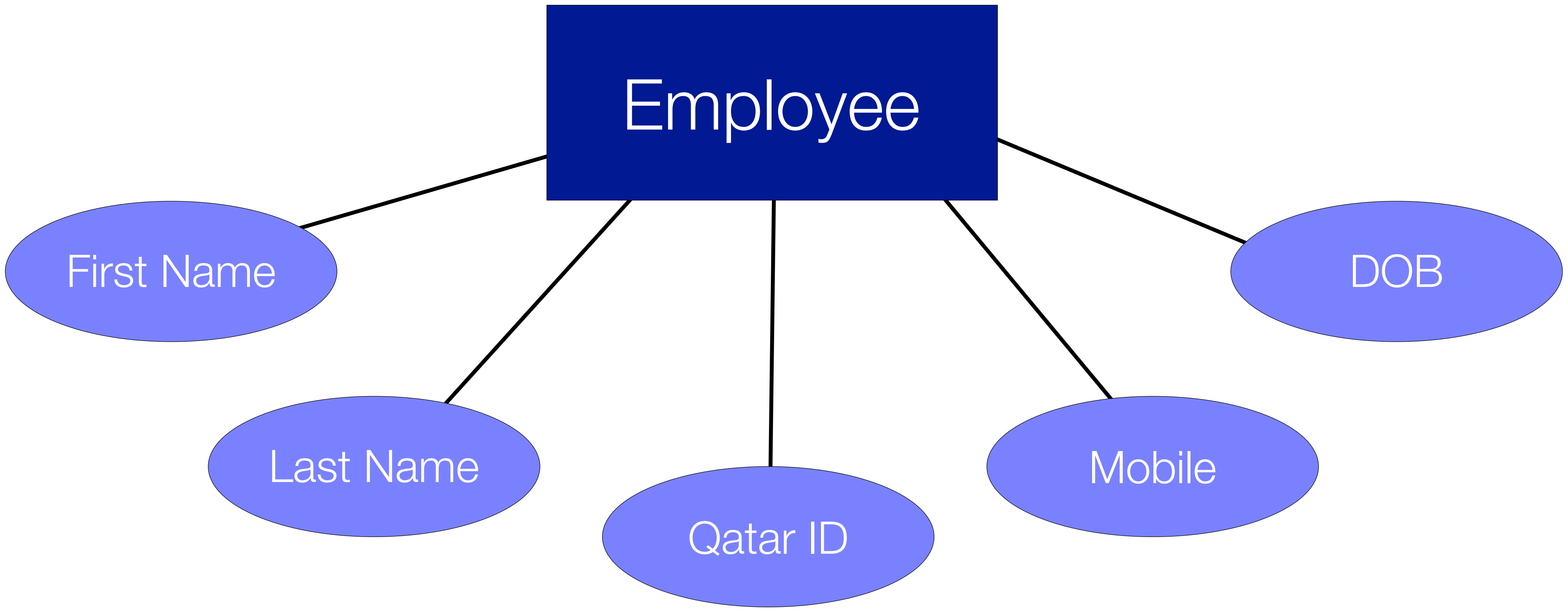
Applied to Creamery



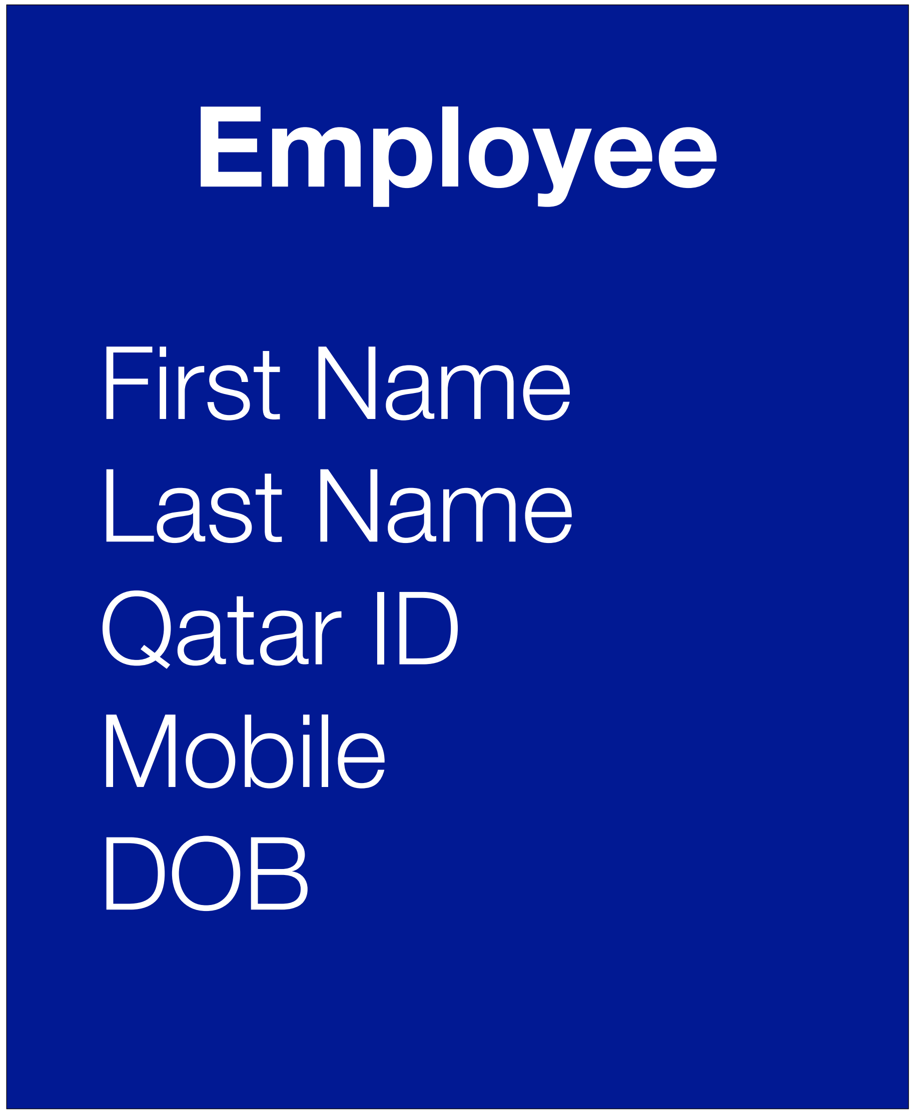
Applied to Creamery



Applied to Creamery



Another look at “Employee”



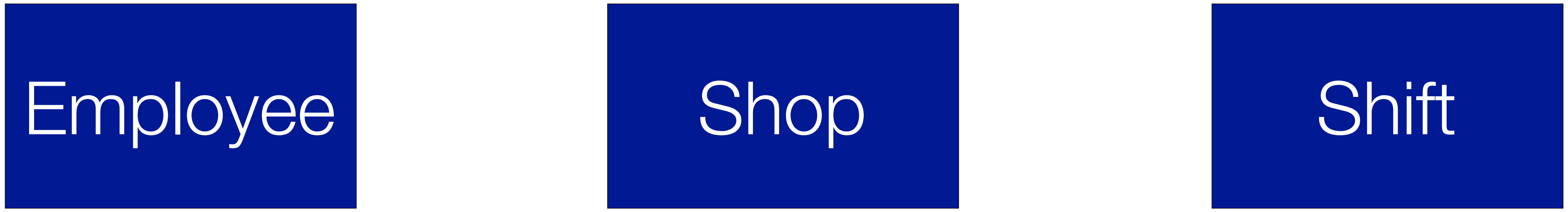
All the entities

All the entities

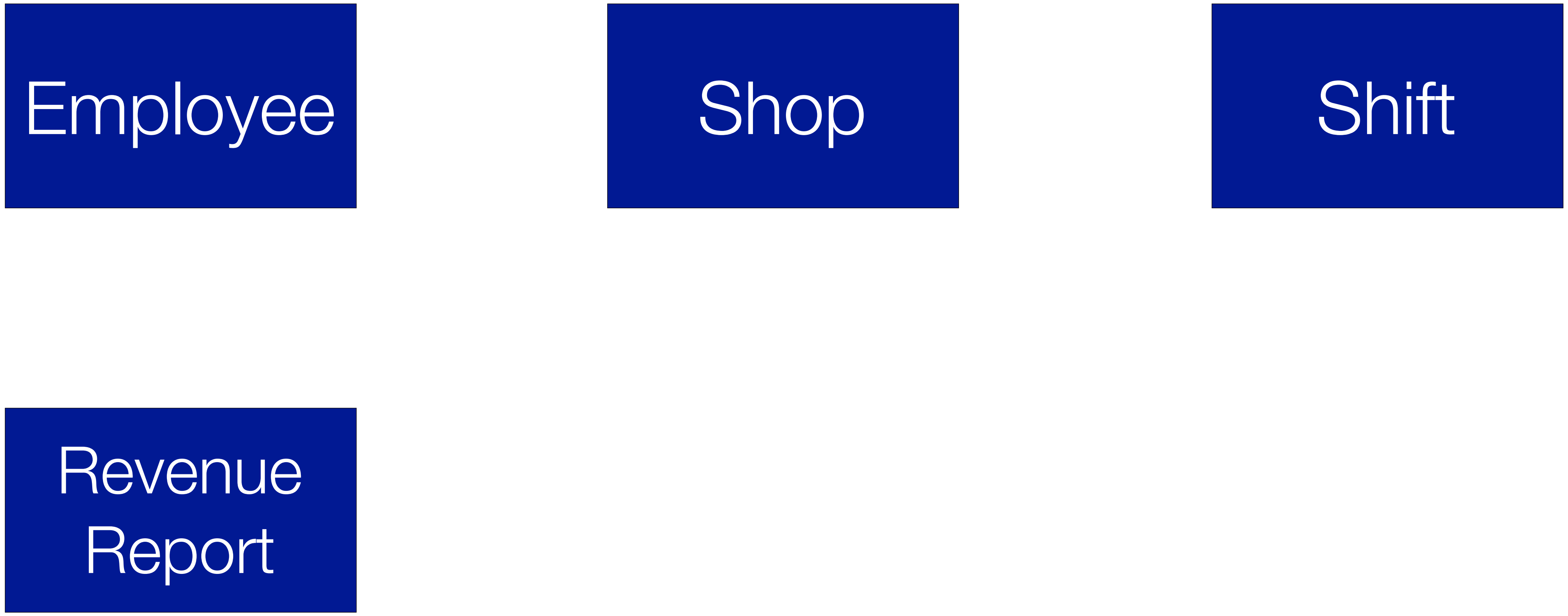
All the entities

Employee
Shop

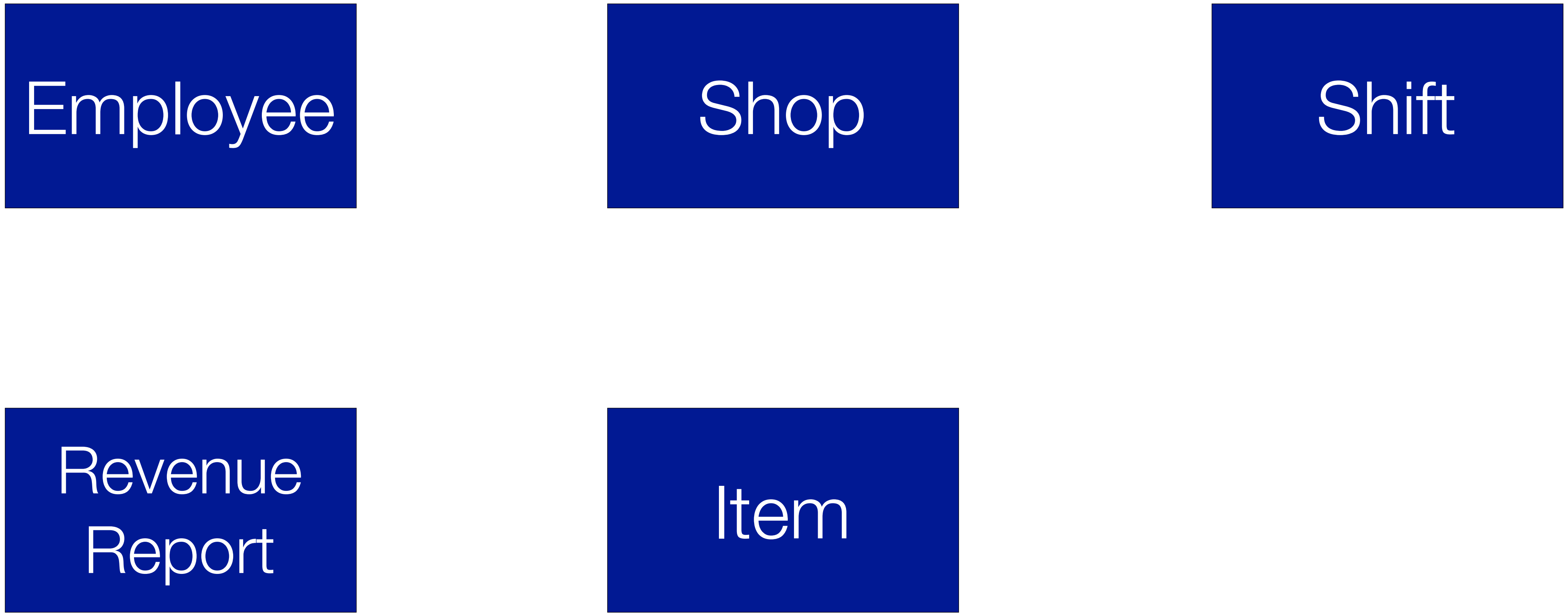
All the entities



All the entities



All the entities



All the entities

Employee

Shop

Shift

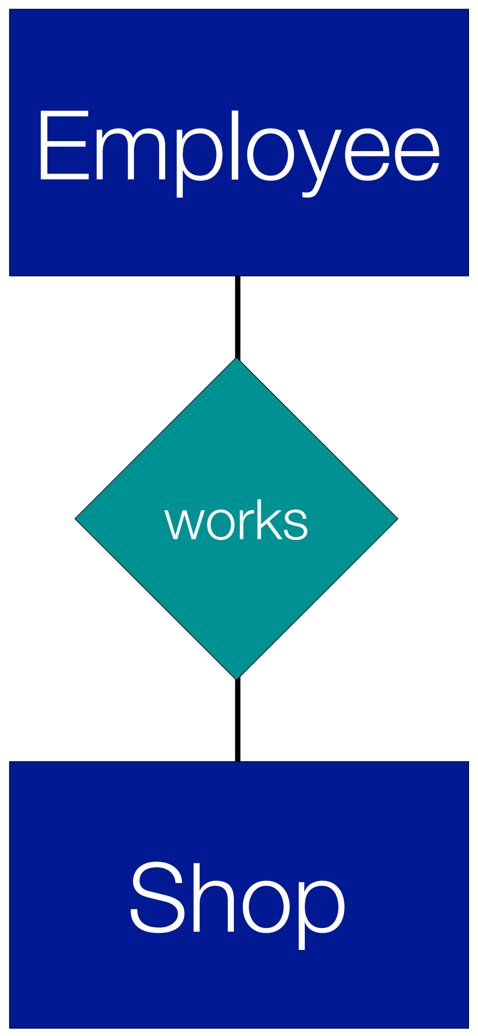
Revenue Report

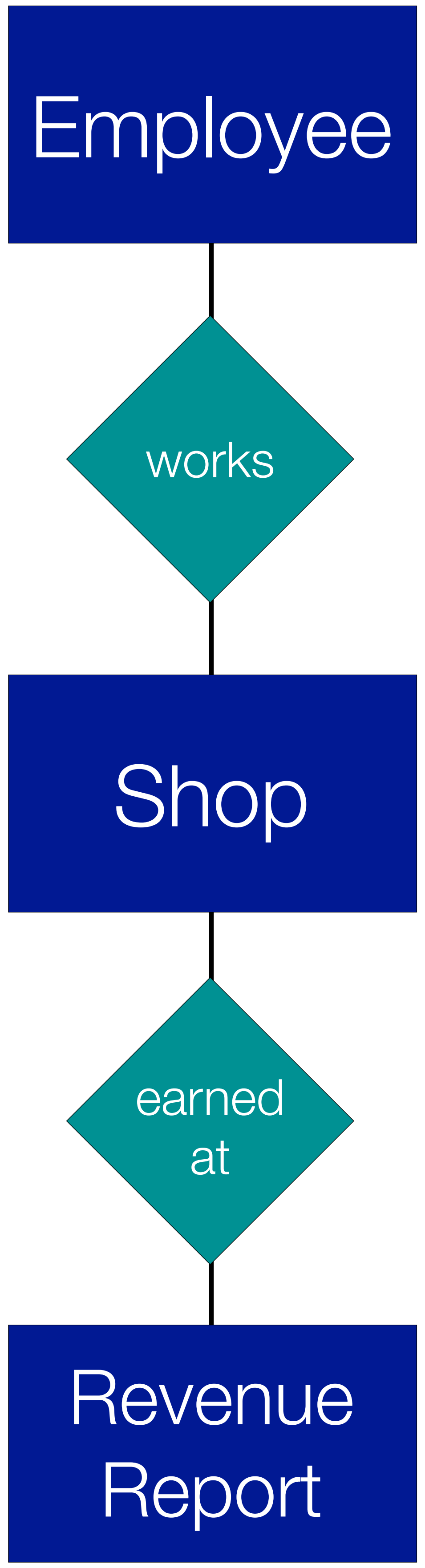
Item

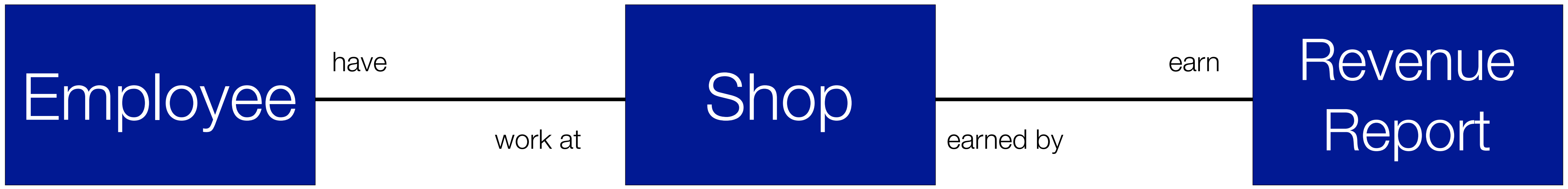
Transfer

What about relationships?

What about relationships?


Employee
works
Shop


Employee
works
Shop
earned at
Revenue Report


Employee
have
work at
Shop
earned by
earn
Revenue Report

Types of Relationships

Types of Relationships

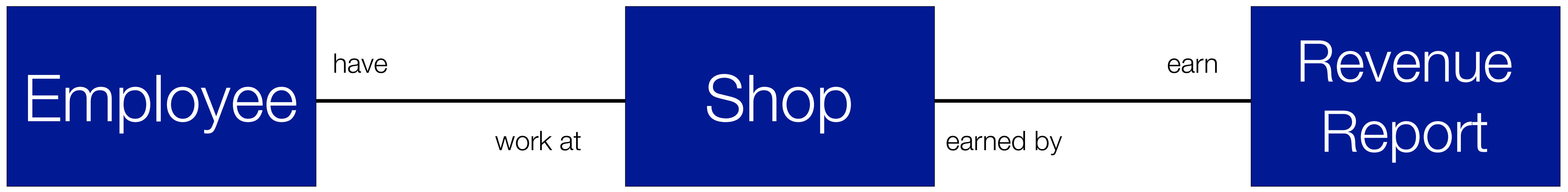
one-to-one...

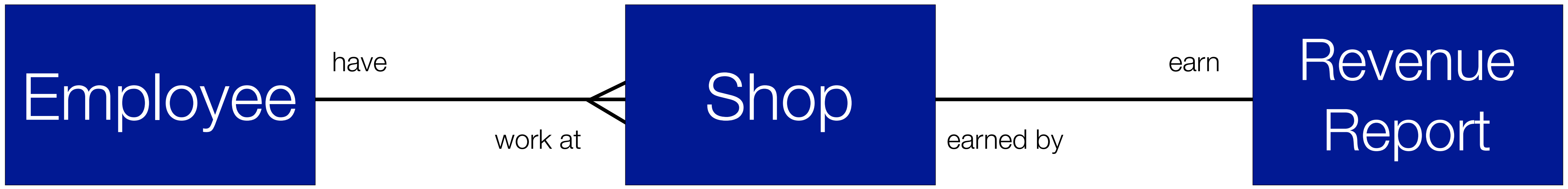
Types of Relationships

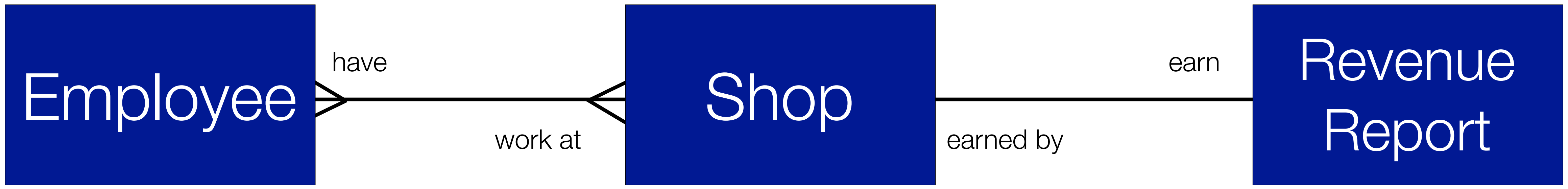
one-to-one...

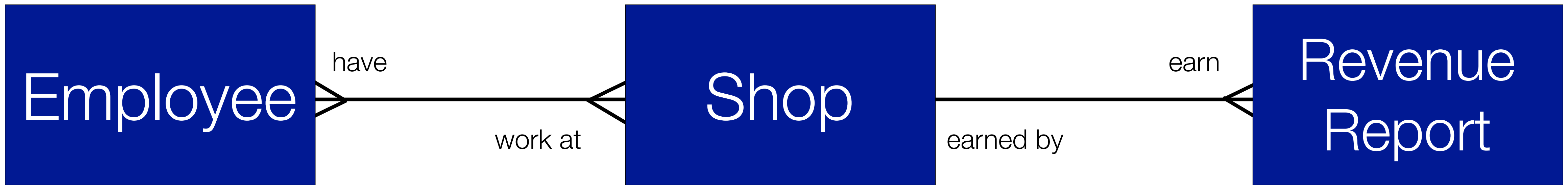
one-to-many (parent to child)

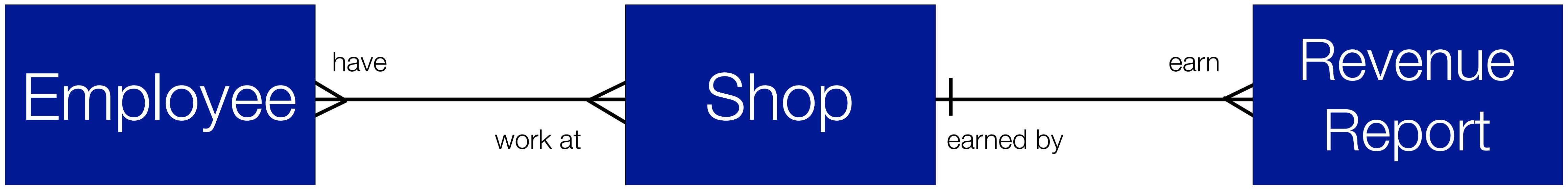
many-to-many (sibling to sibling)

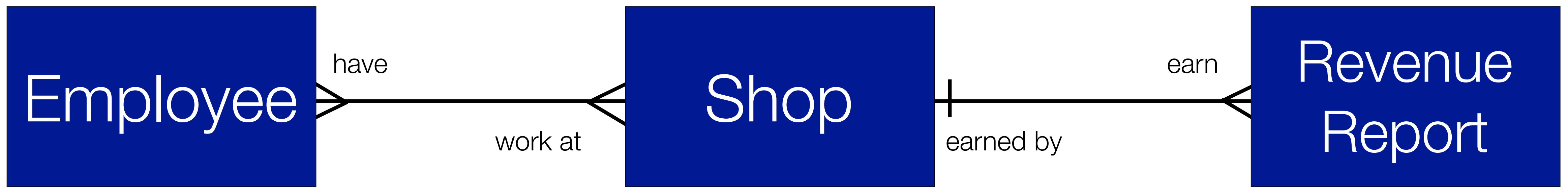

Employee
have
work at
Shop
earn
earned by
Revenue Report


Employee
have
work at
Shop
earned by
earn
Revenue Report

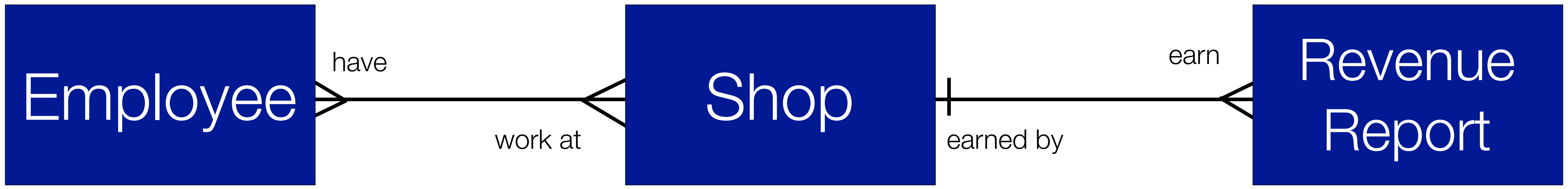

Employee
have
work at
Shop
earn
earned by
Revenue Report

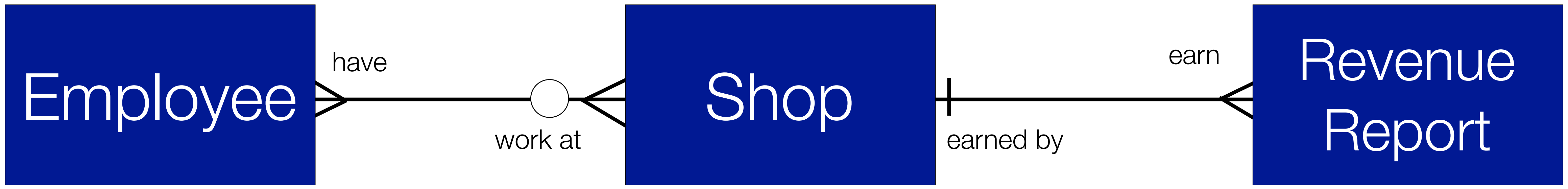

Employee
have
work at
Shop
earned by
earn
Revenue Report

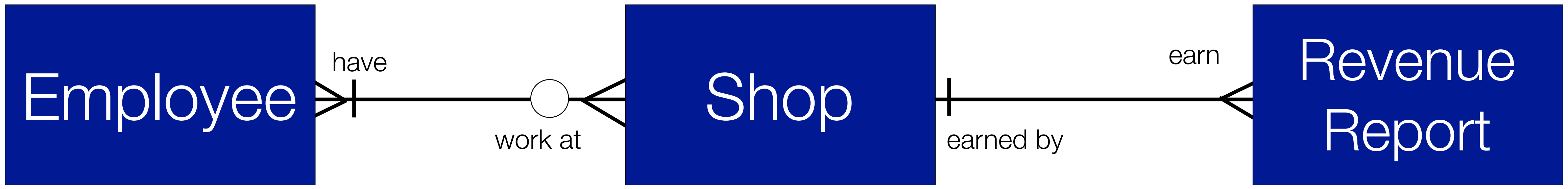

Employee
have
work at
Shop
earned by
earn
Revenue Report

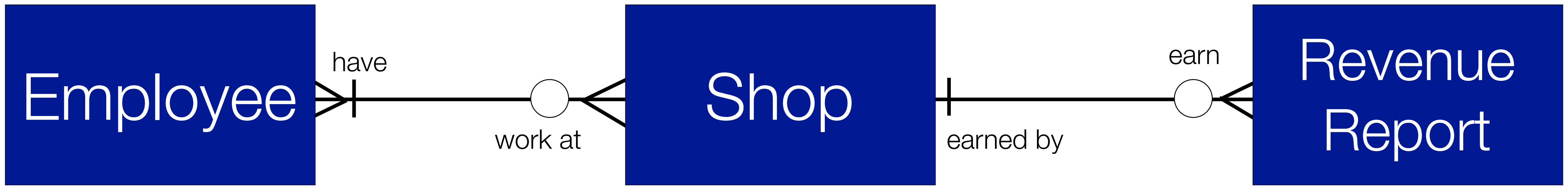


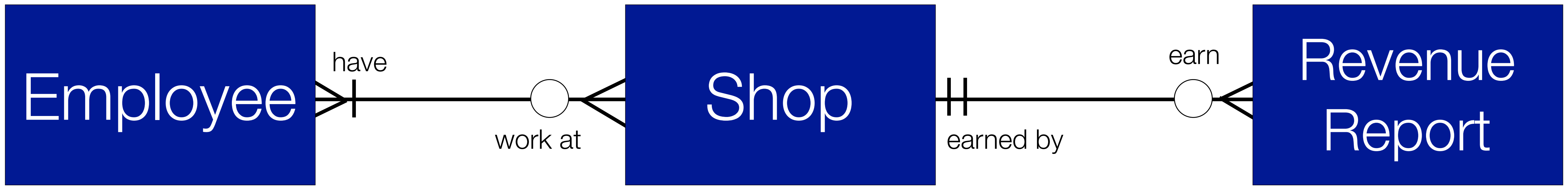
Mandatory (Optional) Relationships?

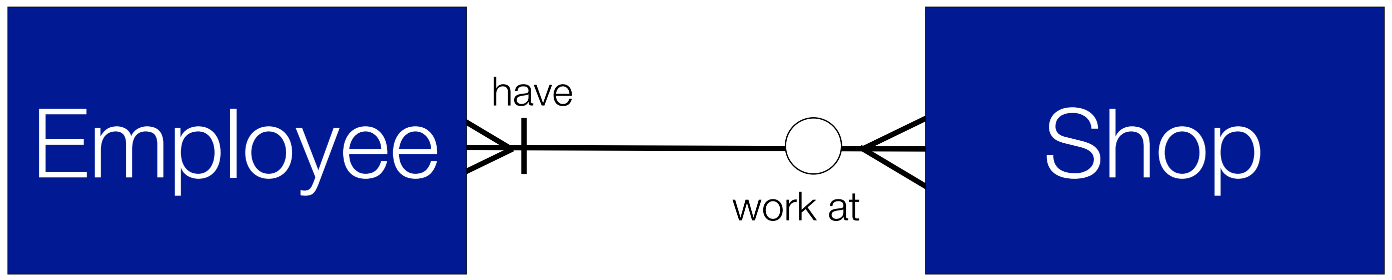

Employee
have
work at
Shop
earned by
earn
Revenue Report

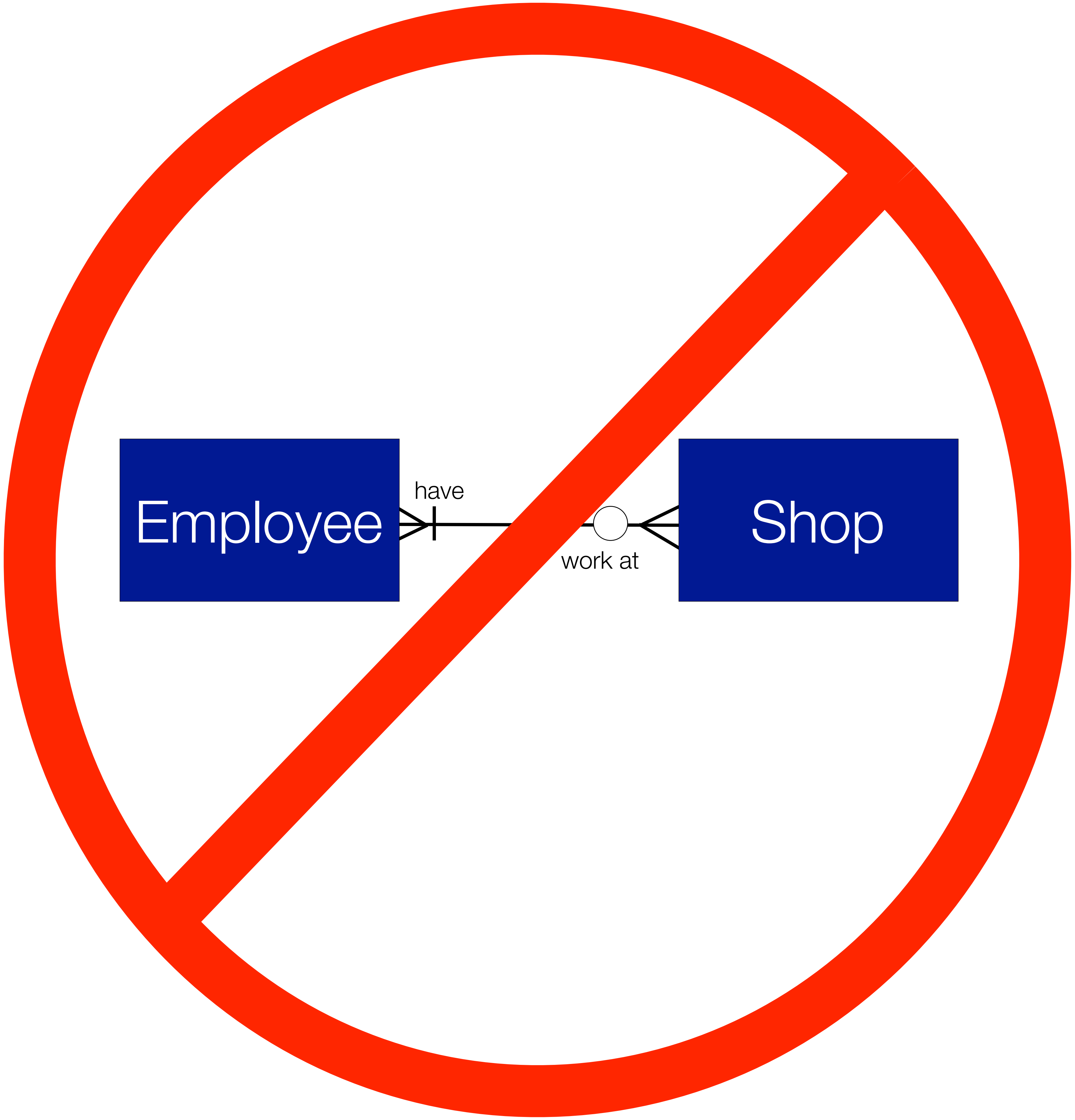

Employee
have
work at
Shop
earned by
earn
Revenue Report


Employee
have
work at
Shop
earned by
earn
Revenue Report

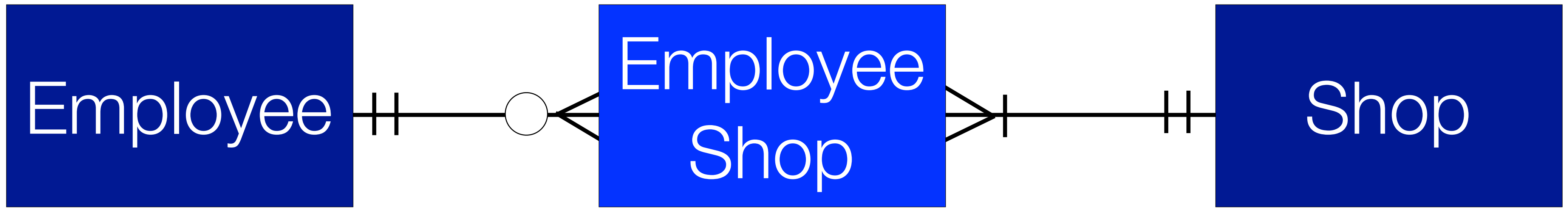

Employee
have
work at
Shop
earned by
earn
Revenue Report

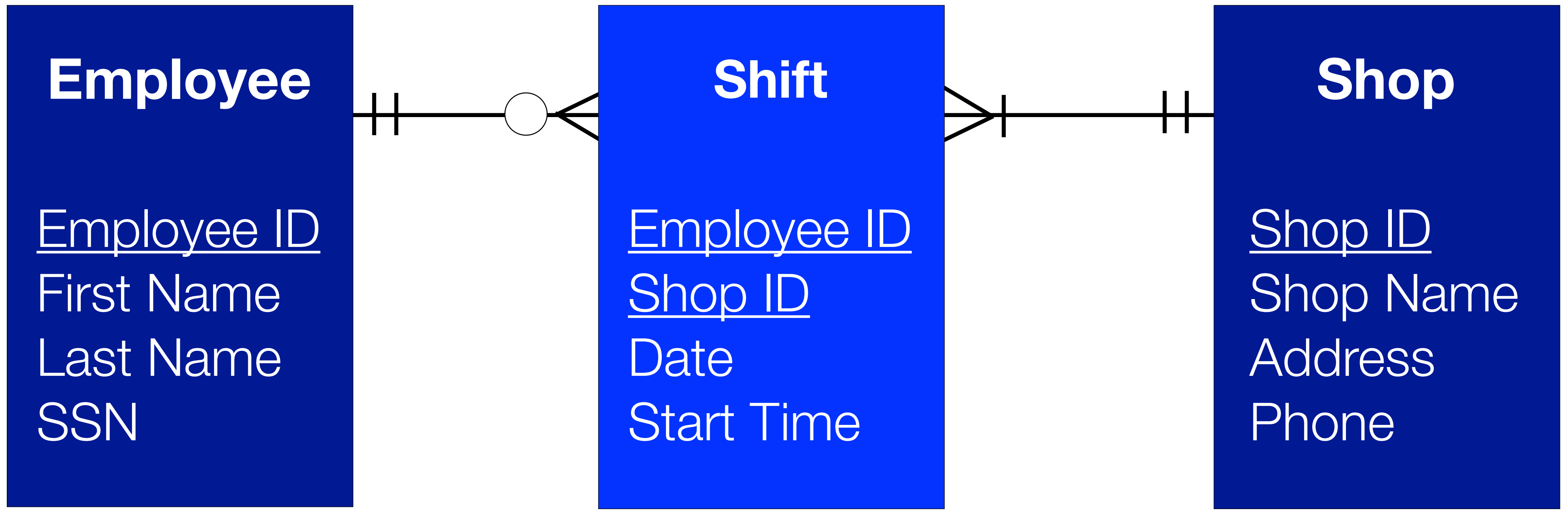

Employee
have
work at
Shop
earned by
earn
Revenue Report

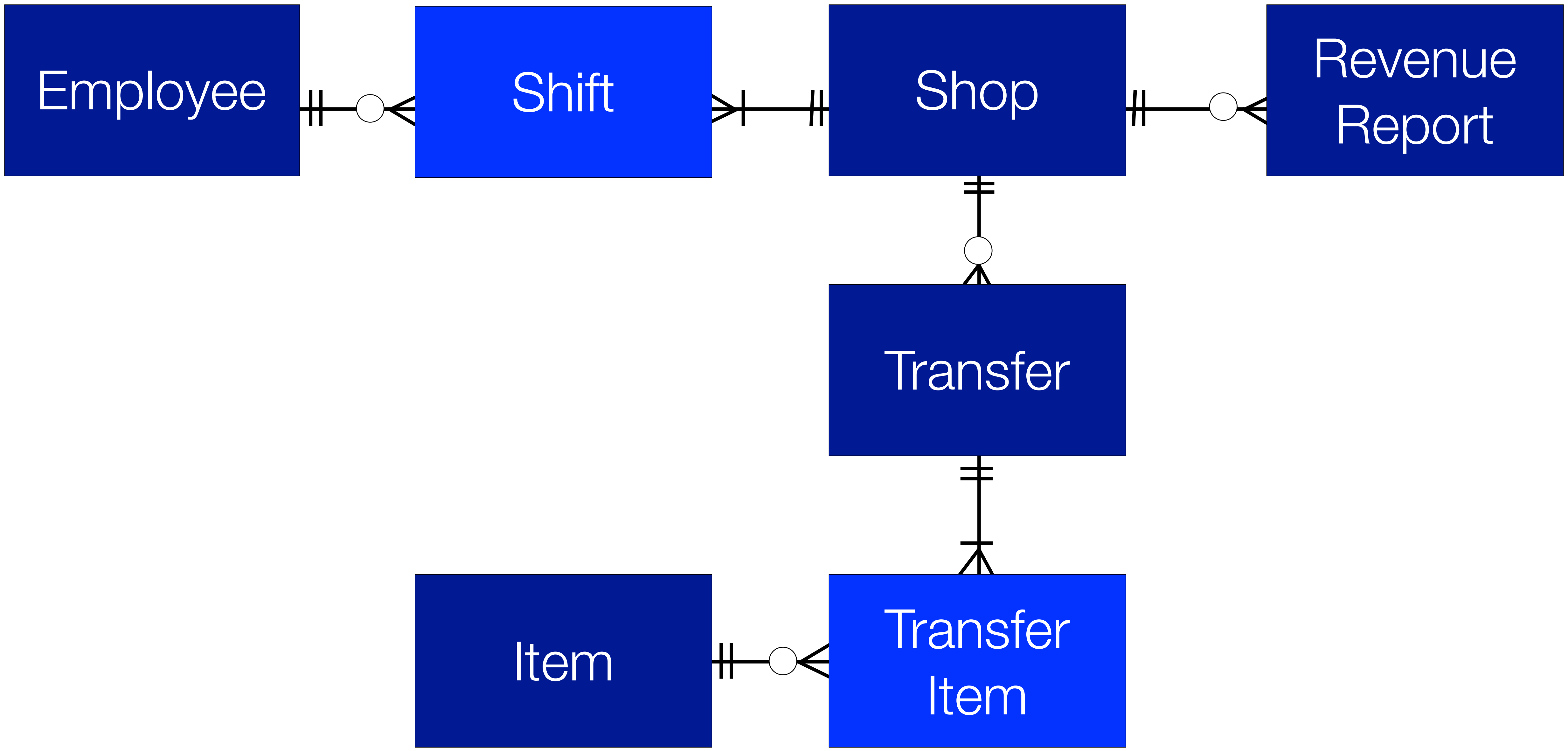

Employee
have
work at
Shop


Employee
have
work at
Shop

Create Associations




Employee
Employee ID
First Name
Last Name
SSN
Shift
Employee ID
Shop ID
Date
Start Time
Shop
Shop ID
Shop Name
Address
Phone


Employee
Shift
Shop
Revenue Report
Transfer
Item
Transfer Item

Summary :: ERD

Summary :: ERD

- Identify all entities and attributes

Summary :: ERD

- Identify all entities and attributes
- Define relationships between entities

Summary :: ERD

- Identify all entities and attributes
- Define relationships between entities
- Determine connectivity and transform many-to-many relationships

Summary :: ERD

- Identify all entities and attributes
- Define relationships between entities
- Determine connectivity and transform many-to-many relationships
- Ascertain whether required/optional

Summary :: ERD

- Identify all entities and attributes
- Define relationships between entities
- Determine connectivity and transform many-to-many relationships
- Ascertain whether required/optional
- Recognize that data modeling is usually iterative process

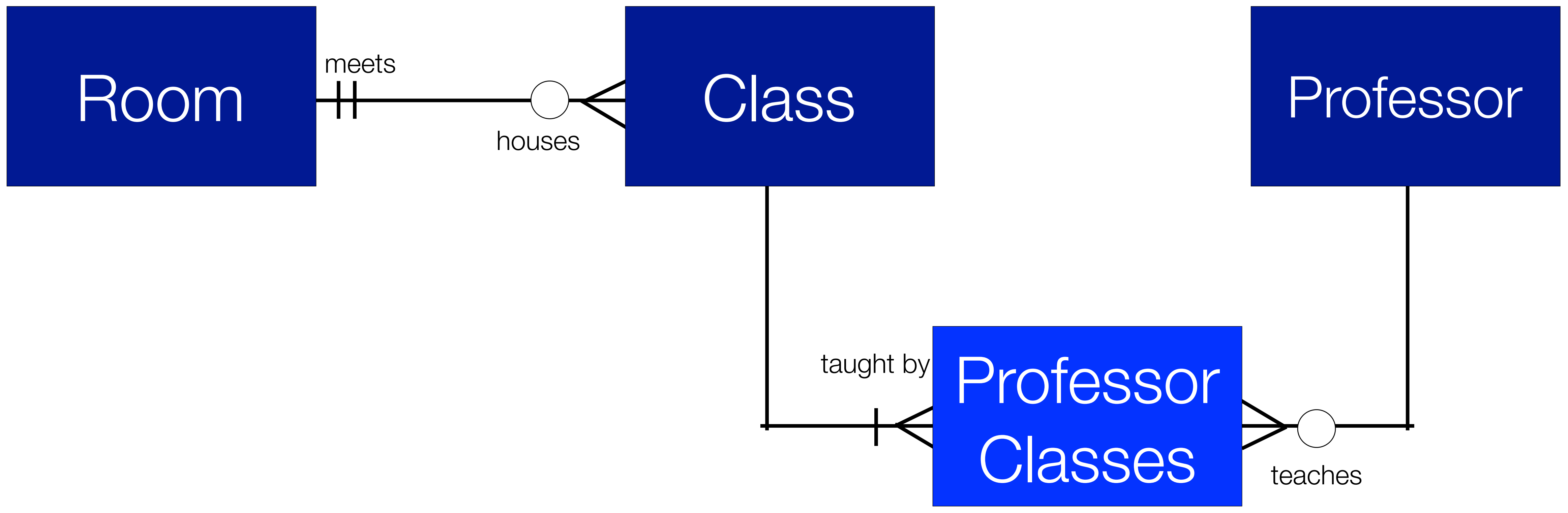
Class Problem

A college runs many classes. Each class is taught by one or more teachers, and a teacher may teach several classes. A particular class always uses the same room. Because classes may meet at different times or on different evenings, it is possible for different classes to use the same room.

Draw out a simple ERD to capture the essential information in this example.

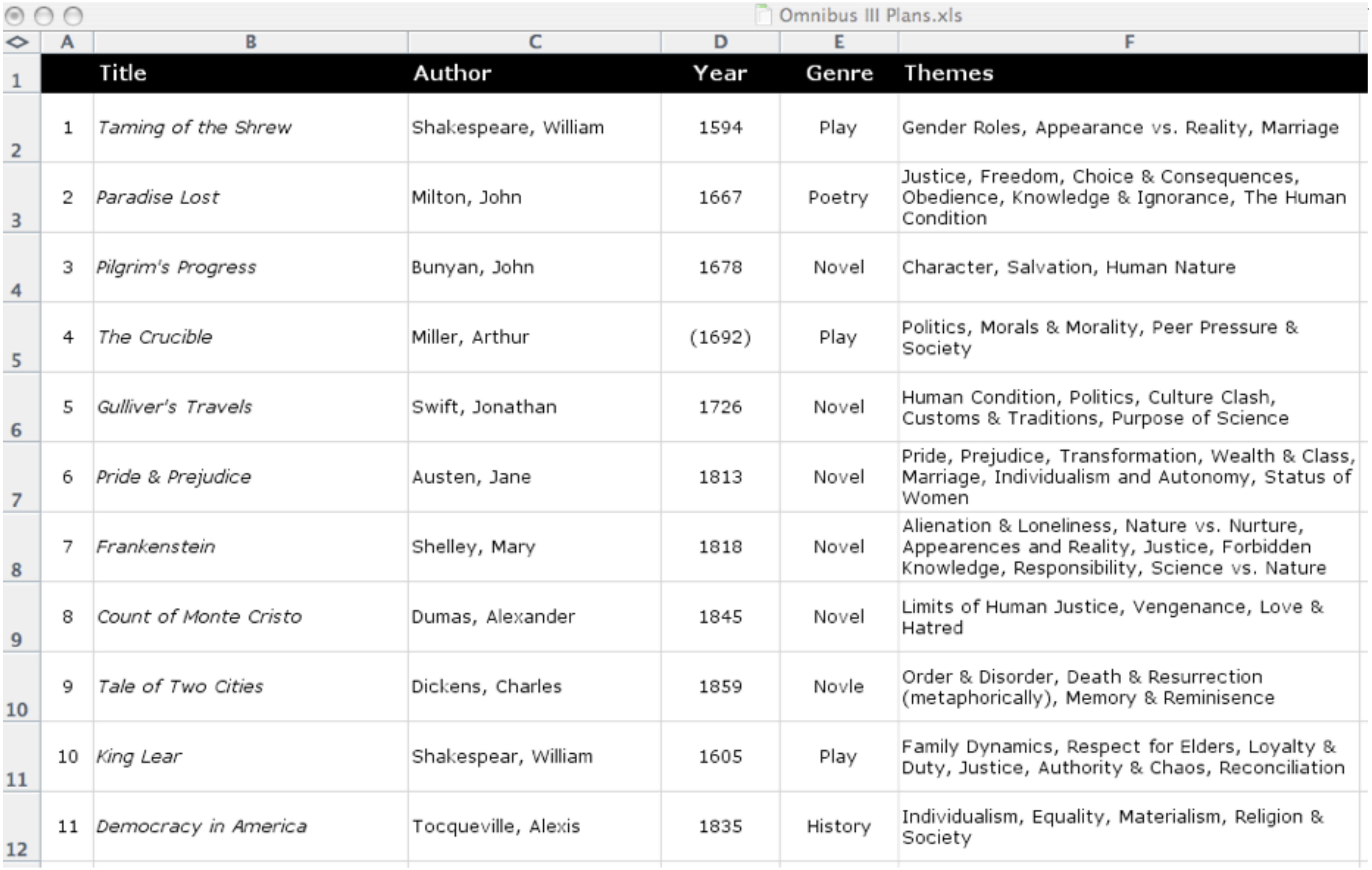
Example courtesy of db4u (http://db4u.wikidot.com)

Solution


Room
meets
houses
Class
Professor
taught by
Professor Classes
teaches

Organizing the Logical Data Model

The old way of doing it...


Omnibus III Plans.xls

	Title	Author	Year	Genre	Themes
1	*Taming of the Shrew*	Shakespeare, William	1594	Play	Gender Roles, Appearance vs. Reality, Marriage
2	*Paradise Lost*	Milton, John	1667	Poetry	Justice, Freedom, Choice & Consequences, Obedience, Knowledge & Ignorance, The Human Condition
3	*Pilgrim's Progress*	Bunyan, John	1678	Novel	Character, Salvation, Human Nature
4	*The Crucible*	Miller, Arthur	(1692)	Play	Politics, Morals & Morality, Peer Pressure & Society
5	*Gulliver's Travels*	Swift, Jonathan	1726	Novel	Human Condition, Politics, Culture Clash, Customs & Traditions, Purpose of Science
6	*Pride & Prejudice*	Austen, Jane	1813	Novel	Pride, Prejudice, Transformation, Wealth & Class, Marriage, Individualism and Autonomy, Status of Women
7	*Frankenstein*	Shelley, Mary	1818	Novel	Alienation & Loneliness, Nature vs. Nurture, Appearences and Reality, Justice, Forbidden Knowledge, Responsibility, Science vs. Nature
8	*Count of Monte Cristo*	Dumas, Alexander	1845	Novel	Limits of Human Justice, Vengenance, Love & Hatred
9	*Tale of Two Cities*	Dickens, Charles	1859	Novle	Order & Disorder, Death & Resurrection (metaphorically), Memory & Reminisence
10	*King Lear*	Shakespear, William	1605	Play	Family Dynamics, Respect for Elders, Loyalty & Duty, Justice, Authority & Chaos, Reconciliation
11	*Democracy in America*	Tocqueville, Alexis	1835	History	Individualism, Equality, Materialism, Religion & Society

A new tool: relational databases

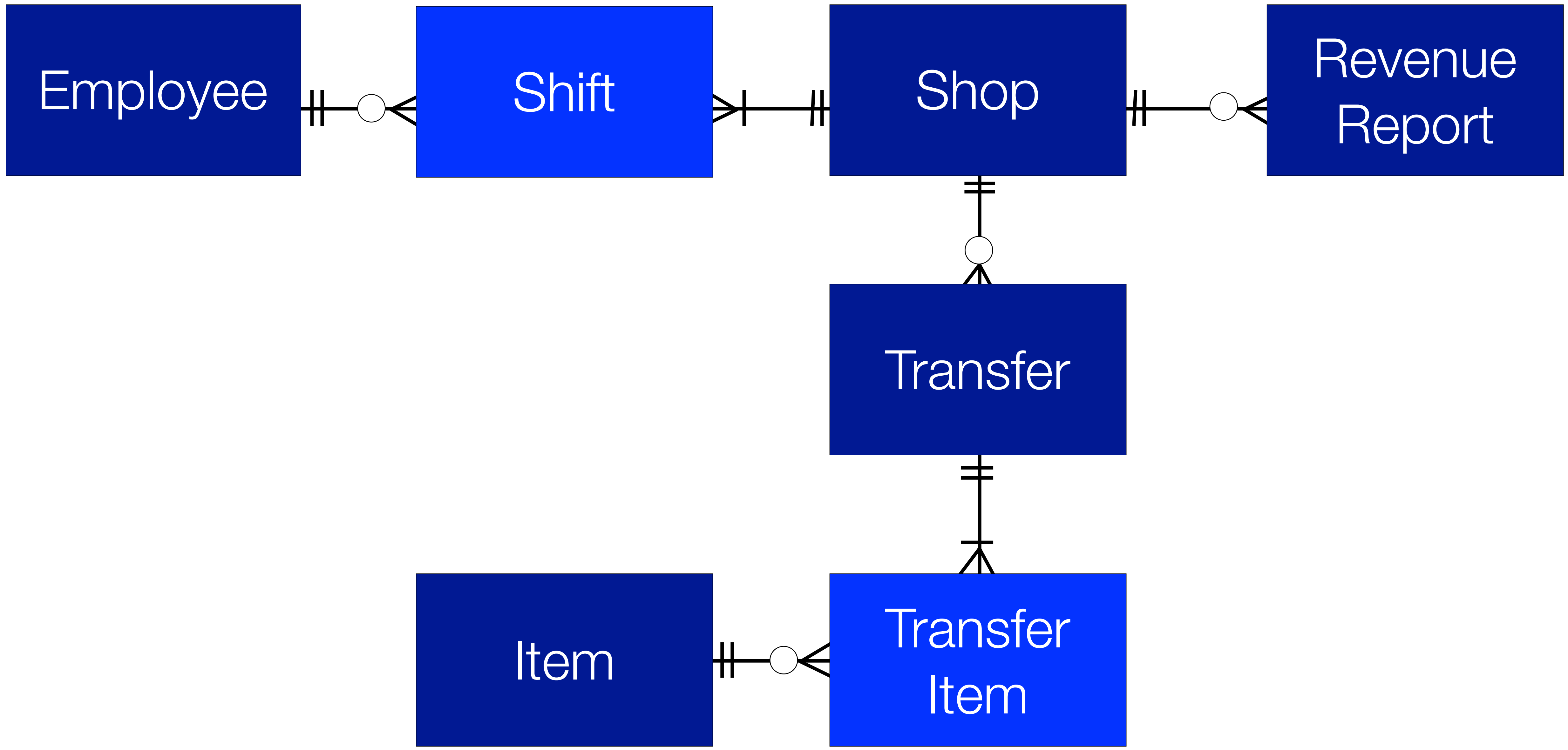
- Create a series of data tables and a means to link them together so that data can be combined and extracted as need be.
- Terminology
  - Table (file)
  - Record (row)
  - Field (column)

employees

id	first_name	last_name	DOB
1	Mark	Heimann	1993-01-25
2	Alex	Heimann	1993-01-25
3	John	Milton	1608-12-09
4	Mary	Shelley	1797-08-30

shifts

id	employee_id	date
1	3	2008-08-04
2	3	2008-08-05
3	4	2008-08-04
4	2	2008-08-05


Employee
Shift
Shop
Revenue Report
Transfer
Item
Transfer Item

Converting from ERD

- All entities become tables
- All attributes become fields
- Primary keys need to be set

What is the “key” to RDB?

Types of keys

Types of keys

Primary: Uniquely identify a record in a table

Types of keys

- Primary: Uniquely identify a record in a table
- Foreign: A field in Table A which is also a primary key in Table B; used to establish links between tables

Types of keys

- Primary: Uniquely identify a record in a table
- Foreign: A field in Table A which is also a primary key in Table B; used to establish links between tables
- Composite: A combination of keys which together serve to uniquely identify a record

Creamery database, v. 0.5

Creamery database, v. 0.5

employees(id, first_name, last_name, date_of_birth, ssn, pay_rate, email, active)

Creamery database, v. 0.5

employees(id, first_name, last_name, date_of_birth, ssn, pay_rate, email, active)

shifts(employee_id, shop_id, start_at, hours_worked)

Creamery database, v. 0.5

employees(id, first_name, last_name, date_of_birth, ssn, pay_rate, email, active)

shifts(id, employee_id, shop_id, start_at, hours_worked)

Creamery database, v. 0.5

employees(id, first_name, last_name, date_of_birth, ssn, pay_rate, email, active)

shifts(id, employee_id, shop_id, start_at, hours_worked)

shops(id, short_name, street, zip, weekly_rent, phone, active)

Creamery database, v. 0.5

employees(id, first_name, last_name, date_of_birth, ssn, pay_rate, email, active)

shifts(id, employee_id, shop_id, start_at, hours_worked)

shops(id, short_name, street, zip, weekly_rent, phone, active)

revenues(id, shop_id, week_ending, weekly_amount)

Database Integrity

Database Integrity

- 1st Type: Entity Integrity
    - key idea -- table must have a valid primary key

Database Integrity

- 1st Type: Entity Integrity
    - key idea -- table must have a valid primary key
- 2nd Type: Domain Integrity
    - key idea -- data type and format must be valid

Database Integrity

- 1st Type: Entity Integrity
  - key idea -- table must have a valid primary key
- 2nd Type: Domain Integrity
  - key idea -- data type and format must be valid
- 3rd Type: Referential Integrity
  - key idea -- don’t leave behind orphaned records

Normalizing databases

Normalizing databases

- What is normalization?

Normalizing databases

- What is normalization?
    - process to create a flexible, nonredundant, and efficient data model that can be implemented in a RDB

Normalizing databases

- What is normalization?
  - process to create a flexible, nonredundant, and efficient data model that can be implemented in a RDB
  - helps preserve referential integrity

Normalizing databases

- What is normalization?
  - process to create a flexible, nonredundant, and efficient data model that can be implemented in a RDB
  - helps preserve referential integrity
- How important is normalization?

Normalizing databases

- What is normalization?
  - process to create a flexible, nonredundant, and efficient data model that can be implemented in a RDB
  - helps preserve referential integrity
- How important is normalization?
  - Usually problematic if < 3NF

Normalizing databases

- What is normalization?
  - process to create a flexible, nonredundant, and efficient data model that can be implemented in a RDB
  - helps preserve referential integrity
- How important is normalization?
  - Usually problematic if < 3NF
  - A case can be made for “sensible normalization”

Normalization & the creamery

transfers(transfer_id, shop_id, shop_name, shop_phone, shop_street, shop_zip, date_requested, date_fulfilled, {1-N occurrences of the following group}: item_id, item_name, item_cost, quantity_requested, quantity_transferred)

Question 1: Does the entity have repeating any elements?

Question 1: Does the entity have repeating any elements?

transfers(transfer_id, shop_id, shop_name, shop_phone, shop_street, shop_zip, date_requested, date_fulfilled)

transfer_items (transfer_id, item_id, item_name, item_cost, quantity_requested, quantity_transferred)

Question 2: Does the entity have a composite key?

Question 2: Does the entity have a composite key?

- If not, go on to question 3...

Question 2: Does the entity have a composite key?

- If not, go on to question 3...
- If so, ask the follow-up question ...

Question 2: Does the entity have a composite key?

- If not, go on to question 3...
- If so, ask the follow-up question ...

Question 2a: Are there any partial dependencies?

Question 2: Does the entity have a composite key?

- If not, go on to question 3...
- If so, ask the follow-up question ...

Question 2a: Are there any partial dependencies?

- That is to say, are there any fields in the table that depend on only *part* of the composite key?

Another look at the example

transfers(transfer_id, shop_id, shop_name, shop_phone, shop_street, shop_zip, date_requested, date_fulfilled)

transfer_items (transfer_id, item_id, item_name, item_cost, quantity_requested, quantity_transferred)

Another look at the example

transfers(transfer_id, shop_id, shop_name, shop_phone, shop_street, shop_zip, date_requested, date_fulfilled)

transfer_items (transfer_id, item_id, quantity_requested, quantity_transferred)

items (item_id, item_name, item_cost)

Question 3: Are there any transitive dependencies?

Question 3: Are there any transitive dependencies?

- That is to say, are there any fields in the table that depend on another field in the table that is *not* the primary key?

Where is the transitive dependency?

transfers(transfer_id, shop_id, shop_name, shop_phone, shop_street, shop_zip, date_requested, date_fulfilled)

transfer_items(transfer_id, item_id, quantity_requested, quantity_transferred)

items(item_id, item_name, item_cost)

A 3NF version

shops(shop_id, shop_name, shop_phone, shop_street, shop_zip)

transfers(transfer_id, shop_id, date_requested, date_fulfilled)

transfer_items(transfer_id, item_id, quantity_requested, quantity_transferred)

items (item_id, item_name, item_cost)

Denormalization

Denormalization

- Examples

Denormalization

- Examples
    - Address, phone data

Denormalization

- Examples
  - Address, phone data
  - City, ST -> Zip Code

Denormalization

- Examples
  - Address, phone data
  - City, ST -> Zip Code
- Why denormalize?

Denormalization

- Examples
    - Address, phone data
    - City, ST -> Zip Code
- Why denormalize?
- Dangers of denormalization

Creamery database, v. 1.0

Creamery database, v. 1.0

employees(<u>id</u>, first_name, last_name, date_of_birth, ssn, pay_rate, email, active)

Creamery database, v. 1.0

employees(id, first_name, last_name, date_of_birth, ssn, pay_rate, email, active)

shifts(employee_id, shop_id, start_at, hours_worked)

Creamery database, v. 1.0

employees(id, first_name, last_name, date_of_birth, ssn, pay_rate, email, active)

shifts(id, employee_id, shop_id, start_at, hours_worked)

Creamery database, v. 1.0

employees(id, first_name, last_name, date_of_birth, ssn, pay_rate, email, active)

shifts(id, employee_id, shop_id, start_at, hours_worked)

shops(id, short_name, street, zip, weekly_rent, phone, active)

Creamery database, v. 1.0

employees(id, first_name, last_name, date_of_birth, ssn, pay_rate, email, active)

shifts(id, employee_id, shop_id, start_at, hours_worked)

shops(id, short_name, street, zip, weekly_rent, phone, active)

revenues(id, shop_id, week_ending, weekly_amount)

Creamery database, v. 1.0

employees(id, first_name, last_name, date_of_birth, ssn, pay_rate, email, active)

shifts(id, employee_id, shop_id, start_at, hours_worked)

shops(id, short_name, street, zip, weekly_rent, phone, active)

revenues(id, shop_id, week_ending, weekly_amount)

transfers(id, shop_id, date_requested, date_fulfilled)

Creamery database, v. 1.0

employees(id, first_name, last_name, date_of_birth, ssn, pay_rate, email, active)

shifts(id, employee_id, shop_id, start_at, hours_worked)

shops(id, short_name, street, zip, weekly_rent, phone, active)

revenues(id, shop_id, week_ending, weekly_amount)

transfers(id, shop_id, date_requested, date_fulfilled)

transfer_items(transfer_id, item_id, quantity_requested, quantity_transferred)

Creamery database, v. 1.0

employees(id, first_name, last_name, date_of_birth, ssn, pay_rate, email, active)

shifts(id, employee_id, shop_id, start_at, hours_worked)

shops(id, short_name, street, zip, weekly_rent, phone, active)

revenues(id, shop_id, week_ending, weekly_amount)

transfers(id, shop_id, date_requested, date_fulfilled)

transfer_items(id, transfer_id, item_id, quantity_requested, quantity_transferred)

Creamery database, v. 1.0

employees(id, first_name, last_name, date_of_birth, ssn, pay_rate, email, active)

shifts(id, employee_id, shop_id, start_at, hours_worked)

shops(id, short_name, street, zip, weekly_rent, phone, active)

revenues(id, shop_id, week_ending, weekly_amount)

transfers(id, shop_id, date_requested, date_fulfilled)

transfer_items(id, transfer_id, item_id, quantity_requested, quantity_transferred)

items(id, name, cost, unit, units_in_stock)

Questions?