Ack: Dr. Chui Chun Kit @ HKU

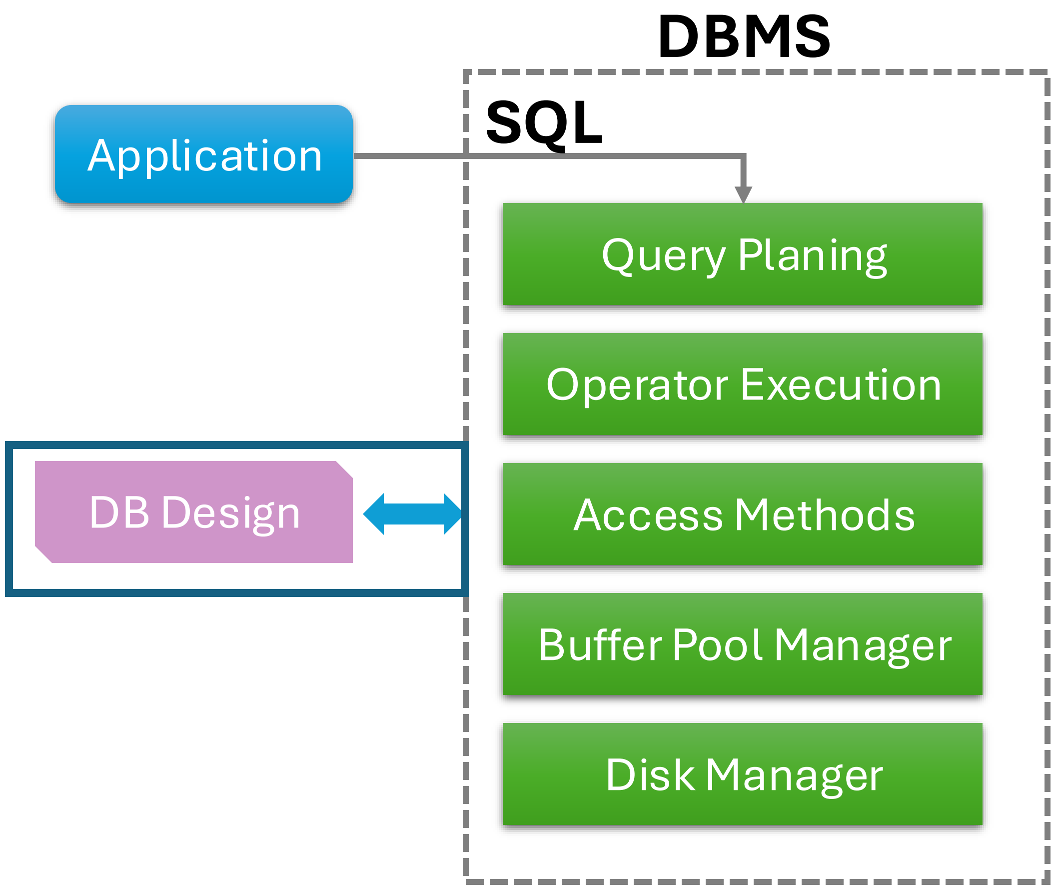
CSC3170

15: DB Design *part a*

Chenhao Ma
School of Data Science
The Chinese University of Hong Kong, Shenzhen

This Lecture

- E-R Diagram
- Functional Dependency



When designing a database, we often use **Entity-Relationship Models** (aka "E-R" models).

E-R Diagram Basics

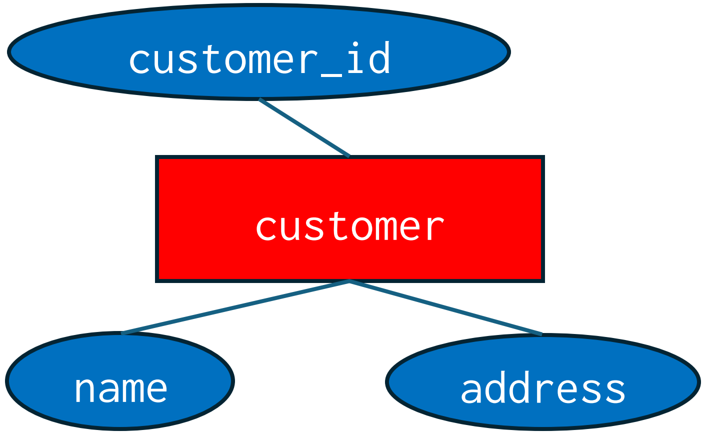
1. Entity and Entity set
2. Relationship and Relationship set
3. Constraints
4. Keys

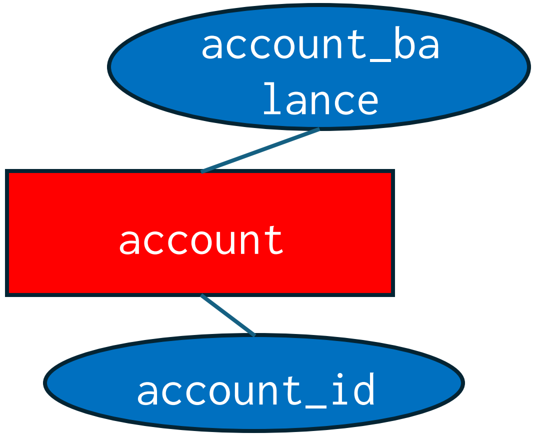
Entity and Entity Set

- **Entity**
  - An object that exists and is distinguishable from other objects.
  - E.g., A customer, an account, a department, etc.
- Entities have **Attributes**
  - People have names and address.
- **Entity set**
  - A set of entities of the same type that share the same properties (attributes).
  - E.g., a set of all customers, all saving accounts, all departments in the company, etc.

Entity and Entity Set

- In the E-R Digram
  - Rectangles – entity sets.
  - Ellipses – attributes.
  - Line between a rectangle and an ellipse – link between an attribute and an entity set.



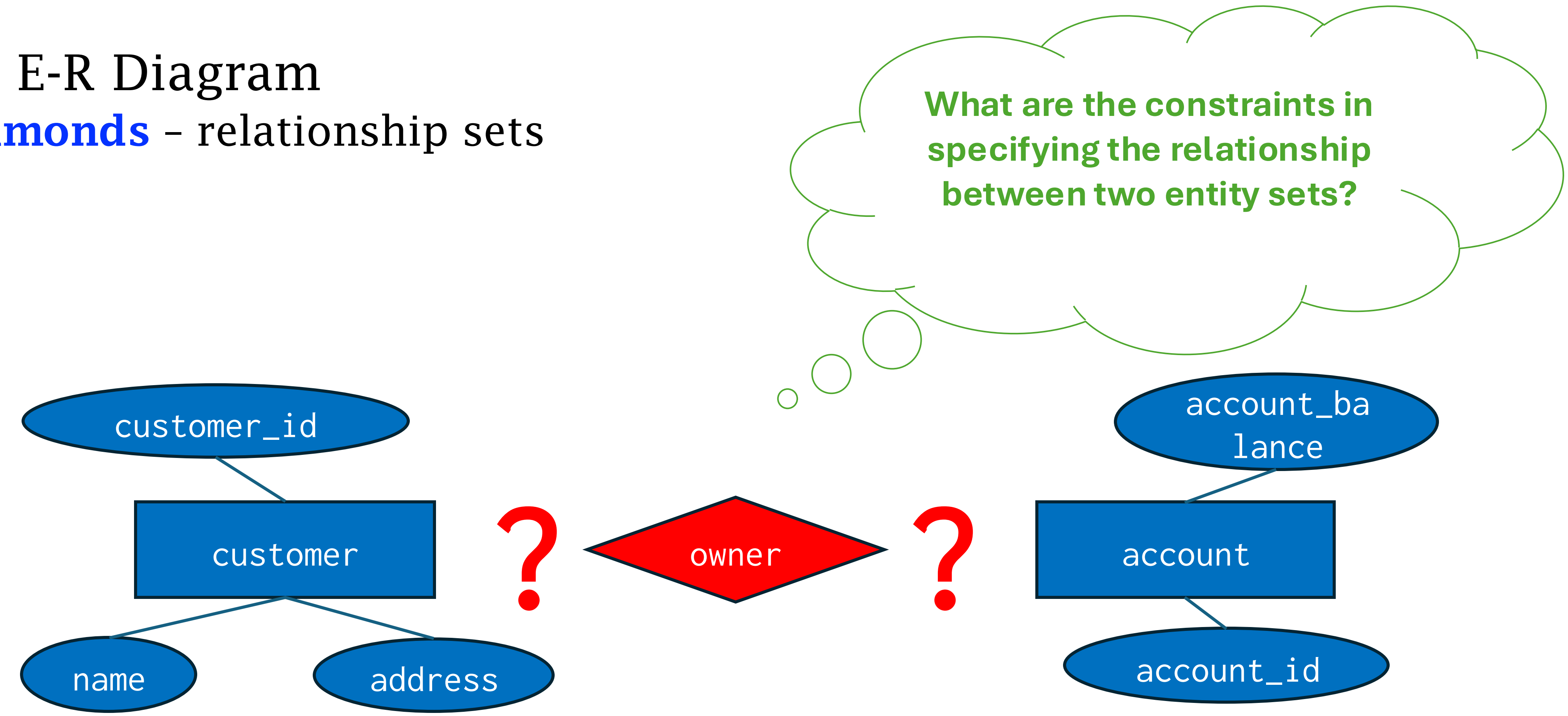


Relationship and Relationship set

- A **relationship** is an association among entities.
  - E.g., the relationship between the customers and the accounts.
- A **relationship set** is a set of relationships of the same type.

Relationship and Relationship set

- In the E-R Diagram
  - **Diamonds** – relationship sets

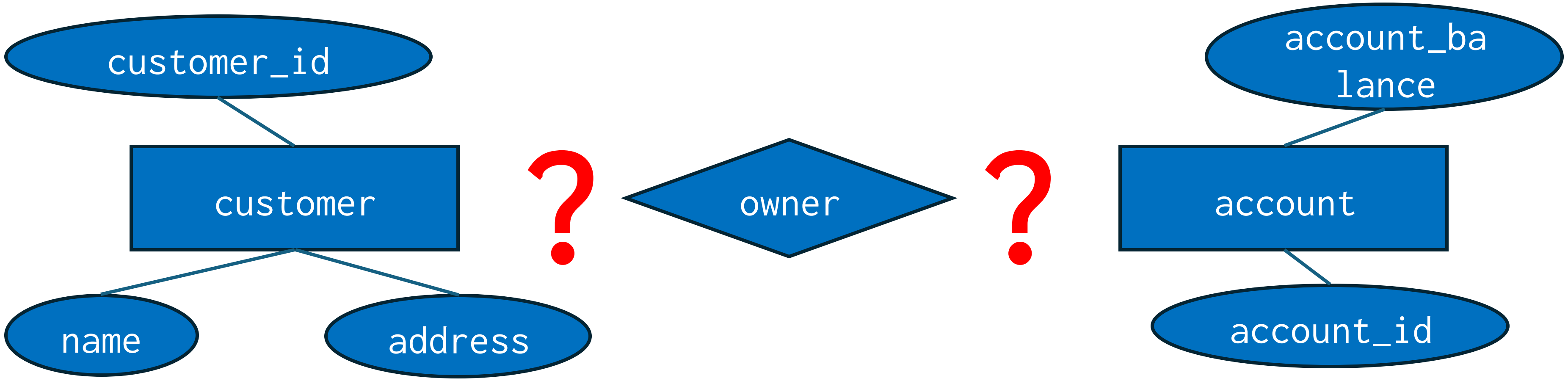


Constraints

- **Mapping cardinalities**
  - Concerns the number of entities to which another entity can be associated via a relationship set.
  - E.g., For each customer, how many accounts he/she can have? **One or more than one?**
- **Participation constraints**
  - Concerns whether all entities in the entity set have to participate in the relationship set.
  - E.g., Whether a customer **must have** an account record, or there can be some customers **without** any accounts?

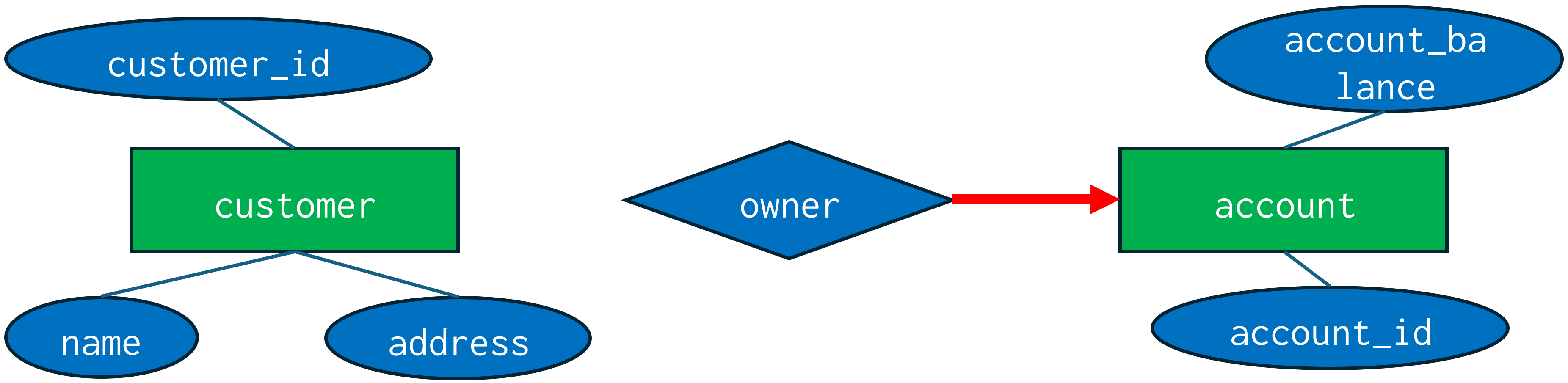
Mapping Cardinalities

- We express **cardinality constraints** by drawing either **a directed line (→)**, signifying "**one**," or **an undirected line (—)**, signifying "**many**," between the relationship set and the entity set.



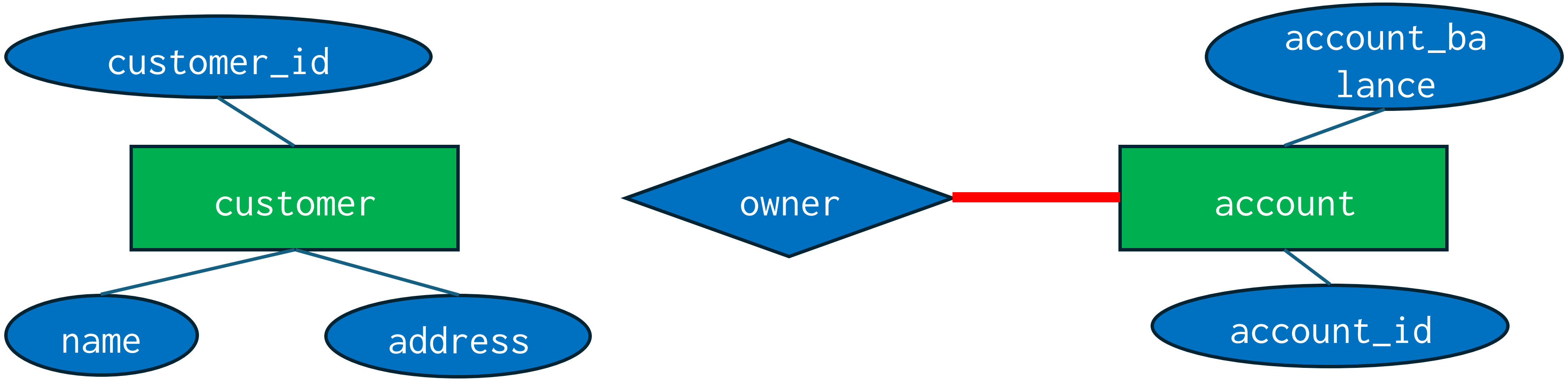
Mapping Cardinalities

- A **customer** can have at most **one** **account**.



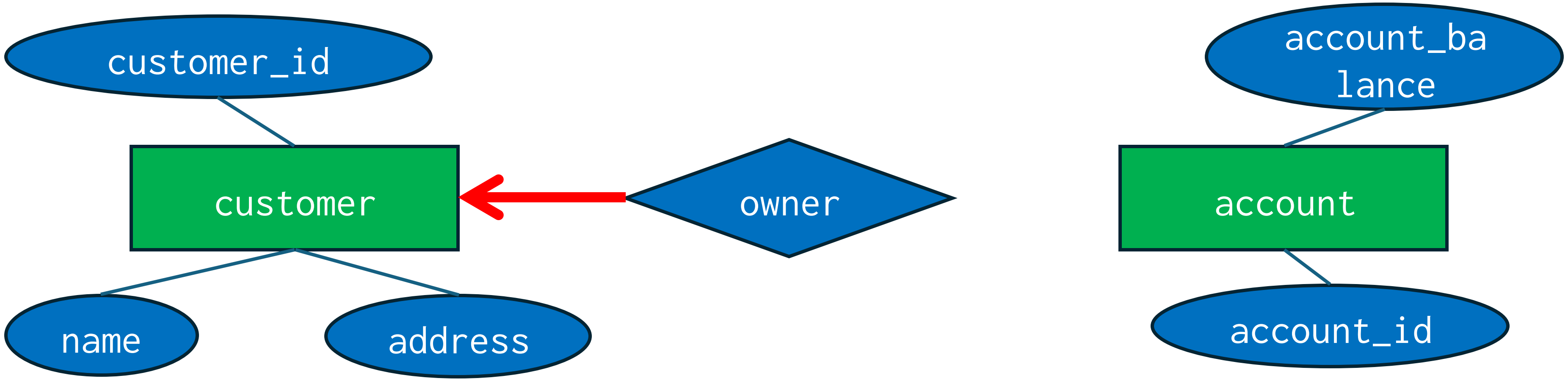
Mapping Cardinalities

- A **customer** can have **more than one** **account**.



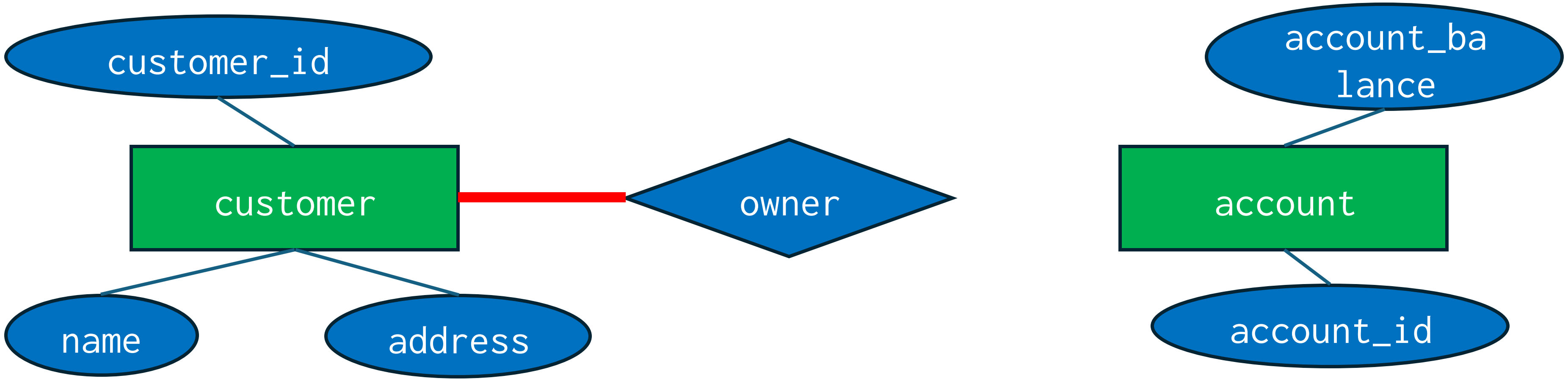
Mapping Cardinalities

- An **account** can be associated with at most **one** **customer**.



Mapping Cardinalities

- An **account** can be associated with **more than one customer**.



Mapping Cardinalities

Please build a system to store the **customer** and **account** information of our bank. For each customer, we record his/her **customer ID**, **name** and **address**; for each account, we record its **account ID** and **account balance**. **Each customer can have one or more accounts, and each account has to be owned by only one customer**.

Step1. Identify the Entity sets.

customer

account

Mapping Cardinalities

Please build a system to store the **customer** and **account** information of our bank. For each customer, we record his/her **customer ID**, **name** and **address**; for each account, we record its **account ID** and **account balance**. **Each customer can have one or more accounts, and each account has to be owned by only one customer**.

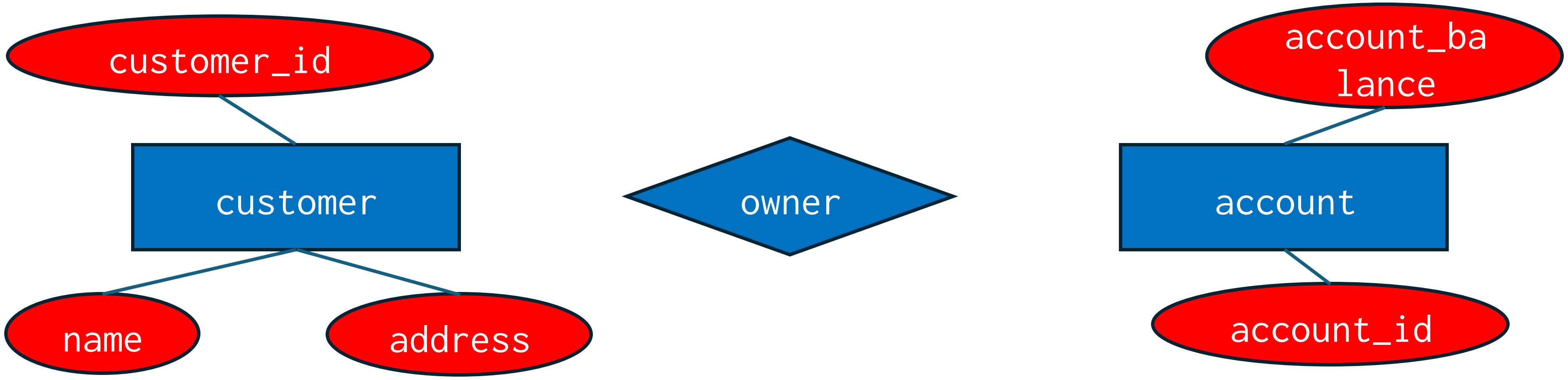
Step2. Identify the Relationship sets.

SCHOOL OF DATA SCIENCE

Mapping Cardinalities

Please build a system to store the **customer** and **account** information of our bank. For each customer, we record his/her **customer ID**, **name** and **address**; for each account, we record its **account ID** and **account balance**. **Each customer can have one or more accounts, and each account has to be owned by only one customer**.

Step3. Identify the attributes.

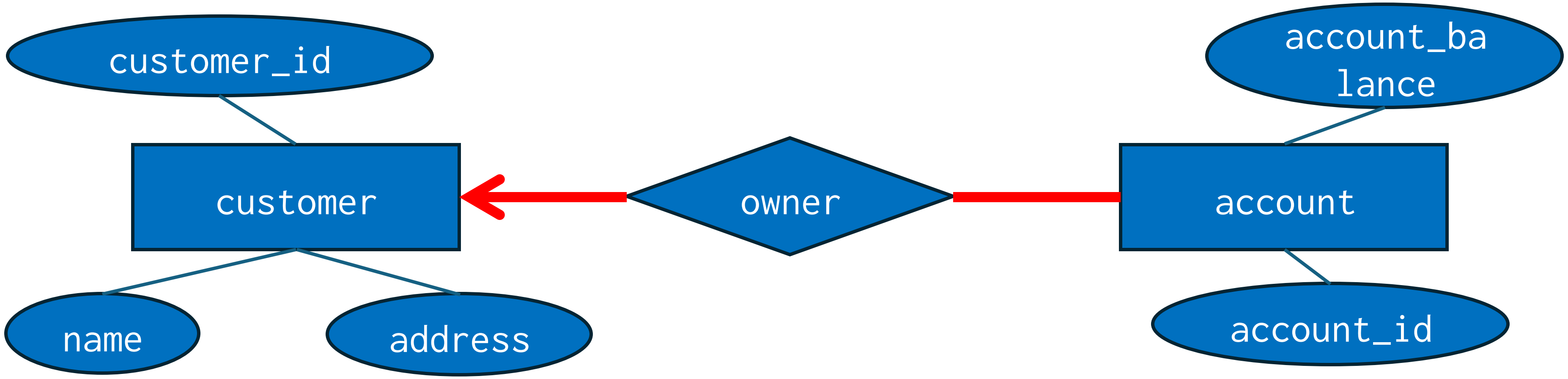


SCHOOL OF DATA SCIENCE

Mapping Cardinalities

Please build a system to store the **customer** and **account** information of our bank. For each customer, we record his/her **customer ID**, **name** and **address**; for each account, we record its **account ID** and **account balance**. **Each customer can have one or more accounts, and each account has to be owned by only one customer**.

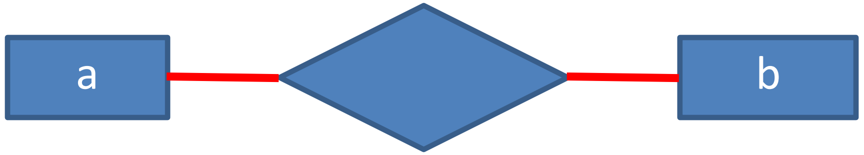
Step1. Mapping cardinalities.



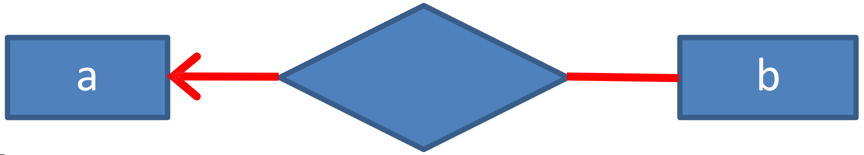
Note: this ER-Diagram is incomplete! Some more steps in the next few slides including the participation, primary keys … etc

Mapping Cardinalities

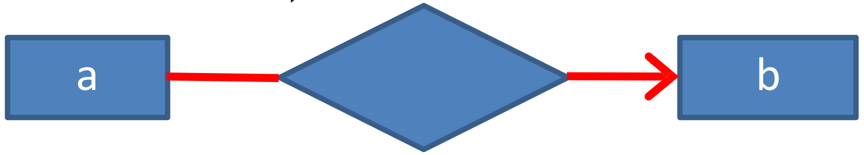
- Different mapping relationships
  - Many to many



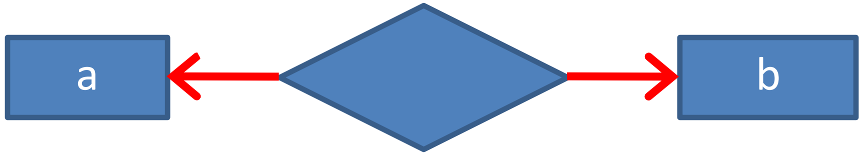
  - One to many (from a to b)



  - Many to one (from a to b)



  - One to one



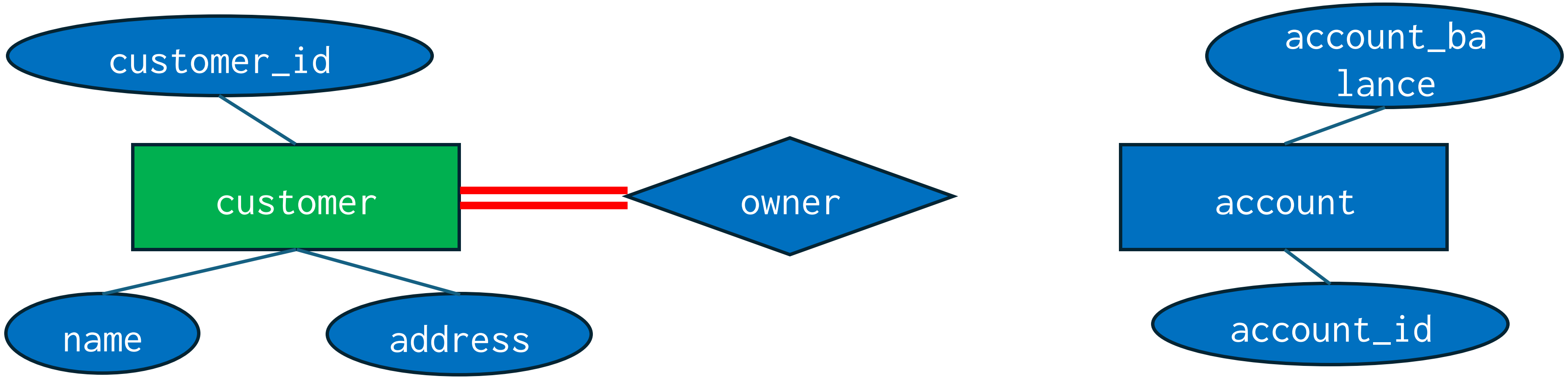
SCHOOL OF DATA SCIENCE

Participation Constraints

- Concerns whether all entities in the entity set have to participate in the relationship.
  - **Total participation** (indicated by double line): every entity in the entity set participates in at least one relationship in the relationship set.
  - **Partial participation** (indicated by single line): some entity may not participate in any relationship in the relationship set.

Participation Constraints

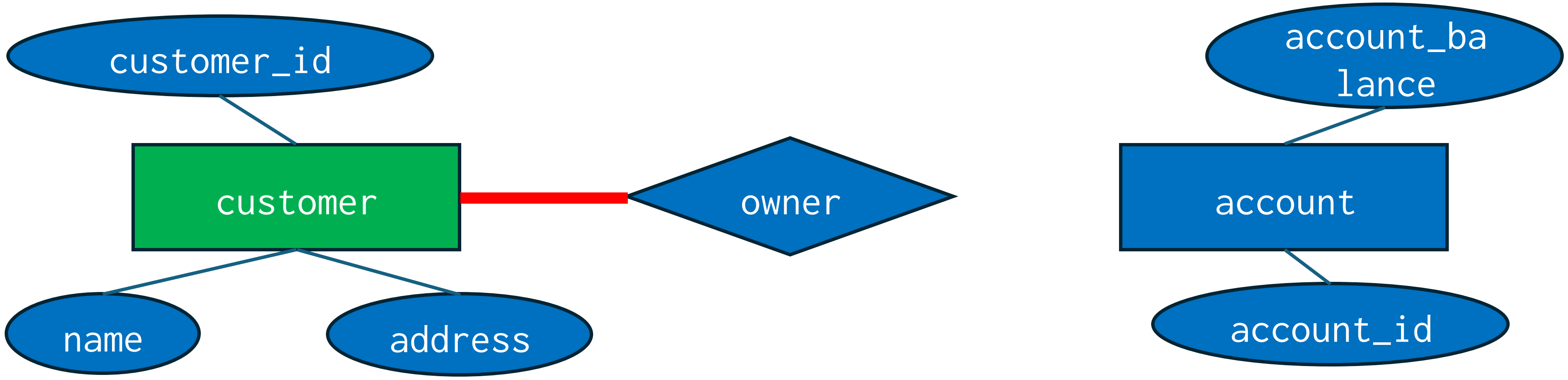
- Each customer **must have an account**.
  - **Total participation**: *All customers* must participate in the owner relationship.



SCHOOL OF
DATA SCIENCE

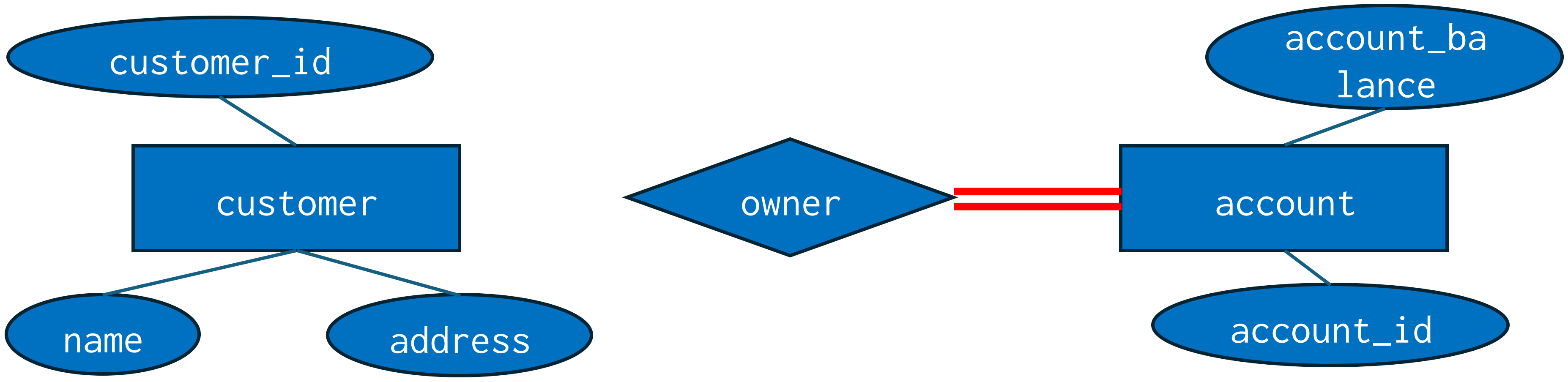
Participation Constraints

- Not all customers are required to have an account.
  - **Partial participation:** *Not all customers* participate in the owner relationship.



Participation Constraints

- Each account must be owned by customers.
  - **Total participation**: *All accounts* participate in the owner relationship.



Exercise

Can you understand the data model captured by this E-R Diagram?

Exercise

- Mapping cardinalities.

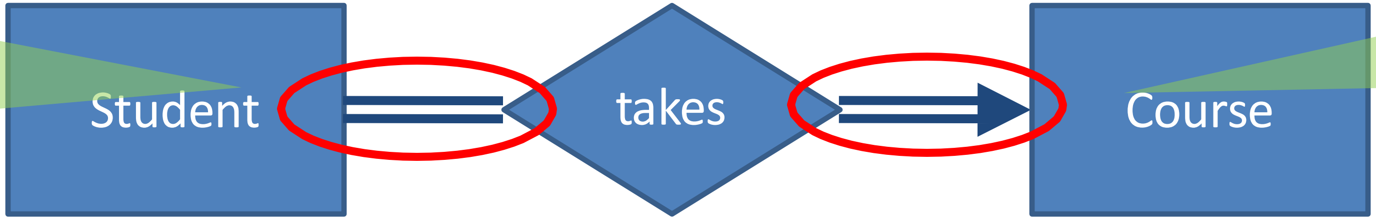
No arrow head (Many)
Each course can be taken by **many students**.

Arrow head (One)
Each student takes **one** course only.

Exercise

- Mapping cardinalities.

No arrow head (Many)
Each course can be taken by **many students**.



Arrow head (One)
Each student takes **one** course only.

- Total / partial participation.

Full participation (double line)
Each entity in the "Student" entity set must participate in this relationship.
(i.e. Each student must take course.)

Full participation (double line)
Each entity in the "Course" entity set must participate in this relationship
(i.e. Each course must be taken by some student).

Practical Issues

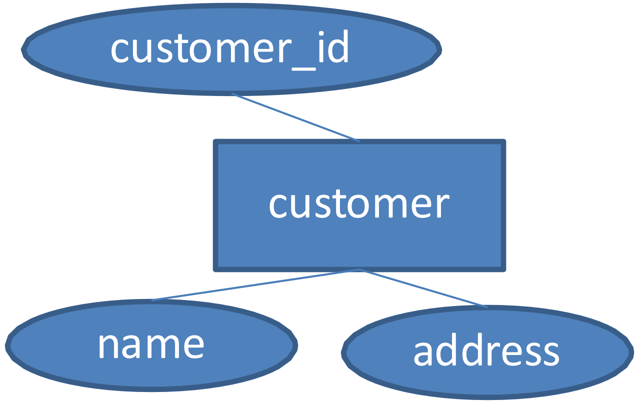
- **As a professional DB designer, you have to be able to:**
  - Understand and model the data of an application using a E-R diagram.
  - Interact with the client to work out a clear problem definition.
  - Realize the missing information and ask your client for clarification.
  - Provide professional suggestions to better design the database for the specific application.

Keys

- **Super key**
- **Candidate keys**
- **Primary key**

Super Key

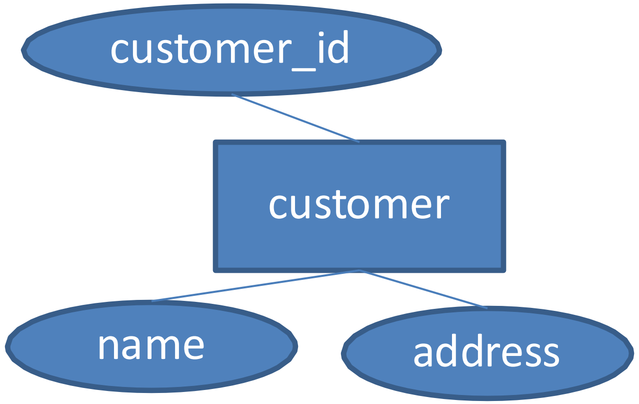
- A **super key** of an entity set is a set of one or more attributes whose values uniquely determine each entity.
  - No two entities have exactly the same values in super key.



In this example, if each customer has his/her **unique customer_id**, then
{customer_id, name} is a super key.
{customer_id, address} is another super key.
{customer_id, name, address} is also a super key.

Candidate Key

- A **candidate key** of an entity set is a **minimal** super key.
  - Minimal – no redundant attributes, i.e., no subset of a candidate key is still a key.

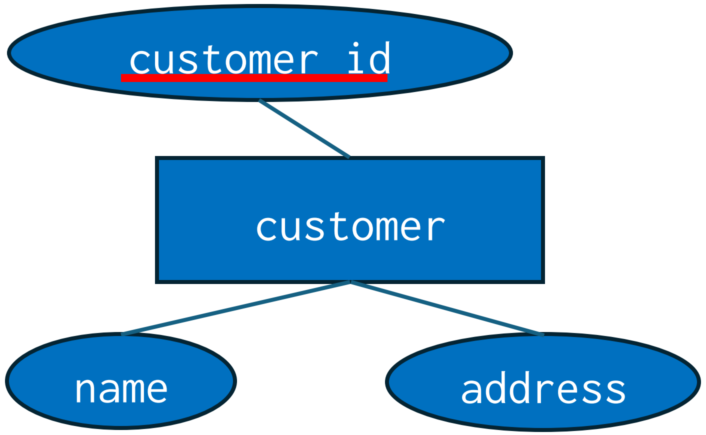


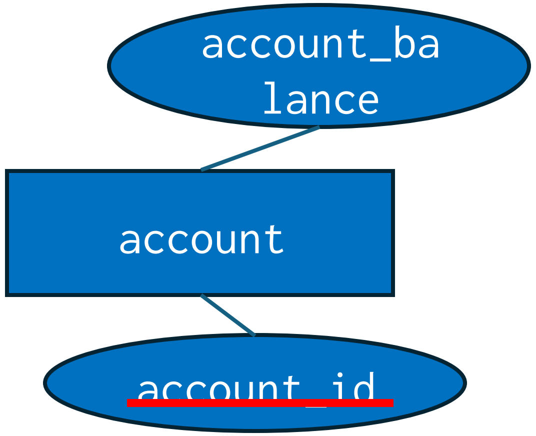
Although the following are super keys:
{ customer_id, name}, {customer_id, address},
{customer_id, name, address}
Only the {customer_id} is a candidate key because it is minimal.

Question: Can it be more than one candidate key?

Primary Key

- Although several candidate keys may exist, one of the candidate keys is selected to be the **primary key**.
- **In the E-R Diagram**
  - **Underline the attribute** – The attribute is a primary key of the entity.





More on E-R Diagram

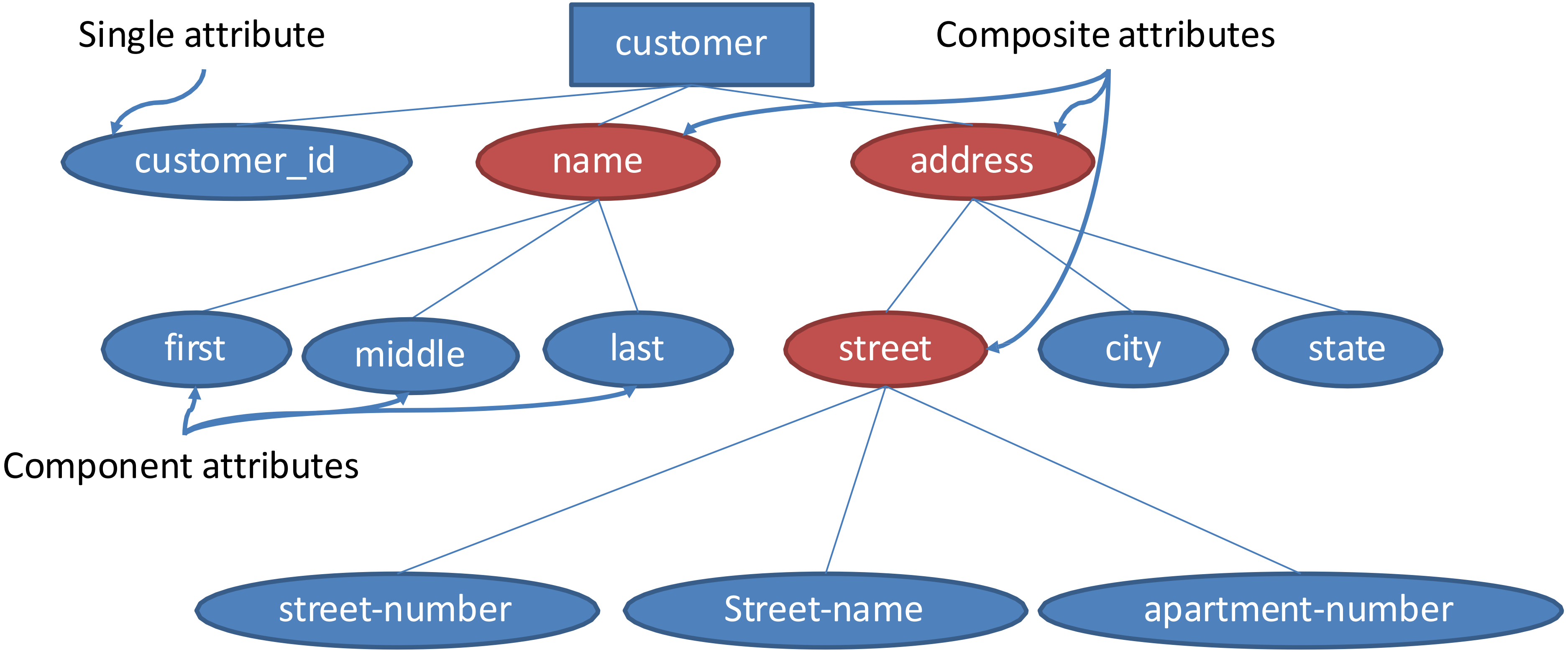
Different attribute types

Role

Specialization and generalization

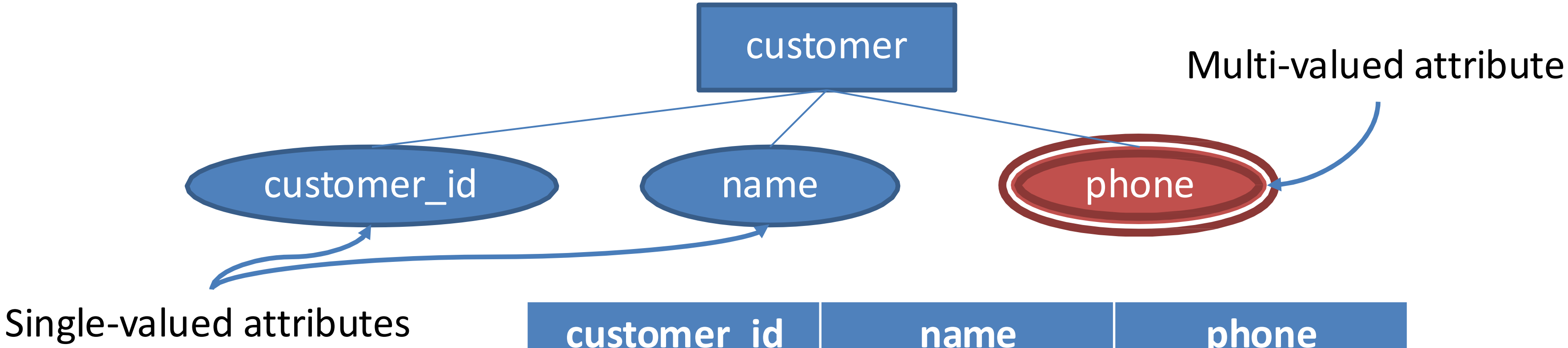
Different Attribute Types

- **Single v.s. Composite attributes**



Different Attribute Types

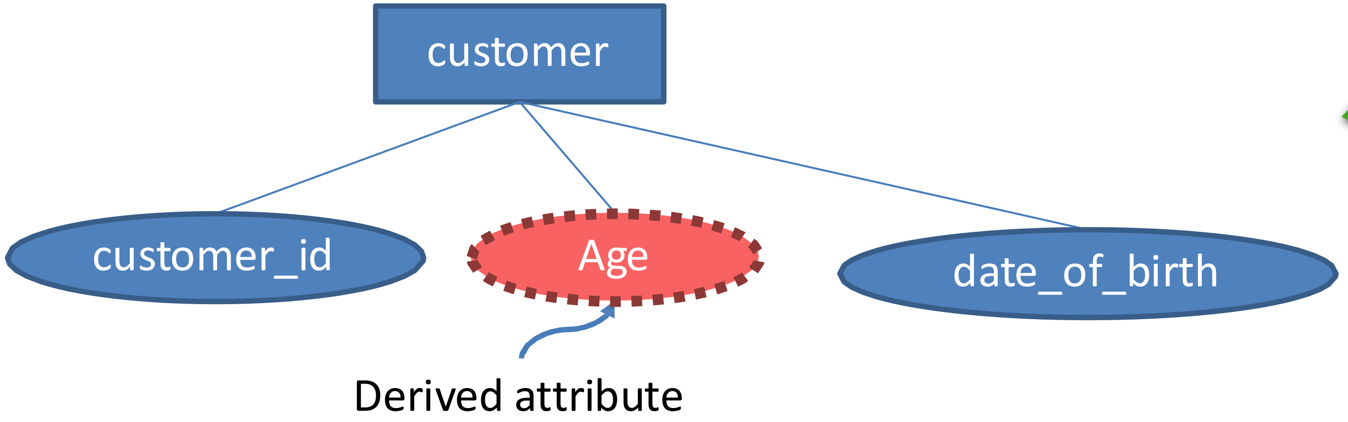
- **Single-valued v.s. Multi-valued attributes**
  - Multi-valued attributes are represented by double ellipses in the E-R Diagram.



customer_id	name	phone
1	Kit	6012 3456, 9888 8888, 2857 8435
2	Yvonne	6987 6543, 2859 1104
3	Jolly	9876 1234, 2857 8434

Different Attribute Types

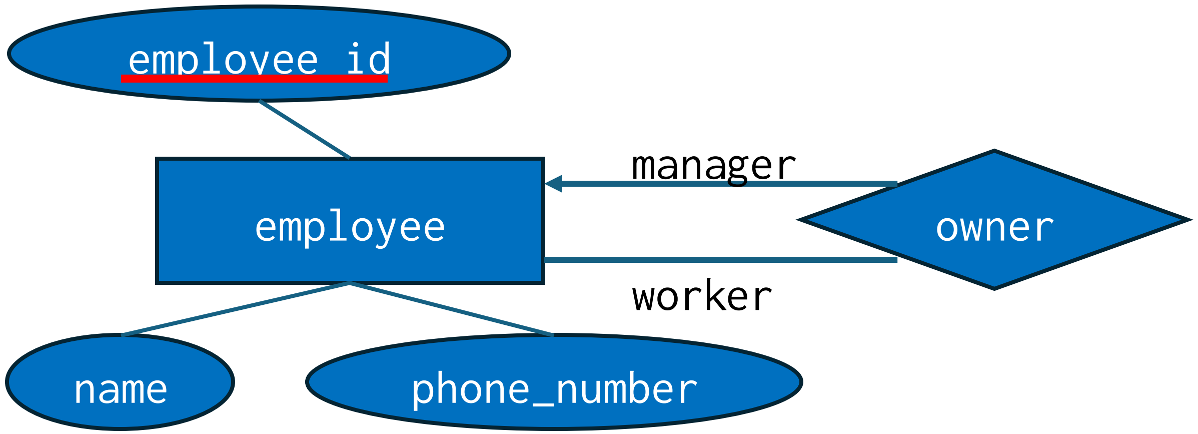
- Derived attribute
  - Values in this attribute can be **derived** from other attributes.
  - Derived attributes are represented by **dashed ellipses** in the E-R Diagram.



Since **“age”** can be derived from the **“date of birth”,** we treat “age” as a derived attribute, and use a dashed ellipse to represent it in the E-R Diagram.

Role

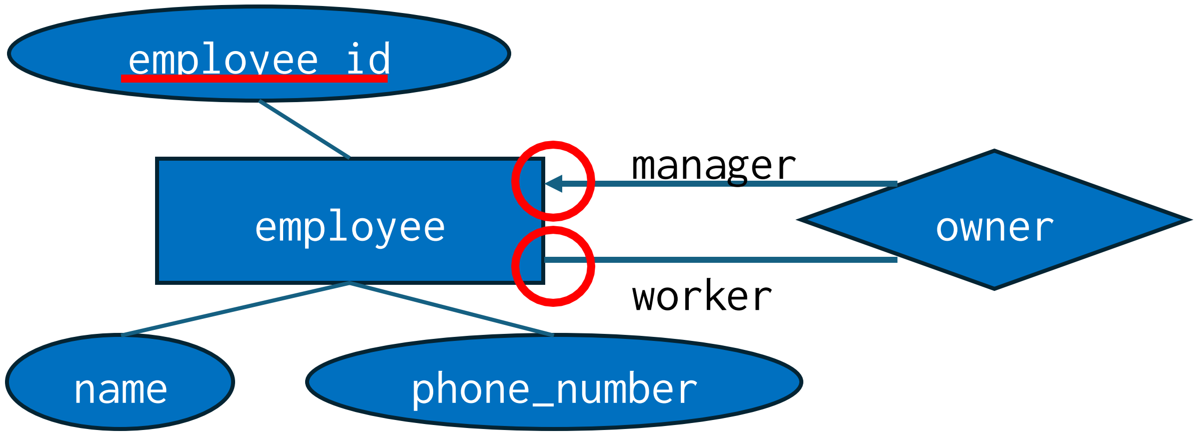
- **Entity sets of a relationship need not be distinct**



- The label "manager" and "worker" are called **roles**. They specify how employee entities interact via the "works-for" relationship set.

Role

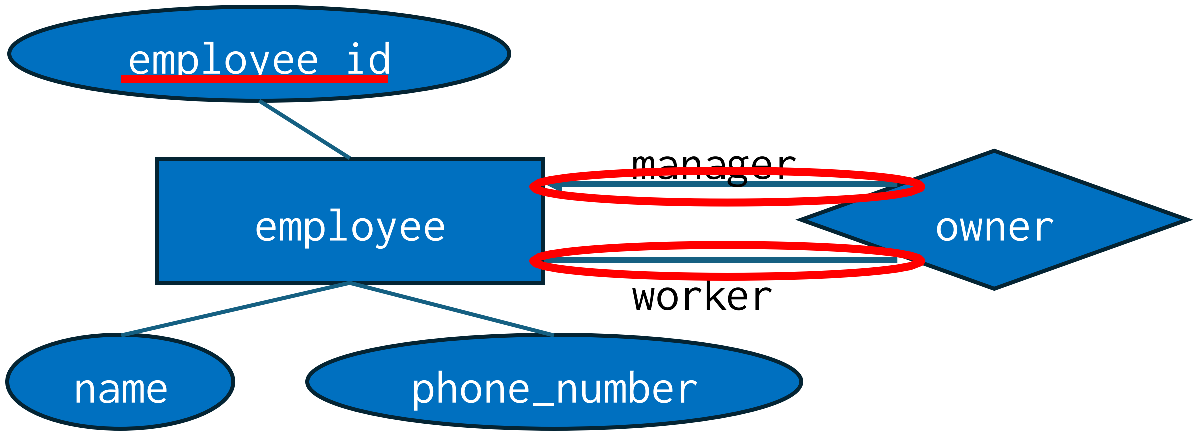
- **Entity sets of a relationship need not be distinct**



- **Cardinality**
  - An employee (worker) works for one manager.
  - An employee (manager) can have more than one workers work for him/her.

Role

- **Entity sets of a relationship need not be distinct**

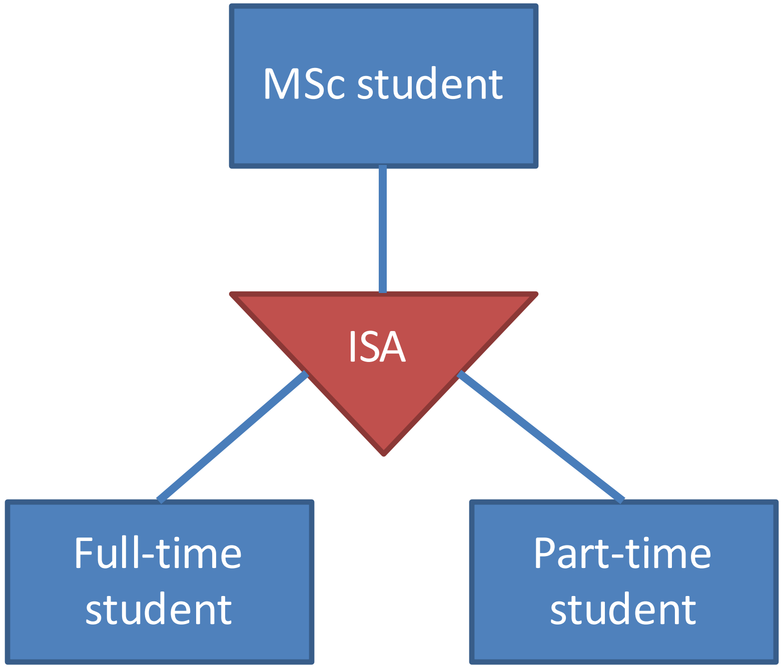


- **Participation**
  - An employee (worker) may not work for any manager.
  - An employee (manager) can have no workers work for him/her.

SCHOOL OF DATA SCIENCE

Specialization and Generalization

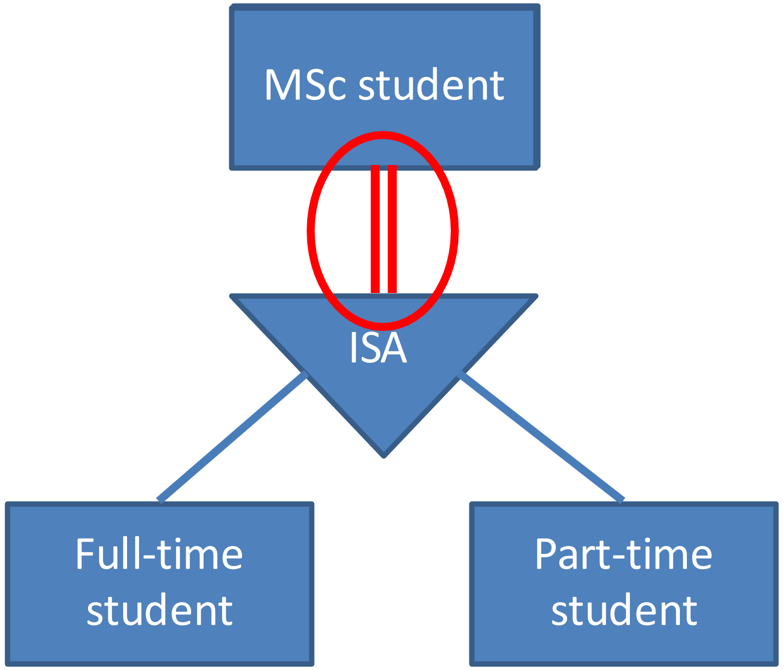
- **Specialization**
  - We designate **sub-groupings** within an entity set that are *distinctive from other entities* in the set.
  - A lower-level entity set inherits all attributes and relationship set participation of the higher-level entity set to which it is linked.
  - Lower-level entity set can have its own attributes.



Specialization and Generalization

- **Total or partial**
  - Specifies whether an entity in the higher level-entity set must belong to at least one of the lower-level entity sets within a specialization.

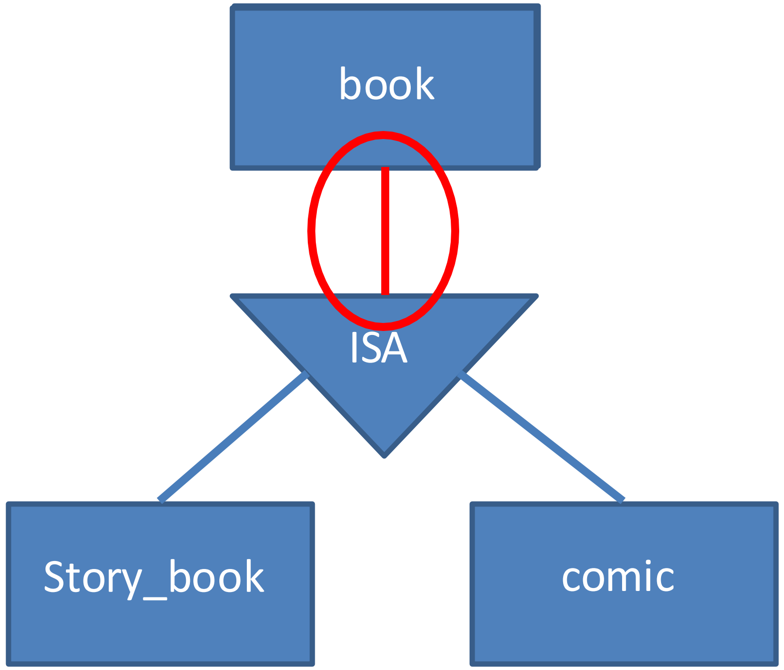
Total specialization:
An MSc student MUST BE either a full-time student or a part- time student, so all MSc students must participate in this specialization.



Specialization and Generalization

- **Total or partial**
  - Specifies whether an entity in the higher level-entity set must belong to at least one of the lower-level entity sets within a specialization.

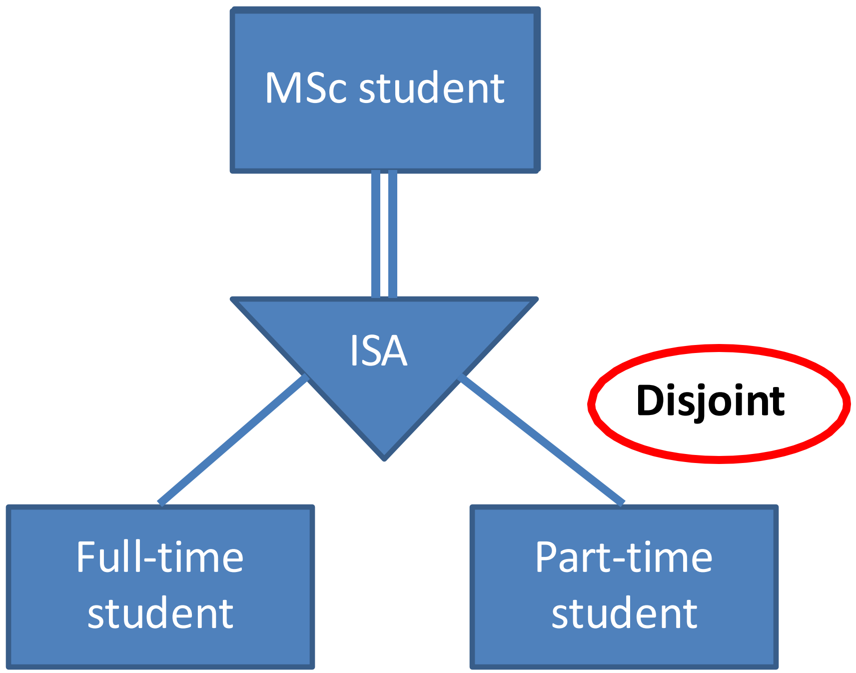
Partial specialization:
A book may not be specialized to story book or comics, so **not all books** are participating in this specialization.



Specialization and Generalization

- **Disjoint or overlapping**
  - Constraints on whether entities may belong to more than one lower-level entity set within a single specialization.

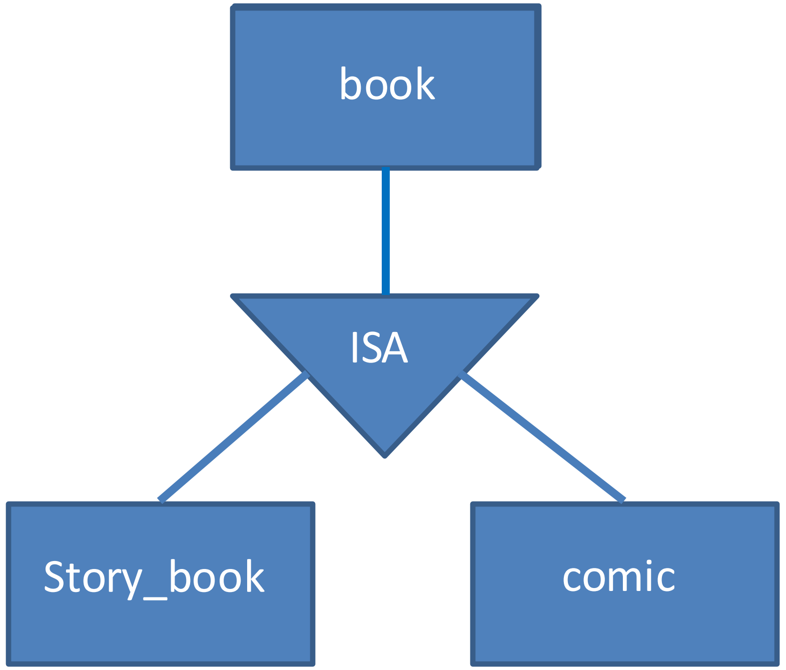
Disjoint specialization:
An MSc student must be either a full-time student or a part-time student, so the specialization is disjoint.
We use a keyword "Disjoint" to indicate it.



Specialization and Generalization

- **Disjoint or overlapping**
  - Constraints on whether entities may belong to more than one lower-level entity set within a single specialization.

Overlapping specialization:
A book can be both a story book and comic, so the specialization is overlapping.
We do not need to specify anything, as overlapping specialization is the default.



Functional Dependency & Normalization

What is FD?

- **Functional dependency (FD)** is a constraint between two sets of attributes in a relation from a database .
- It requires that the values of a certain set of attributes **uniquely determine (imply) the values for** another set of attributes.

$$X \longrightarrow Y$$

Determine attribute set

Dependent attribute set

X->Y means that for two tuples t_1 and t_2, if their values in **X** are the same, then their values in **Y** are also the same.

$$t_1[X] = t_2[X] \Rightarrow t_1[Y] = t_2[Y]$$

What is FD?

- Given a relation R, a set of attributes X in R is said to **functionally determine** another attribute Y, also in R, (written $X \rightarrow Y$) if, and only if, each X value is associated with precisely one Y value.

{employee_id} → {name, phone} ✔

Employees

employee_id	name	phone
1	Jones	62225214
2	Smith	64459574
3	Parker	35564872
4	Smith	28975152

Important concept:
Primary key is just one of the FDs, we can have other FD constraints in the design of a database.

What is FD?

- Given a relation R, a set of attributes X in R is said to **functionally determine** another attribute Y, also in R, (written $X \rightarrow Y$) if, and only if, each X value is associated with precisely one Y value.

{employee_id} → {name, phone} ✓

{phone} → {name} ✓

Employees

employee_id	name	phone
1	Jones	62225214
2	Smith	64459574
3	Parker	35564872
4	Smith	28975152

In the company, each employee has his/her own phone number.
Therefore, the name attribute is functionally determined by the phone attribute.
Each phone number is associated with precisely one name .

What is FD?

- Given a relation R, a set of attributes X in R is said to **functionally determine** another attribute Y, also in R, (written $X \rightarrow Y$) if, and only if, each X value is associated with precisely one Y value.

Employees

employee_id	name	phone
1	Jones	62225214
2	Smith	64459574
3	Parker	35564872
4	Smith	28975152

{employee_id} → {name, phone} ✔

{phone} → {name} ✔

{name} → {phone} ✗

Question
Why is this FD not true?

SCHOOL OF DATA SCIENCE

Exercise

R

A	B	C	D	E
1	5	2	5	4
1	4	3	2	3
3	4	3	2	2
4	2	4	1	4
4	1	4	1	4

Q: Do the following FDs hold?

$A \rightarrow B$ ✗

$B \rightarrow C$ ✓

$C \rightarrow D$ ✓

A	B
1	5
1	4
3	4
4	2
4	1

Armstrong's Axioms

- **3 basic axioms.**
  - 1. **Reflexivity** - if $\beta \subseteq \alpha$, then $\alpha \to \beta$.
  - 2. **Transitivity** - if $\alpha \to \beta$ and $\beta \to \gamma$, then $\alpha \to \gamma$.
  - 3. **Augmentation** - if $\alpha \to \beta$, then $\gamma\alpha \to \gamma\beta$.
- Based on these rules, we can generate/extend more…

Armstrong's Axioms

R

A	B	C	D	E
1	5	2	5	4
1	4	3	2	3
3	4	3	2	2
4	2	4	1	4
4	1	4	1	4

A → B ✗

B → C ✓

C → D ✓

AB → A ✓

A	B
1	5
1	4
3	4
4	2
4	1

A
1
1
3
4
4

1. Reflexivity - if $\beta \subseteq \alpha$, then $\alpha \rightarrow \beta$.

Reflexivity: If RHS is a subset of LHS, then the FD must be true.

SCHOOL OF
DATA SCIENCE

Armstrong's Axioms

R

A	B	C	D	E
1	5	2	5	4
1	4	3	2	3
3	4	3	2	2
4	2	4	1	4
4	1	4	1	4

A → B ✗

B → C ✓

C → D ✓

AB → A ✓

B → D ✓

B	→	C	→	D
5		2		5
4		3		2
4		3		2
2		4		1
1		4		1

1. Reflexivity - if $\beta \subseteq \alpha$, then $\alpha \rightarrow \beta$.

2. Transitivity - if $\alpha \rightarrow \beta$ and $\beta \rightarrow \gamma$, then $\alpha \rightarrow \gamma$.

1) if two tuples have the same B values, their C values must be the same.
2) if their C values are the same, their D values must be the same.

Therefore, B → D.

Armstrong's Axioms

R

A	B	C	D	E
1	5	2	5	4
1	4	3	2	3
3	4	3	2	2
4	2	4	1	4
4	1	4	1	4

A → B ✗

B → C ✓

C → D ✓

AB → A ✓

B → D ✓

AB → AD ✓

A	B
1	5
1	4
3	4
4	2
4	1

A	D
1	5
1	2
3	2
4	1
4	1

1. Reflexivity - if $\beta \subseteq \alpha$, then $\alpha \rightarrow \beta$.
2. Transitivity - if $\alpha \rightarrow \beta$ and $\beta \rightarrow \gamma$, then $\alpha \rightarrow \gamma$.
3. **Augmentation - if $\alpha \rightarrow \beta$, then $\gamma\alpha \rightarrow \gamma\beta$.**

Observation

Since **A** appears on both sides of the FD, whether the tuple values are the same will not be determined by **A.**

Question

Armstrong's axioms

1. **Reflexivity** - if $\beta \subseteq \alpha$, then $\alpha \rightarrow \beta$.
2. **Transitivity** - if $\alpha \rightarrow \beta$ and $\beta \rightarrow \gamma$, then $\alpha \rightarrow \gamma$.
3. **Augmentation** - if $\alpha \rightarrow \beta$, then $\gamma\alpha \rightarrow \gamma\beta$.

- Given a set of functional dependencies
  - F={ **A→B**, **B→C**, **DE→A** }.
- Prove, **DE →ABC** is true.

- Since **A→B**, **A→AB** (by **Augmentation**)
- Since **B→C**, **AB→ABC** (by **Augmentation**)
- Since **A→AB** and **AB→ABC**, **A→ABC** (by **Transitivity**)
- Since **DE→A** and **A→ABC**, **DE→ABC** (by **Transitivity**)

Armstrong's Axioms

- **3 basic axioms.**
  - 1. **Reflexivity** - if $\beta \subseteq \alpha$, then $\alpha \to \beta$.
  - 2. **Transitivity** - if $\alpha \to \beta$ and $\beta \to \gamma$, then $\alpha \to \gamma$.
  - 3. **Augmentation** - if $\alpha \to \beta$, then $\gamma \alpha \to \gamma \beta$.
- **3 more axioms to help easier prove!**
  - 4. **Union** - if $\alpha \to \beta$ and $\alpha \to \gamma$, then $\alpha \to \beta\gamma$.
  - 5. **Decomposition** - if $\alpha \to \beta\gamma$, then $\alpha \to \beta$ and $\alpha \to \gamma$.
  - 6. **Pseudo-transtivity** - if $\alpha \to \beta$ and $\gamma\beta \to \delta$, then $\gamma \alpha \to \delta$.

SCHOOL OF DATA SCIENCE

Armstrong's Axioms

R

A	B	C	D	E
1	5	2	5	4
1	4	3	2	3
3	4	3	2	2
4	2	4	1	4
4	1	4	1	4

A → B	✗	AB → AD	✓
B → C	✓	AC → CE	✓
C → D	✓	A → E	✗
AB → A	✓	B → CD	✓
B → D	✓		

4. Union - if $\alpha \rightarrow \beta$ and $\alpha \rightarrow \gamma$, then $\alpha \rightarrow \beta\gamma$.

Think in this way…

If B → C, then B → BC is also true **(by augmentation)**

If B → D, then BC → CD is also true **(by augmentation)**

Therefore, with **B → BC** and **BC → CD ,**

B → CD is also true **(by transitivity).**

Armstrong's Axioms

R

A	B	C	D	E
1	5	2	5	4
1	4	3	2	3
3	4	3	2	2
4	2	4	1	4
4	1	4	1	4

A → B
B → C
C → D
AB → A
B → D

AB → AD
AC → CE ✔
A → E
B → CD

AC → C ✔
AC → E ✔

4. Union - if $\alpha \rightarrow \beta$ and $\alpha \rightarrow \gamma$, then $\alpha \rightarrow \beta\gamma$.

5. Decomposition - if $\alpha \rightarrow \beta\gamma$, then $\alpha \rightarrow \beta$ and $\alpha \rightarrow \gamma$.

Think in this way…

CE → C and CE → E are always true (by reflexivity)
Therefore, given AC → CE,
AC → C and AC → E are also true (by transitivity).

Armstrong's Axioms

R

A	B	C	D	E
1	5	2	5	4
1	4	3	2	3
3	4	3	2	2
4	2	4	1	4
4	1	4	1	4

A → B ✗
B → C ✓
C → D ✓
AB → A ✓
B → D ✓

AB → AD ✓
AC → CE ✓
A → E ✗
B → CD ✓

AC → C ✓
AC → E ✓
AB → CE ✓

4. Union - if $\alpha \rightarrow \beta$ and $\alpha \rightarrow \gamma$, then $\alpha \rightarrow \beta\gamma$.

5. Decomposition - if $\alpha \rightarrow \beta\gamma$, then $\alpha \rightarrow \beta$ and $\alpha \rightarrow \gamma$.

6. Pseudo-transitivity - if $\alpha \rightarrow \beta$ and $\gamma\beta \rightarrow \delta$, then $\alpha\gamma \rightarrow \delta$.

Think in this way…

If **B → C**, then **AB → AC** is true (**by augmentation**)
Therefore, given **AC → CE**,
AB → CE is also true (**by transitivity**).

Question

- Derive the following rule with Armstrong's axioms and the additional rules.

Prove, if **L**→**IJ** and **J**→**KH** then **L**→**KH**.

Since **L**→**IJ**, **L**→**I** and **L**→**J** (by **Decomposition**)

Since **L**→**J** and **J**→**KH** , **L**→**KH** (by **Transitivity**)

Armstrong's axioms

1. **Reflexivity** - if $\beta \subseteq \alpha$, then $\alpha \rightarrow \beta$.
2. **Transitivity** - if $\alpha \rightarrow \beta$ and $\beta \rightarrow \gamma$, then $\alpha \rightarrow \gamma$.
3. **Augmentation** - if $\alpha \rightarrow \beta$, then $\gamma\alpha \rightarrow \gamma\beta$.
4. **Union** - if $\alpha \rightarrow \beta$ and $\alpha \rightarrow \gamma$, then $\alpha \rightarrow \beta\gamma$.
5. **Decomposition** - if $\alpha \rightarrow \beta\gamma$, then $\alpha \rightarrow \beta$ and $\alpha \rightarrow \gamma$.
6. **Pseudo-transitivity** - if $\alpha \rightarrow \beta$ and $\gamma\beta \rightarrow \delta$, then $\alpha\gamma \rightarrow \delta$.

SCHOOL OF DATA SCIENCE

What is FD?

- Functional dependency is **useful in database design**.
  - We can use FD to test if a **database instance** is **legal**.
  - We can specify constraints on the **legality of relation**.
  - It can help us to design a better database (less redundancy).

Conclusion

- E-R Diagrams
  - How to design a database
- Functional dependencies
  - Reduce redundancy
  - Data quality

Next Lecture

- Normalization