COMP 3311
DATABASE MANAGEMENT SYSTEMS

TUTORIAL 2
RELATIONAL MODEL AND RELATIONAL DATABASE DESIGN

REVIEW: RELATIONAL MODEL

- A set of relation schemas define a relational database.

  Employee(empId, name, address, hkid, projectNo)

  Project(projectNo, name, budget)

- A table can be used to show the instances of a relation schema.

Employee

empId	name	address	hkid	projectNo
1	Holmes D.	86 Queen	A450361	3
5	Chan B.	21 Minto	C461378	2
35	Hui J.	16 Peak	F562916	1
8	Bell G.	53 Water	A417394	2
15	Wing R.	58 Aster	C538294	3

Project

projectNo	name	budget
1	E-commerce	200,000
2	Stock control	100,000
3	Web store	500,000

Relational Model		Representation	Notation
Relation	⇔	table	$R(A_1, A_2, \ldots, A_n)$
Attribute	⇔	column	A_i
Domain	⇔	type and range of attribute values	$dom(A_i)$
Tuple / Record	⇔	row	
Attribute value	⇔	value in table cell	

REVIEW: Keys

Superkey

A superkey, $\mathcal{S}$, of relation R={A_1, A_2, …, A_n} is a set of attributes $\mathcal{S} \subseteq$ R such that for any two tuples t_1 and $t_2 \in$ r(R), $t_1[\mathcal{S}] \neq t_2[\mathcal{S}]$

- A superkey is *any* set of attributes that can identify a unique tuple in r(R).

```
Employee(emp#, name, address, hkid, works_on_project)
```
where `emp#` and `hkid` are unique.

Possible superkeys: `emp#`
`hkid`
`{emp#, name}`
`{hkid, name, address}`
plus many others

Candidate key

A superkey that is minimal (A relation may have several candidate keys)

Primary key

One of the candidate keys (Chosen by the database designer)

REVIEW: E-R TO RELATION SCHEMA REDUCTION

We need to reduce:

generalizations / specializations	⇒	inheritance, coverage
attributes	⇒	composite, multivalued
entities	⇒	strong, weak
relationships	⇒	degree (unary, binary, ternary, etc.)
	⇒	constraints (cardinality, participation)

Cardinality/participation constraints in the E-R model are reduced to referential integrity constraints in the relational model.

REVIEW: REFERENTIAL INTEGRITY ACTIONS

If relation **T** contains the primary key k_S of relation **S** as a foreign key fk_S, which can be specified as the foreign key constraint

foreign key (fk_S) references **S**(k_S)

then the value of fk_S in a tuple of **T** must either be equal to the value of the primary key k_S of a tuple in **S** or be entirely null.

To enforce this constraint, the following actions are required.

For E-R model: total participation

on delete cascade - Delete all tuples with foreign key values in **T** that match the primary key value of the deleted tuple in **S**.

For E-R model: partial participation

on delete set null - Set to null the foreign key value of all tuples in **T** whose foreign key value matches the primary key value of the deleted tuple in **S**.

REVIEW: (7+1)-Steps for Mapping

Input: an ER model

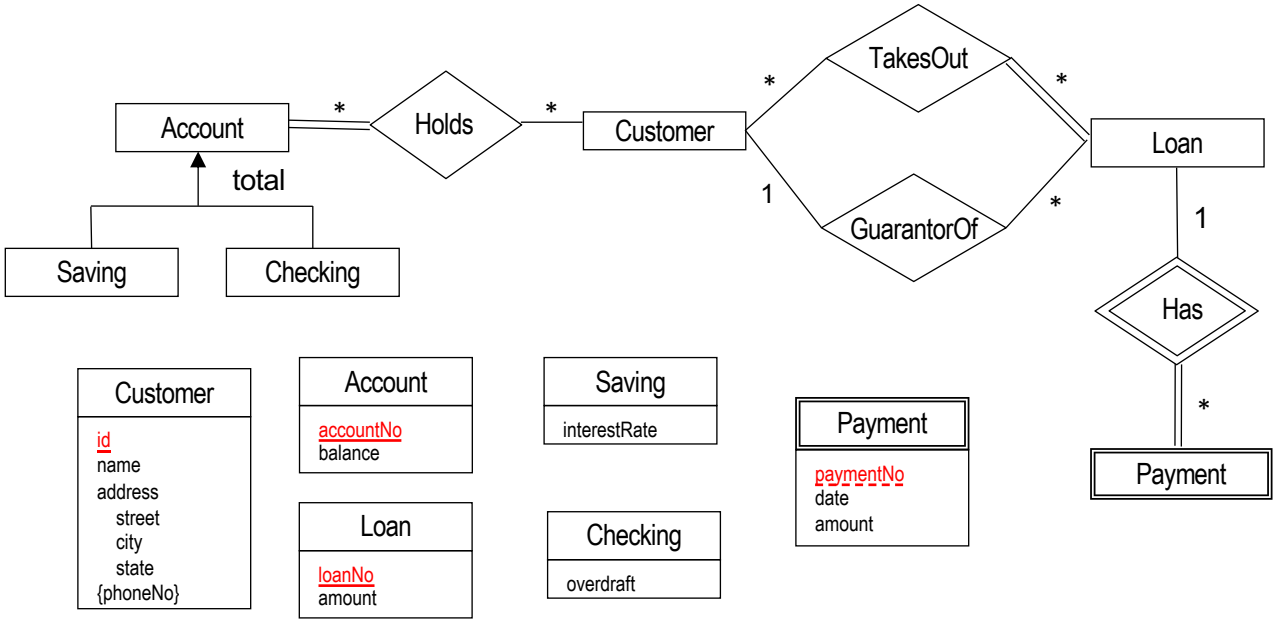
Output: relations with primary/foreign key constraints

Steps:

1. Entity mapping (strong entity with simple/composite attributes)
2. Weak entity mapping
3. Multi-valued attribute mapping
4. Binary relationship mapping (1:1)
5. Binary relationship mapping (1:*)
6. Binary relationship mapping (*:*)
7. n-ary relationship mapping
8. Super & sub-class mapping

EXERCISE 1: BANK APPLICATION

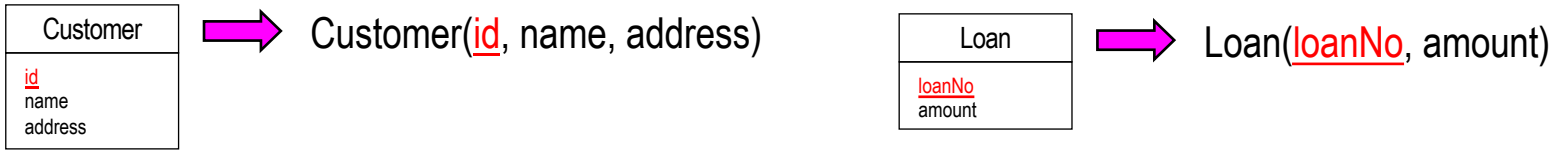
Reduce the bank E-R schema to relation schemas. Specify all referential integrity constraints. Where possible use schema combination to reduce relationships.



STEP 1: REDUCE REGULAR ENTITIES

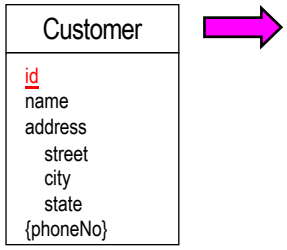
- Regular: non-weak entity with simple attributes

- For each entity type E, create a relation with all attributes
  - Include only simple component attributes of a composite attribute
  - Don't include derived attributes
  - Choose one candidate key of E as the primary key



STEP 2: REDUCE COMPOSITE/MULTIVALUED ATTRIBUTES

Composite attributes: address



Option 1: single attribute

Customer(id, name, address)

Option 2: separate attributes

Customer(id, name, street, city, state)

Which option to select?

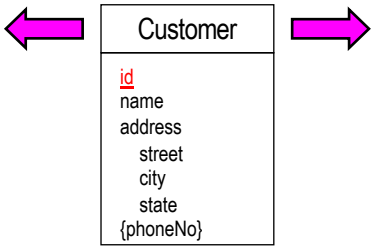
Which option to select will depend on the requirements of the application. Here we subsequently use option 2.

STEP 2: REDUCE COMPOSITE/MULTIVALUED ATTRIBUTES

- For each multi-valued attribute `A`, create a new relation `R` that includes an attribute corresponding to `A` plus the primary key `K` (as a foreign key of `R`) of the relation that represents the entity type or relationship type that has `A` as an attribute
  - The primary key of `R` is the combination of attributes `A` & `K`

Multivalued attributes: phoneNo

Customer(id, name, address)
(*previously reduced*)



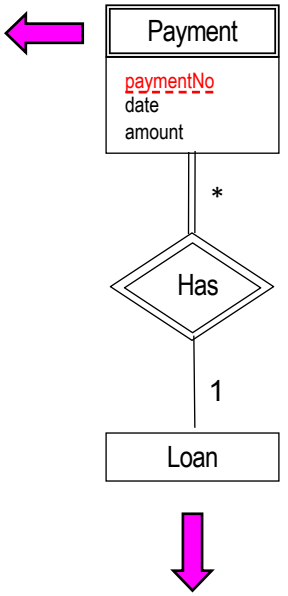
CustomerPhone(id, phoneNo)
foreign key (id) references Customer(id)
on delete cascade

STEP 3: REDUCE WEAK ENTITIES

- For each weak entity type `W` with owner entity type `E`
  - Create a relation that includes all simple attributes of `W`
  - Include as foreign key attributes in `W` to the primary key attributes of `E`
  - The primary key of `W` is the combination of the primary key of `E` and the partial key of `W`

Payment entity

Payment(loanNo, paymentNo, date, amount)
foreign key (loanNo) references Loan(loanNo)
on delete cascade



How do we reduce this entity?

⟹ Create a relation from Payment and include loanNo, the key of Loan, as a foreign key.

Loan(loanNo, amount)
(*previously reduced*)

STEP 4: REDUCE 1:N RELATIONSHIPS

- For each (non-weak) binary 1:* relationship type R, identify relation T that represents the participating entity type at the n-side of the relationship type
  - Include as foreign key of T the primary key of relation S that represents the other entity type participating in R
  - Include any simple attributes (or the simple components of composite attributes) of R as attributes of T

GuarantorOf between **Customer** and **Loan** (using schema combination)

Customer(id, name, address) *(previously reduced)*

Loan(loanNo, amount, id)
foreign key (id) references Customer(id)
on delete set null

Which relation do we use?

⟹ Loan (Add id, the key of the Customer relation, as a foreign key.)

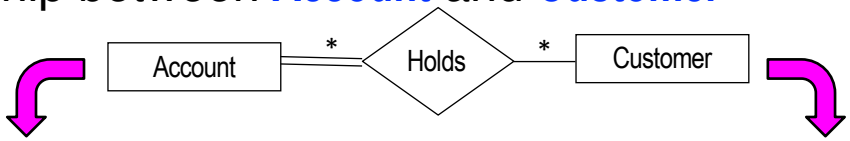
The referential integrity action is determined by the participation constraint of the entity into which the foreign key is placed.

- partial: on delete set null
- total: on delete cascade

STEP 5: REDUCE N:M RELATIONSHIPS

- For each binary *:* relationship type R, create a new relation R
  - Include the primary keys of the relations that represent the participating entity types (i.e., S and T) as foreign key of R
  - The combination of foreign keys will form the primary key of R
    - Note: cannot represent the *:* using a single foreign key in one relation because of the *:* cardinality ratio
  - Include any simple attributes of R as attributes of R

Holds relationship between **Account** and **Customer**



Account(accountNo, balance)

Customer(id, name, address) (*previously reduced*)

Holds(accountNo, id)

foreign key (accountNo) references Account(accountNo)
on delete cascade
foreign key (id) references Customer(id)
on delete cascade

How do we reduce this relationship?

⇒ Create a relation, Holds, with the key, accountNo, of the Account relation and the key, id, of the Customer relation.

For a relation that represents a relationship, the referential integrity action is always on delete cascade.

STEP 5: REDUCE N:M RELATIONSHIPS

- For each binary *:* relationship type R, create a new relation R
  - Include the primary keys of the relations that represent the participating entity types (i.e., S and T) as foreign key of R
  - The combination of foreign keys will form the primary key of R
    - Note: cannot represent the *:* using a single foreign key in one relation because of the *:* cardinality ratio
  - Include any simple attributes of R as attributes of R

TakesOut relationship between Customer and Loan

Customer(id, name, address) (*previously reduced*)

Loan(loanNo, amount) (*previously reduced*)

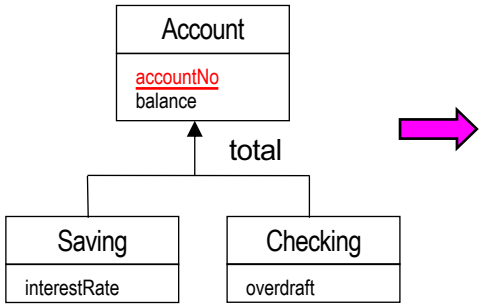
TakesOut(id, loanNo)
  foreign key (id) references Customer(id)
    on delete cascade
  foreign key (loanNo) references Loan(loanNo)
    on delete cascade

How do we reduce this relationship?

⇒ Create a relation, TakesOut, with the key, id, of the Customer relation and the key, loanNo, of the Loan relation.

STEP 6: SUPER & SUB-CLASSES

- A general case
  - We create a relational table for the superclass and create a relational table for each subclass
  - The primary key of each of the subclass is the primary key of the superclass, which also become foreign keys



Option 1: Reduce *all entities* to relation schemas.

Account(accountNo, balance)

Saving(accountNo, interestRate)
foreign key (accountNo) references Account(accountNo)
on delete cascade

Checking(accountNo, overdraft)
foreign key (accountNo) references Account(accountNo)
on delete cascade

Which option to select?

Select Option 1 since Account has a relationship to other entities and all the subclass entities have their own attributes.

Option 2: Reduce *only subclass entities* to relation schemas.

Saving(accountNo, balance, interestRate)

Checking(accountNo, balance, overdraft)

ANSWER: BANK APPLICATION REDUCTION

Account(accountNo, balance)

Saving(accountNo, interestRate)
foreign key (accountNo) references Account(accountNo)
on delete cascade

Checking(accountNo, overdraft)
foreign key (accountNo) references Account(accountNo)
on delete cascade

Customer(id, name, street, city, state)

CustomerPhone(id, phoneNo)
foreign key (id) references Customer(id)
on delete cascade

Payment(loanNo, paymentNo, date, amount)
foreign key (loanNo) references Loan(loanNo)
on delete cascade

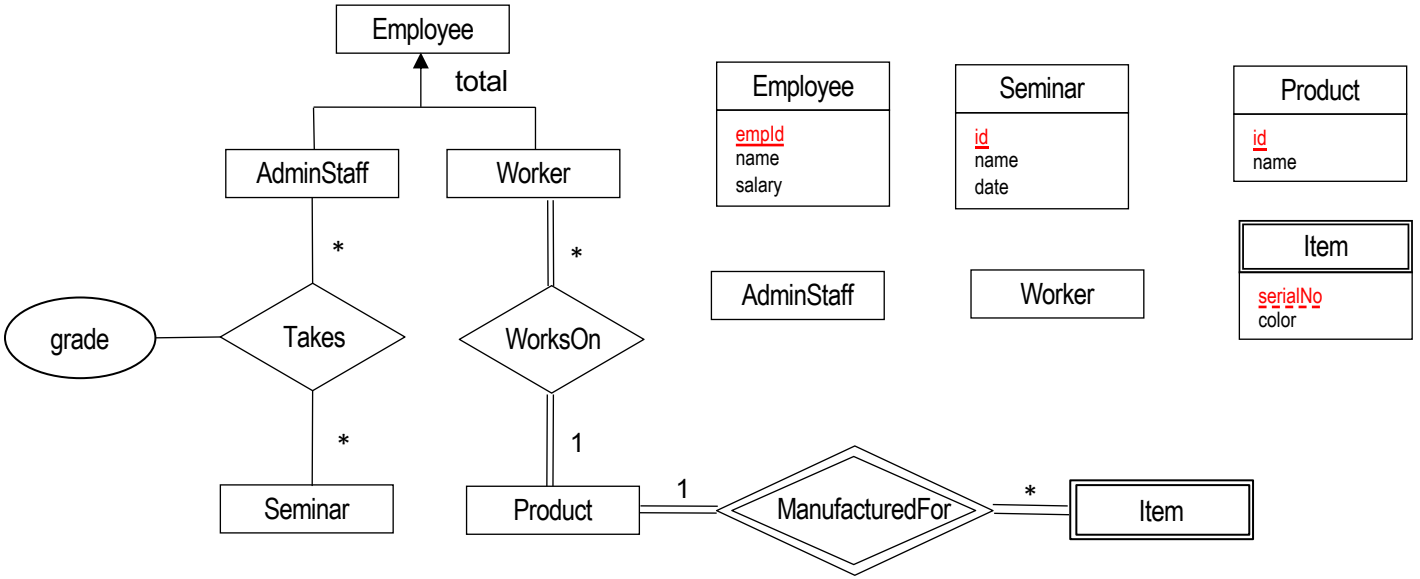
Loan(loanNo, amount, id)
foreign key (id) references Customer(id)
on delete set null

Holds(accountNo, id)
foreign key (accountNo) references
Account(accountNo)
on delete cascade
foreign key (id) references Customer(id)
on delete cascade

TakesOut(id, loanNo)
foreign key (id) references Customer(id)
on delete cascade
foreign key (loanNo) references Loan(loanNo)
on delete cascade

EXERCISE 2: FACTORY APPLICATION

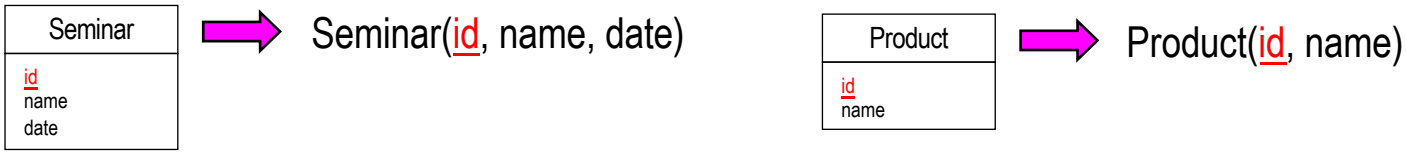
Reduce the factory E-R schema to relation schemas. Specify all referential integrity constraints. Where possible, use schema combination to reduce relationships.



STEP 1: REDUCE REGULAR ENTITIES

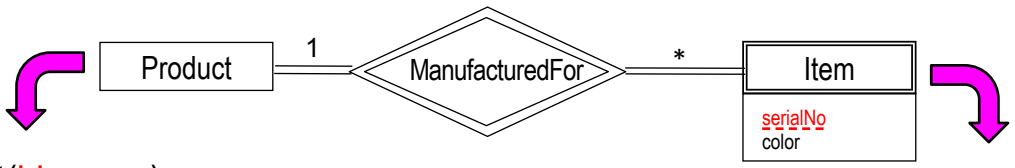
- Regular: non-weak entity with simple attributes

- For each entity type E, create a relation with all attributes
  - Include only simple component attributes of a composite attribute
  - Don't include derived attributes
  - Choose one candidate key of E as the primary key



STEP 2: REDUCE WEAK ENTITIES

- For each weak entity type W with owner entity type E
  - Create a relation that includes all simple attributes of W
  - Include as foreign key attributes in W to the primary key attributes of E
  - The primary key of W is the combination of the primary key of E and the partial key of W



Product(id, name) (*previously reduced*)

Item(id, serialNo, color)
foreign key (id) references Product(id)
on delete cascade

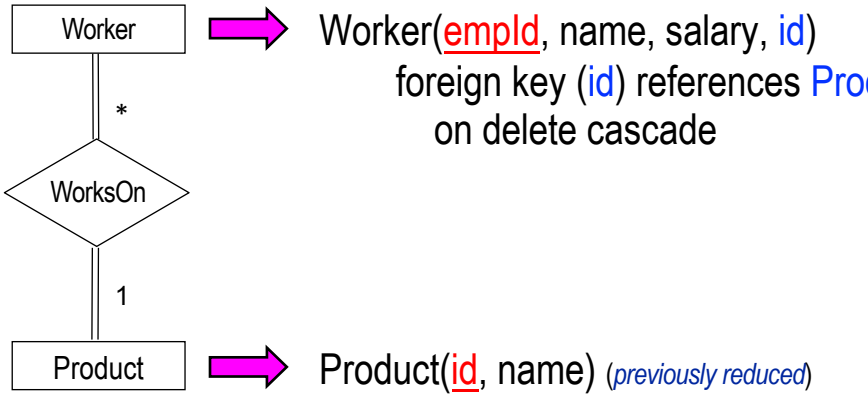
How do we reduce this entity?

⟹ Create a relation from Item that includes the key, id, of the Product relation.

STEP 3: REDUCE 1:N RELATIONSHIPS

- For each (non-weak) binary 1:* relationship type `R`, identify relation `T` that represents the participating entity type at the n-side of the relationship type
  - Include as foreign key of `T` the primary key of relation `S` that represents the other entity type participating in `R`
  - Include any simple attributes (or the simple components of composite attributes) of `R` as attributes of `T`

WorksOn relationship between **Worker** and **Product**



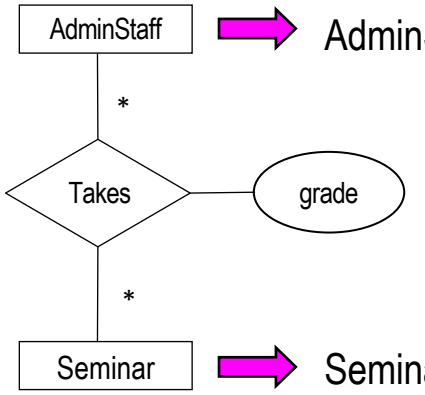
Which relation do we use?

⟹ Worker (Add the key, id, of the Product relation as a foreign key.)

STEP 4: REDUCE N:M RELATIONSHIPS

- For each binary *:* relationship type `R`, create a new relation `R`
  - Include the primary keys of the relations that represent the participating entity types (i.e., `S` and `T`) as foreign key of `R`
  - The combination of foreign keys will form the primary key of `R`
    - Note: cannot represent the *:* using a single foreign key in one relation because of the *:* cardinality ratio
  - Include any simple attributes of `R` as attributes of `R`

Takes relationship between **AdminStaff** and **Seminar**



AdminStaff(empId)

Takes(empId, id, grade)
foreign key (empId) references AdminStaff(empId)
on delete cascade
foreign key (id) references Seminar(id)
on delete cascade

Seminar(id, name, date) (*previously reduced*)

How do we reduce this relationship?

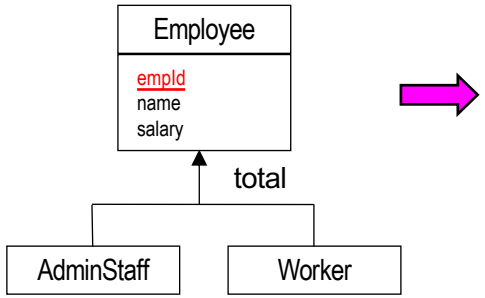
⇒ Create a relation Takes with the key of AdminStaff and Seminar.

Anything else?

⇒ Add the attribute grade.

STEP 5: SUPER & SUB-CLASSES

- A special case for disjoint-total only
  - We create a relational table for each subclass. The attributes of the superclass are merged into each of the subclasses.
  - The primary key of the subclass table is the primary key of the superclass.



Option 1: Reduce all entities to relation schemas.

Employee(empId, name, salary)

AdminStaff(empId)
foreign key (empId) references Employee(empId)
on delete cascade

Worker(empId)
foreign key (empId) references Employee(empId)
on delete cascade

Which option to select?

Option 2: Reduce only subclass entities to relation schemas.

AdminStaff(empId, name, salary)

Worker(empId, name, salary)

Select Option 2 since Employee has no relationships to other entities, the subclasses have no attributes, and the generalization is disjoint and total.

ANSWER: FACTORY APPLICATION REDUCTION

AdminStaff(empId, name, salary)

Worker(empId, name, salary, id)
    foreign key (id) references Product(id)
        on delete cascade

Seminar(id, name, date)

Product(id, name)

Item(id, serialNo, color)
    foreign key (id) references Product(id)
        on delete cascade

Takes(empId, id, grade)
    foreign key (empId) references AdminStaff(empId)
        on delete cascade
    foreign key (id) references Seminar(id)
        on delete cascade