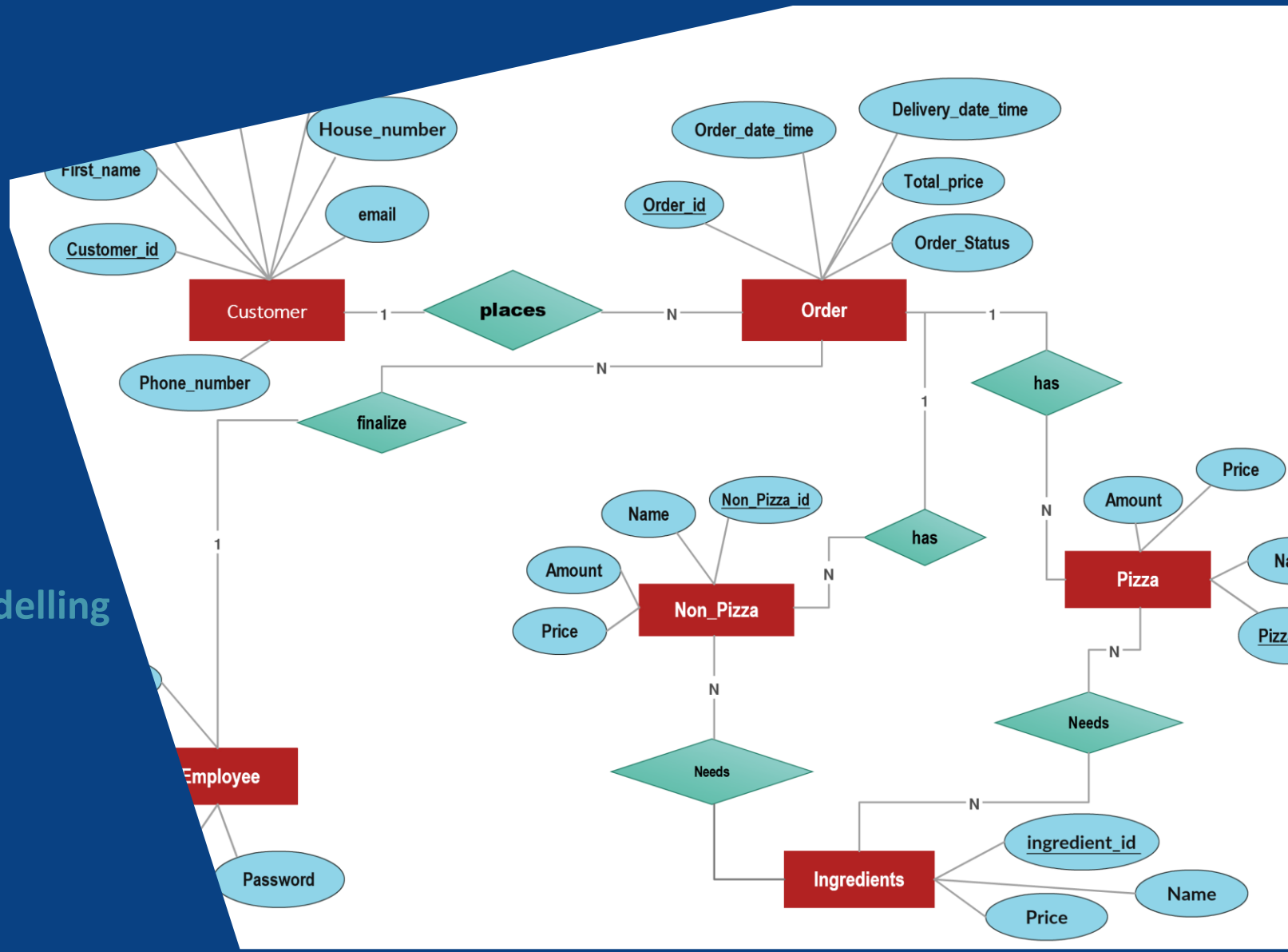
THE UNIVERSITY OF MELBOURNE

Introduction to Data Modelling

Database Systems & Information Modelling
INFO90002



Week 2 – Data Modelling
Dr Tanya Linden

Announcements

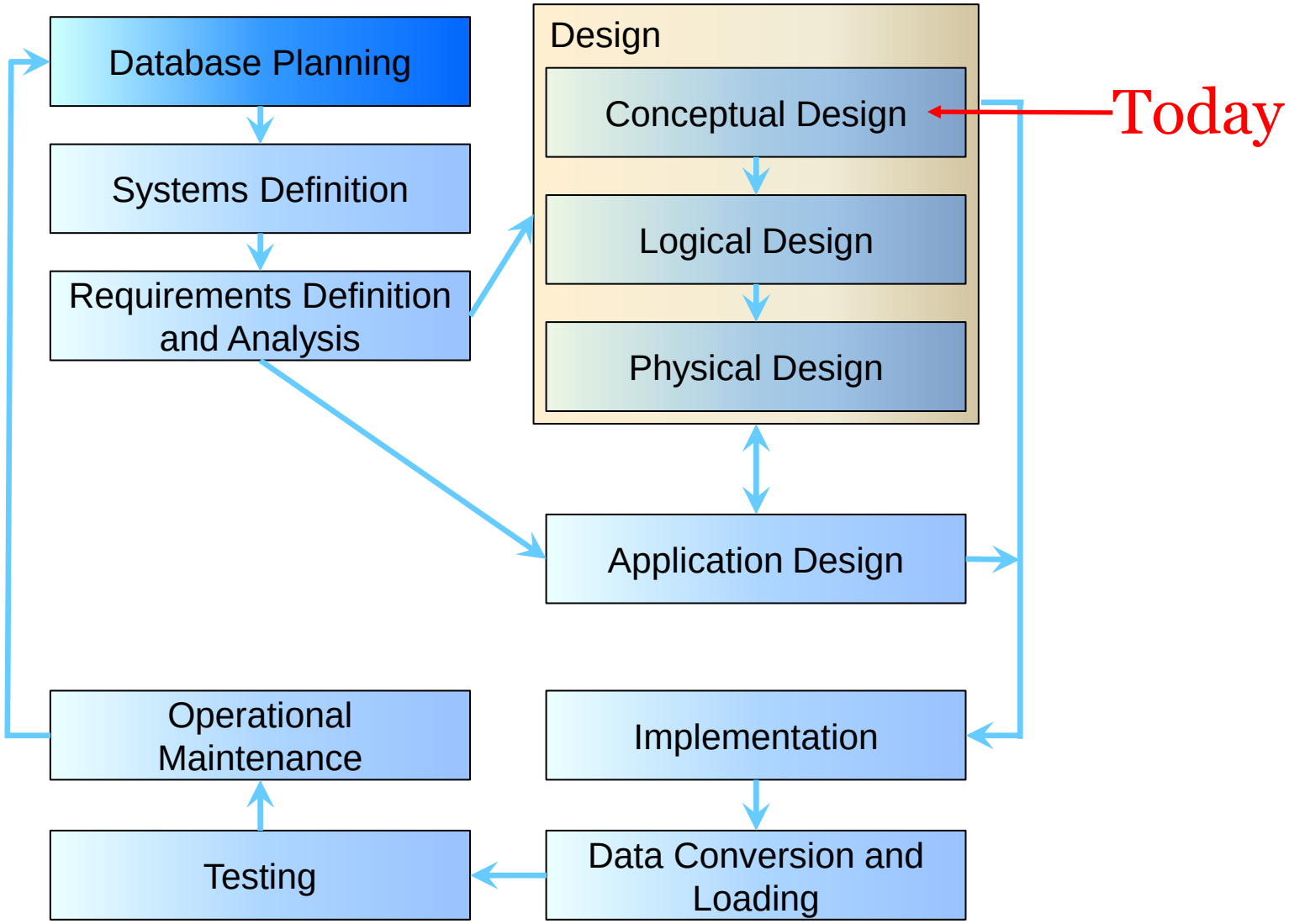
Assignment 1 - Time to form a team!

- Teams of 4
- All team members must belong to the same tute
- Register your team with your tutor

Monday 14 March – Labour day – NOT a public holiday

- All classes as normal

Database Development Lifecycle: Review



The Entity-Relationship Model

Basic ER modeling concepts

Constraints

Conceptual Design

Readings: Chapter 2, Ramakrishnan & Gehrke, Database Systems

Conceptual Design: Objectives

What are the *entities* and *relationships* in the enterprise?

What information about these entities and relationships should we store in the database?

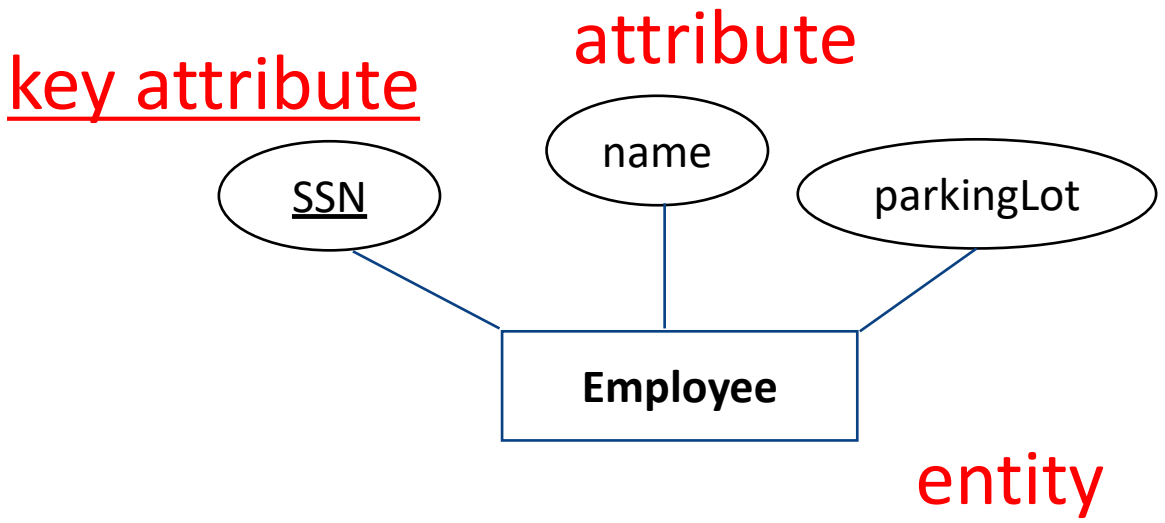
What are the *integrity constraints* that hold?

ER Model: Entity and its attributes

Entity: Real-world object distinguishable from other objects. An entity is described (in DB) using a set of *attributes.*

***Entity Set*:** A collection of entities of the same type (e.g. *all employees*)

- All entities in an entity set have the same set of attributes
- Each entity has a unique *key (underlined)*



Attributes

Attributes have a domain, i.e. the set of possible values for a given attribute

Domain is described by the company's business rules

Examples:

- Attribute **Mark** at any Australian University must be **between 0 and 100** inclusive
- Attribute **Grade** at the University of Melbourne can only have one of the following values H1, H2A, H2B, H3, P, N, NS
- Attribute **Grade** at Deakin or Swinburne Universities can only have one of the following values HD, D, C, P, N
- Attribute **Name** can contain only alphabetical characters and dashes

Attributes may share a domain

- Phone numbers of staff and customers adhere to the same rules, therefore sharing the same domain

Required attributes – the attributes that must have a value

Optional attributes may be left blank

Attributes (cont)

Composite vs **Simple** attributes

A **composite attribute** can be further subdivided into additional attributes

- *Address* contains *street number*, *street name*, *city*, *state*, *postcode*, *country*

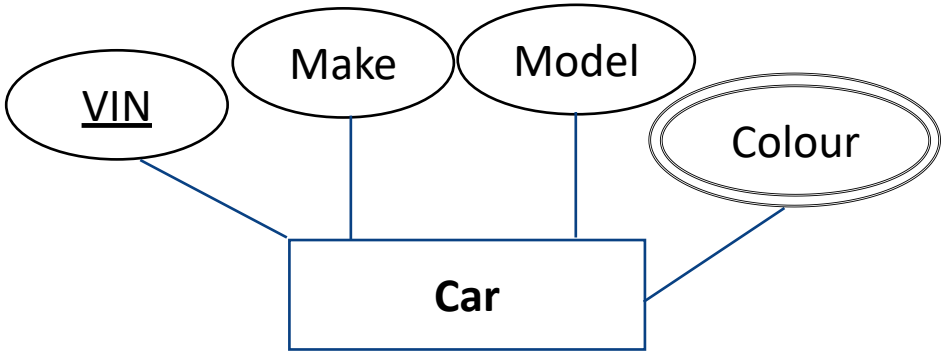
A **simple attribute** cannot be subdivided further, e.g. *gender*, *age*, *year of birth* (but not *date of birth*)

Single-valued vs **Multi-valued** attributes

A **single-valued attribute** can have only a single value, e.g. *year of birth*, *car rego*, *email address*

A **multi-valued attribute** can have many values, e.g. a car may have several colours, one for a roof, another for body and another for trim

In Chen's notation a multi-valued attribute is shown with a double line border (in some notations connected by double line)



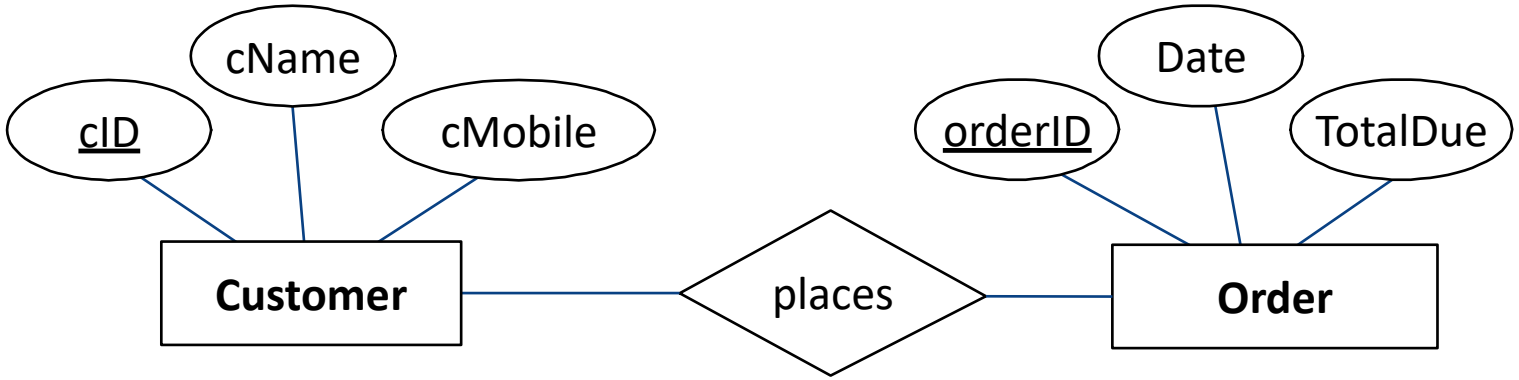
ER Model: Relationship

Relationship: Association among two or more entities.

- Example: John *places* a Pizza order.

***Relationship Set*:** Collection of relationships of the same type.

- Example: Customers *place* orders.



relationship set

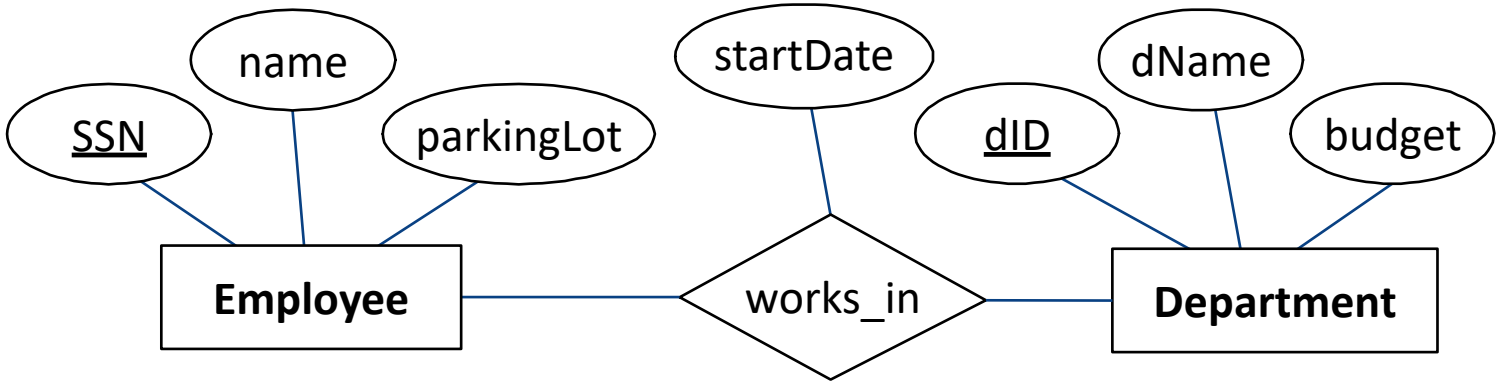
ER Model: Relationship

Relationship: Relationships can have their own attributes.

- Example: John *works in* the Pharmacy department starting 1 July 2021.

***Relationship Set*:** Collection of relationships of the same type.

- Example: Employees *work in* departments.

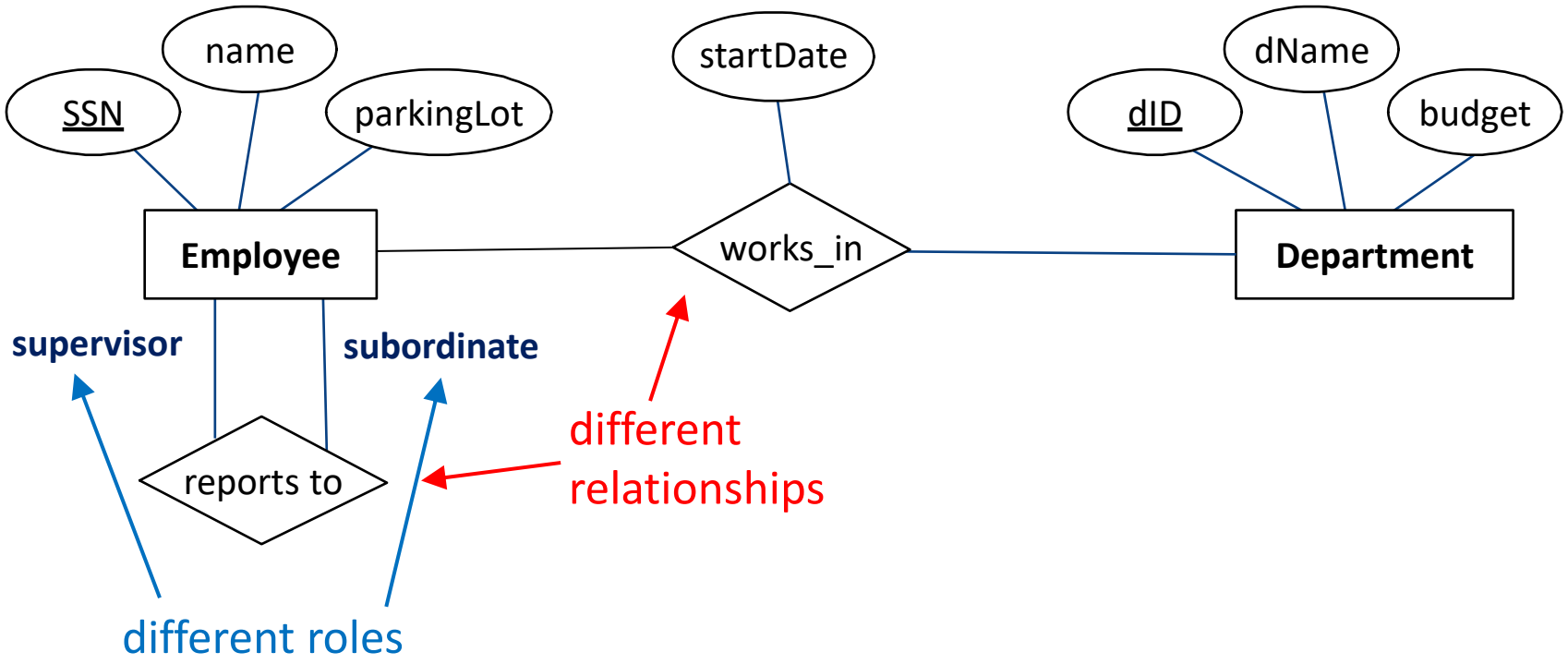


relationship set (with an attribute)

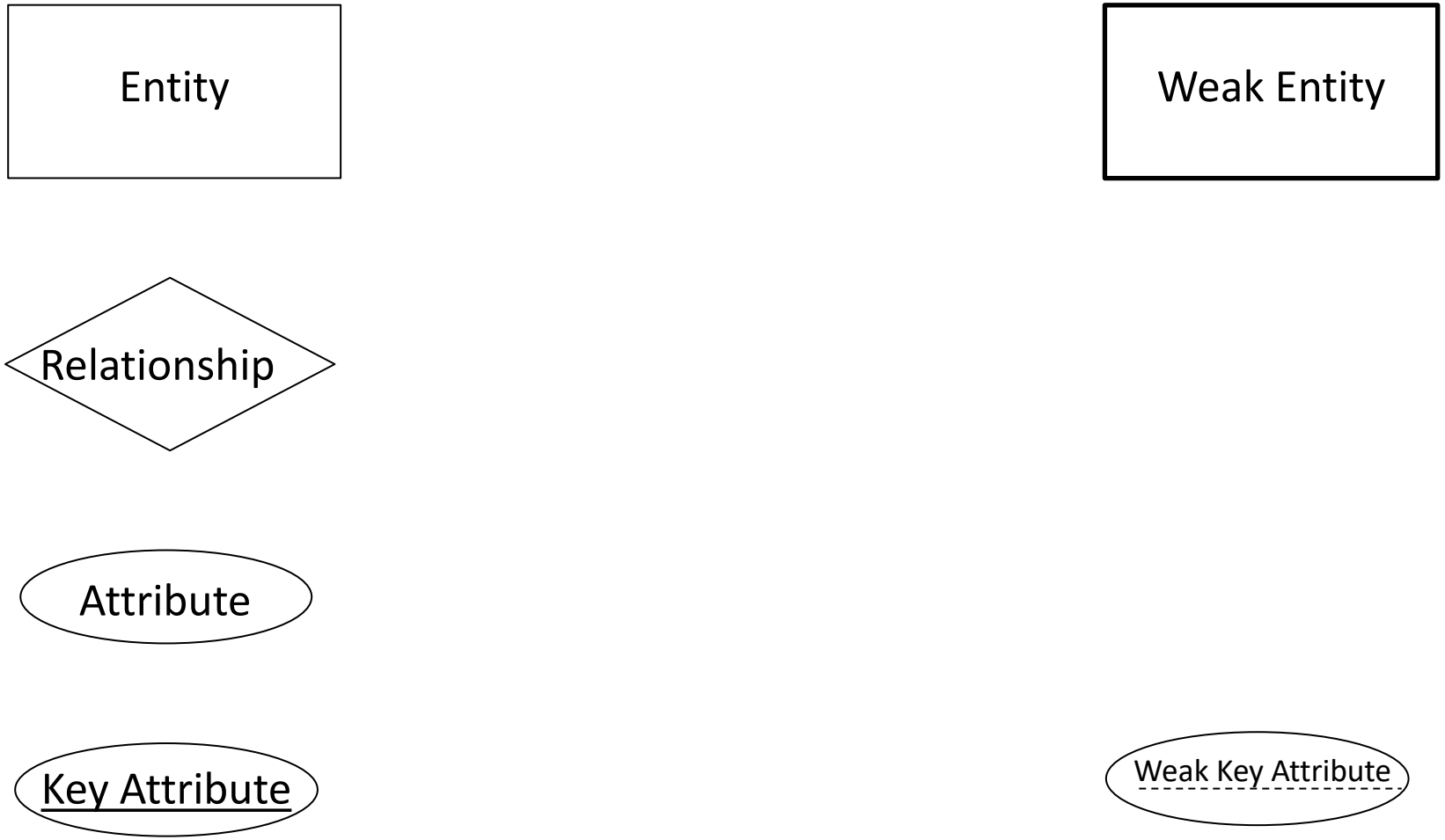
ER Model: Relationship roles

Same entity set can participate in:

- *different* relationship sets, or even
- *different "roles"* in the same set



Chen notation key shapes



The Entity-Relationship Model

Basic ER modeling concepts

Constraints

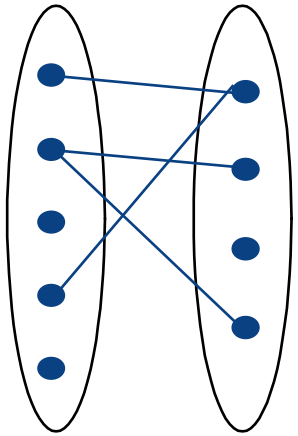
Conceptual Design

Readings: Chapter 2, Ramakrishnan & Gehrke, Database Systems

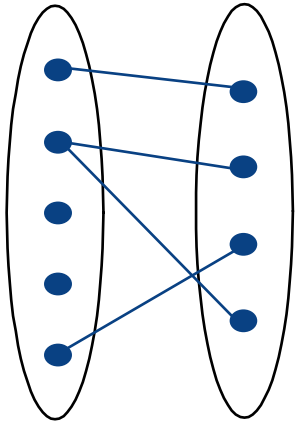
Key Constraints: Types

Key constraints determine the number of objects taking part in the relationship set (how many from each side)

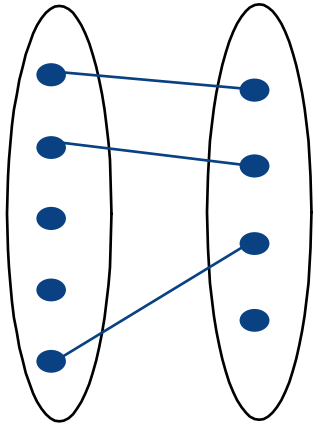
Types of key constraints:



Many-to-Many



One-to-Many



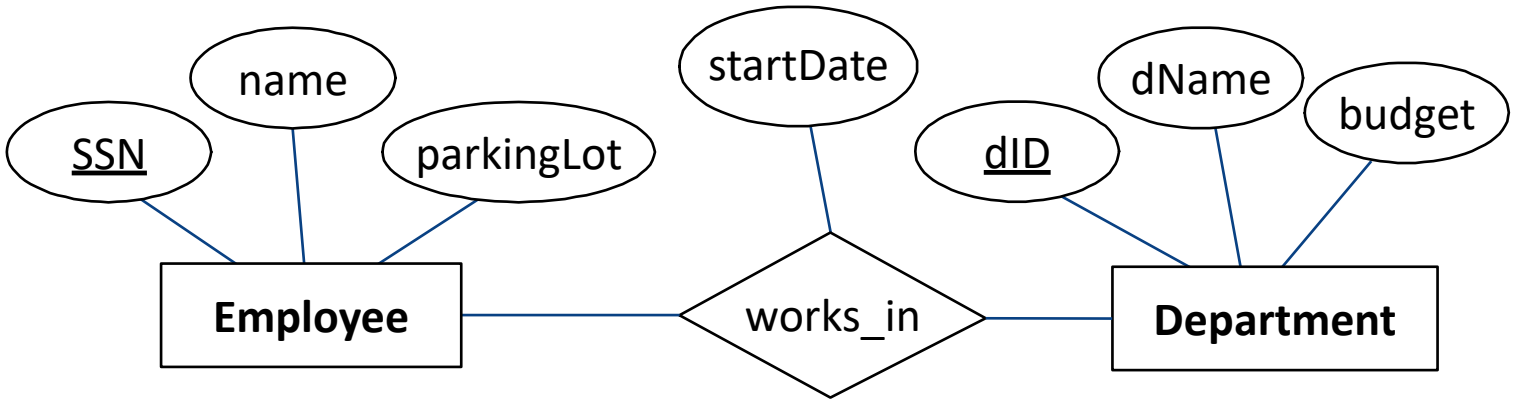
One-to-One

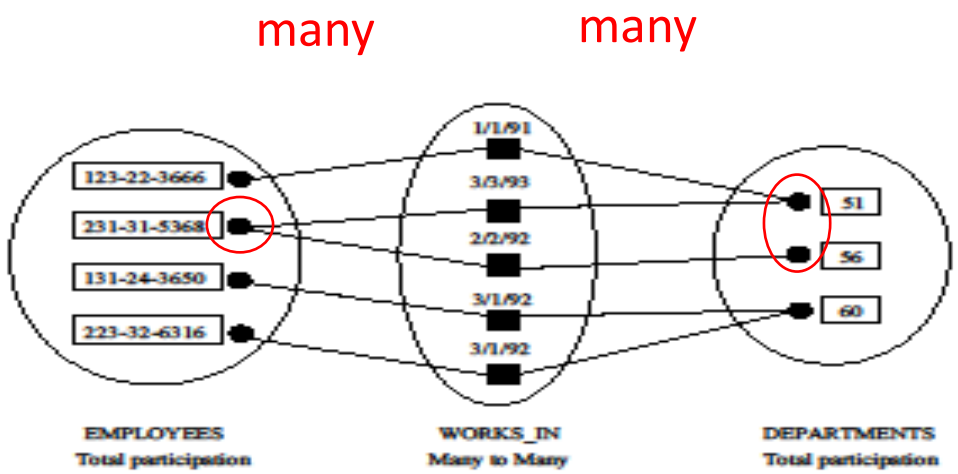
Key Constraints: Many-to-Many

Example:

An employee can work in many departments; a department can have many employees.

Many is represented by a *line*.





Employee works in many departments
A department has many employees

Key Constraints: One-to-Many

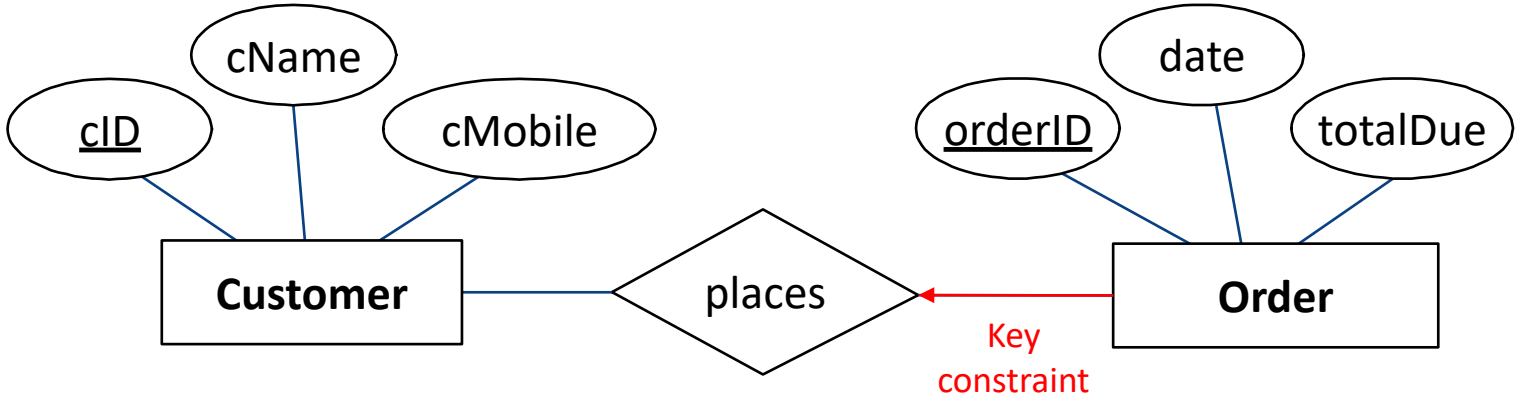
One-to-many constrains one entity set to have a *single* entity per a relationship. An entity of that set can never participate in two relationships of the same relationship set. This is called a **key constraint** and is represented by an *arrow*.

Example:

A customer can place many orders

Each order belongs to at most one customer.

This is the key constraint on *Places*.

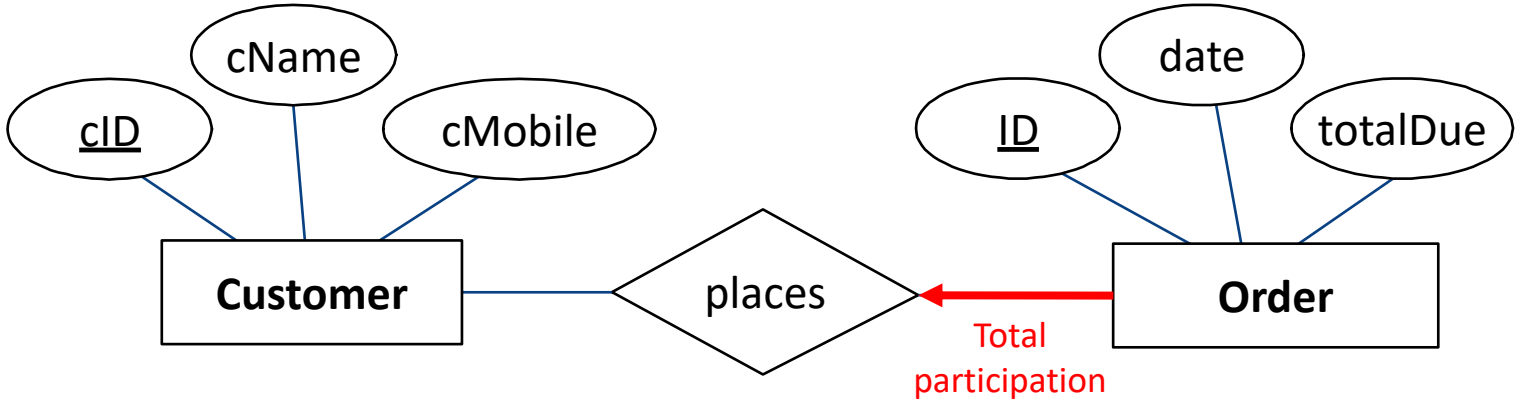


Participation Constraints

Participation constraint explores whether all entities of one entity set take part in a relationship. If yes this is a **total** participation, otherwise it is **partial**. Total participation says that each entity takes part in "**at least one**" relationship and is represented by a **bold line** (in some books – double line).

Example: *Every order must belong to a customer, i.e. an order cannot exist on its own without being placed by a customer.* The participation of *order* in *places* relationship is **total participation**.

A customer may register with the system (becomes a customer) but *do not have to place an order*. So the participation of *customer* in *places* relationship is **partial participation**.



Strong and Weak entities

An entity is a person, place, object, event or concept.

Entities can be classified into two groups

- **Strong** Entities
- **Weak** Entities

A **strong entity** is :

"An entity type that exists independently of other entity types" *(Hoffer Prescott McFadden)*

All previous examples had **Strong Entities**

- Department, Employee
- Customer, Order

A **strong entity** can be identified by its own attributes, meaning a key attribute (a unique identifier) can be chosen from its own attributes

Weak entity

A weak entity is an entity that meets 2 conditions

- The entity cannot exist without the entity with which it has a relationship
- The entity has a unique identifier that is partially or totally derived from the parent entity in the relationship

Owner entity set and weak entity set must participate in a *one-to-many* relationship set (one owner, many weak entities)

Weak entities are represented by double border rectangle

Weak entity set must have total participation in this relationship set. Such relationship is called *identifying* and is represented as “bold” or double-border

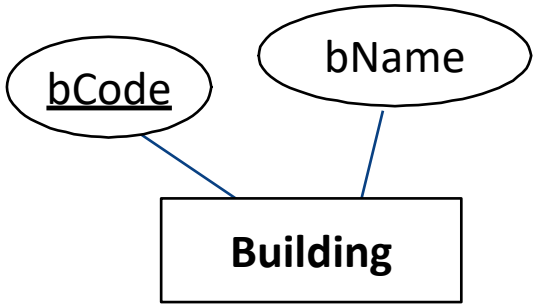
Example

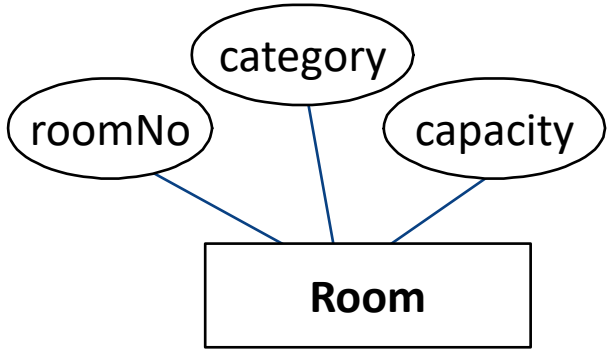
Buildings at a university contain rooms.

Each building have a building name and building code (e.g. Melbourne Connect – MC, Doug McDonnell – DM) – strong entity

Each room has room number (e.g. 4.05, 2.01), room category (staff office, lecture theatre, lab) and capacity

Can the attribute roomNo be used as a unique identifier? No, because several buildings have the same room numbers





Example (cont.)

Relationship:

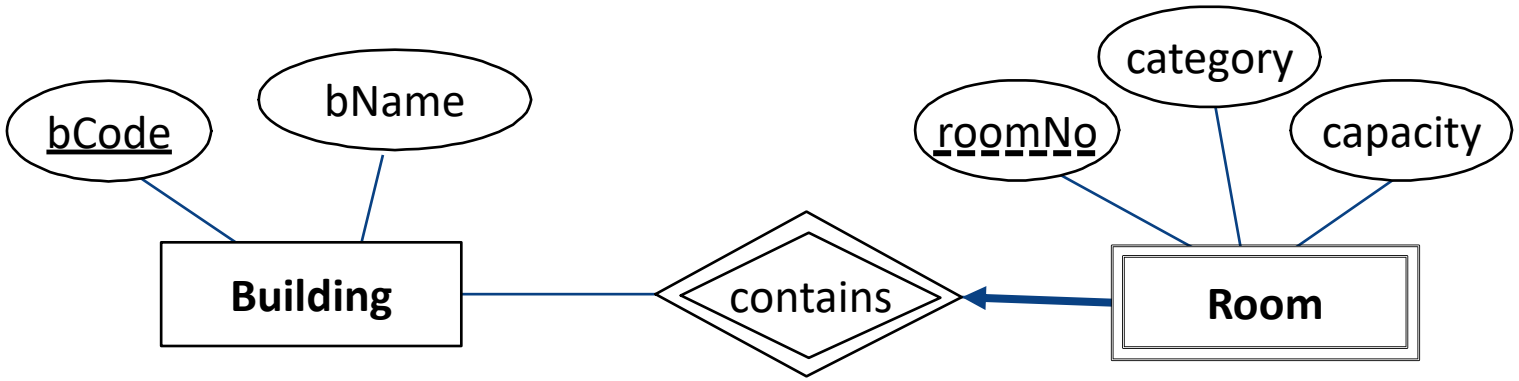
- One building contains many rooms
- A room is contained in one building only

So we need a composite identifier for the room entity, i.e. bCode+roomNo

Problem: bCode is not an attribute of room

Solution: Room borrows identifier from building

Room is a **weak** entity



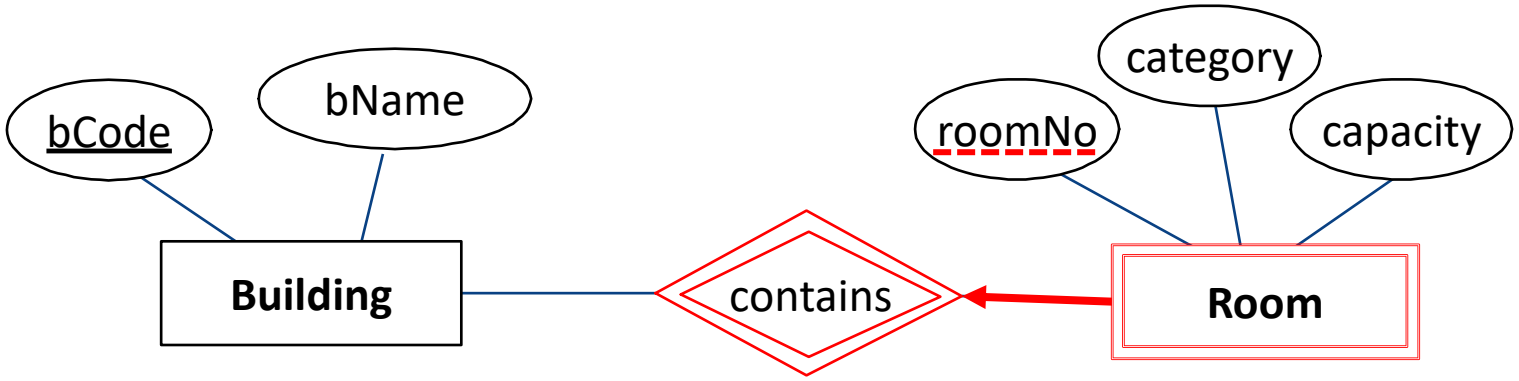
Weak entity notation

Entity denoted by a double border rectangle

If a weak entity has an attribute that is a partial identifier, it is marked with a dashed underline

The relationship is shown with a double border and is called an identifying relationship

Total participation is denoted with a bold arrow



Weak entity

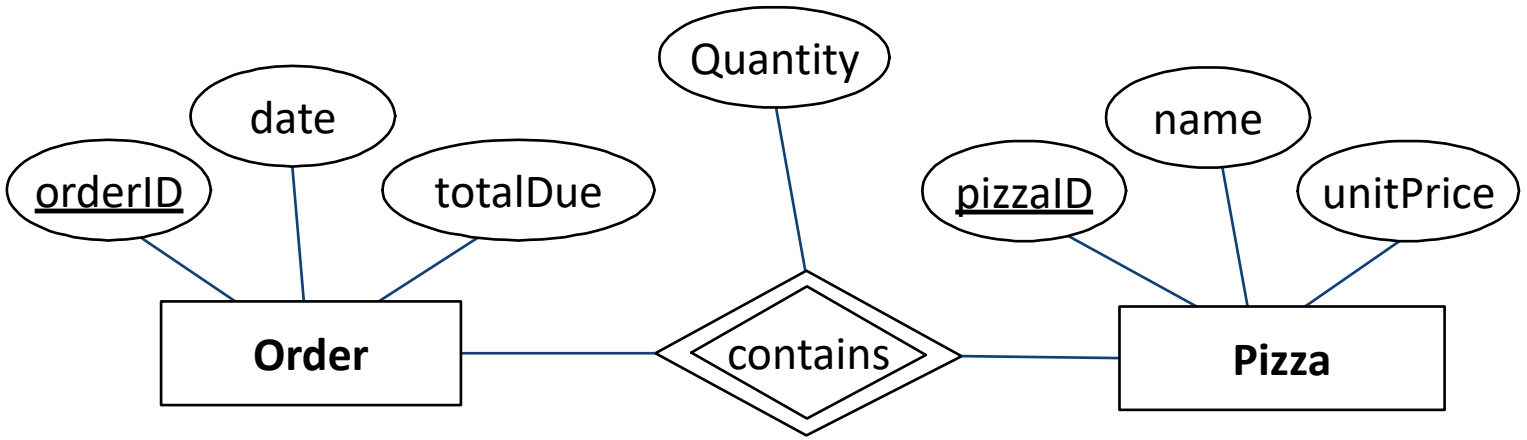
An order contains pizza(s)

Each pizza can be in many orders

} many-to-many

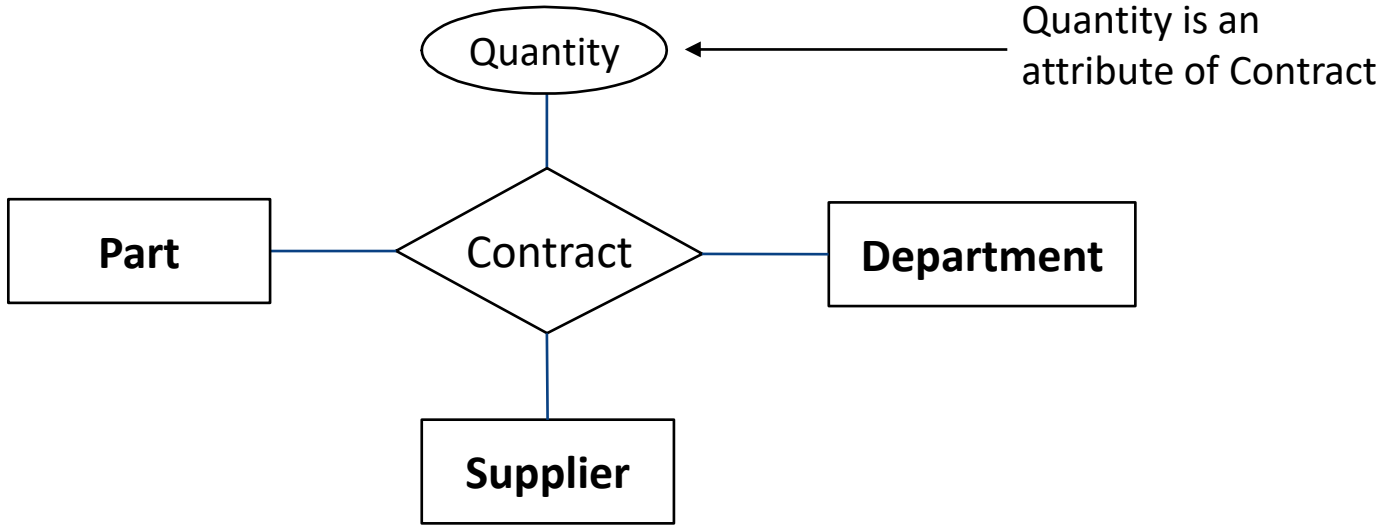
Problem: where do we put quantity for each pizza type ordered?

Solution:



Ternary Relationships

In general, we can have **n**-ary relationships, and relationships can have attributes



This is a ternary relationship with one relationship attribute.

Hint: Count the number of entities to determine the n-ary relationship name

"Departments will use one or more suppliers to obtain one or more parts for producing the product"

The Entity-Relationship Model

Basic ER modeling concepts

Constraints

Conceptual Design

Readings: Chapter 2, Ramakrishnan & Gehrke, Database Systems

Conceptual Design Using the ER Model

Design choices:

- Should a concept be modelled as an entity or an attribute?
- Should a concept be modelled as an entity or a relationship?
- Should we model relationships as binary, ternary, n-ary?

Constraints in the ER Model:

- A lot of data semantics can (and should) be captured

Entity vs Attribute

Example:

Should "*address*" be an attribute of Employees or an entity (related to Employees)?

Answer:

Depends upon how we want to use address information, and the semantics of the data:

- If we have several addresses per employee, *address* must be an entity
- If the structure (city, street, etc.) is important, *address* should be modeled as an entity

See Ramakrishnan & Gehrke p.41

Notes on the ER design

ER design is *subjective*. There are often many ways to model a given scenario!

Where to start – noun-verb analysis

Analyzing alternatives can be tricky, especially for a large enterprise. Common choices include:

- Entity vs. attribute, entity vs. relationship, binary or n-ary relationship.

(See Ramakrishnan & Gehrke p.41-45 sections 2.5.1-2.5.3)

There is no standard notation (we will cover two notations, so far we learned **Chen's** notation)

Community Toy Library – noun-verb analysis

Sample Business Narrative:

Community Toy Library has many **Members**. The library records a member's **name** and **email**. Each new Member is assigned a sequential **member number**.

Over years the library acquired a number of **toys**. Every toy in the library has a unique **toy number.** Each toy has a **description** and **year** of acquisition.

Each time a toy is loaned to a member, the **date of the loan** is recorded. When the toy is returned, the **returned date** is recorded and the number of **days borrowed** is determined.

Each loan is for a single toy (for now).

Identify Entities and Attributes

Element	Circle best alternative
Community Toy Library	StrongEntity / WeakEntity / Attribute / Calculated / Ignore
Member	StrongEntity / WeakEntity / Attribute / Calculated / Ignore
MemberNo	StrongEntity / WeakEntity / Attribute / Calculated / Ignore
Firstname	StrongEntity / WeakEntity / Attribute / Calculated / Ignore
Surname	StrongEntity / WeakEntity / Attribute / Calculated / Ignore
Member email	StrongEntity / WeakEntity / Attribute / Calculated / Ignore
Toy	StrongEntity / WeakEntity / Attribute / Calculated / Ignore
ToyNo	StrongEntity / WeakEntity / Attribute / Calculated / Ignore
Description	StrongEntity / WeakEntity / Attribute / Calculated / Ignore
YearAquired	StrongEntity / WeakEntity / Attribute / Calculated / Ignore
DateBorrowed	StrongEntity / WeakEntity / Attribute / Calculated / Ignore
DateReturned	StrongEntity / WeakEntity / Attribute / Calculated / Ignore
DaysBorrowed	StrongEntity / WeakEntity / Attribute / Calculated / Ignore

Summary of Conceptual Design

Conceptual design follows requirements analysis

- Yields a high-level description of data to be stored

ER model popular for conceptual design

- Constructs are expressive, close to the way people think about their applications
- Originally proposed by Peter Chen, 1976

Note: there are many variations on ER model notation

Basic constructs: *entities*, *relationships*, and *attributes* (of entities and relationships)

Some additional constructs: *weak entities*

What's examinable*

Need to be able to draw conceptual diagrams on your own

- **Given a problem, *determine entities, attributes, relationships***
- **What is key constraint and participation constraint?**
- **Do we need a weak entity?**
- **Determine constraints for the given entities and their relationships**
- **You must use CHEN notation for conceptual models**

* All material is examinable – these are the suggested key skills you would need to demonstrate in an exam scenario

THE UNIVERSITY OF MELBOURNE

Thank you