Logical Modeling

Key Points

A *data model* is a notation (concepts) for describing data or information.

- describes structure of the data
- describes constraints on the data
- provides operations on the data (queries and modifications)

- relational model
  - supported by relational databases
- ER-to-relational mapping
  - model data requirements using a conceptual model because it is higher-level and closer to normal language
  - translate ER model to relational model for database implementation

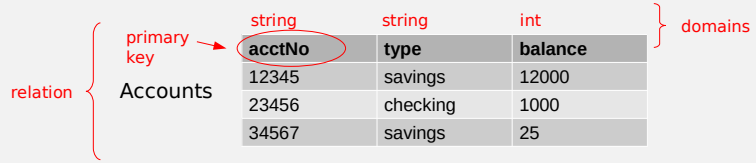
ER Model

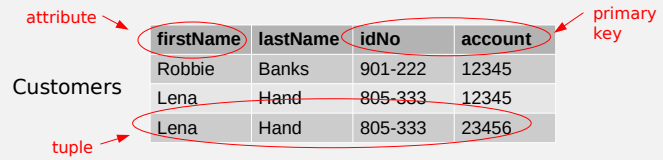
- structure
  - *entities*
  - *relationships* between entities
  - *attributes* of entities and relationships
- constraints
  - attributes
    - *key* – set of attributes which together uniquely identify an entity
  - relationships
    - *cardinality ratio* defines maximum number of relationship instances a given entity can participate in
    - *participation* defines the minimum number of relationship instances that a given entity must participate in

Relational Model

- structure
  - *relations* (tables)
  - *attributes* (columns)
  - *tuples* (rows)
- constraints
  - *domain* – attribute data type
  - *NOT NULL* – attribute is required (value cannot be NULL)
  - *primary key* – a set of attributes which together distinguish tuples
  - *entity integrity constraint*
    - primary key attributes cannot be NULL
  - *referential integrity constraint*
    - requires that values for attribute A for tuples in relation R appear in attribute B for some tuple in relation R

The Relational Model: Structure





relation schema

Accounts(acctNo,type,balance)
Customers(firstName,lastName,idNo,account)

database schema (with primary key)

The Relational Model: Terminology

Choose matching items on the right so that "[left thing] is an example of [right thing]" is true. Use the following relation schema.

```
Accounts(acctNo, type, balance)
Customers(firstName, lastName, idNo, account)
```

Accounts.acctNo | Customers.account->Accounts.acctNo | Customers.account in Customers.account->Accounts.acctNo | Customers.firstName, Customers.lastName, Customers.idNo | Accounts.acctNo, Accounts.type | Accounts.balance is a number | Customer.account cannot be NULL

superkey		0 %
referential integrity constraint		0 %
domain constraint		0 %
schematic constraint		0 %
foreign key	1 respondent	17 %
entity integrity constraint		0 %
primary key	5 respondents	83 %
none of these		0 %

The Relational Model: Terminology

```
Accounts(acctNo, type, balance)
Customers(firstName, lastName, idNo, account)
```

Accounts.acctNo | **Customers.account->Accounts.acctNo** | Customers.account in Customers.account->Accounts.acctNo | Customers.firstName, Customers.lastName, Customers.idNo | Accounts.acctNo, Accounts.type | Accounts.balance is a number | Customer.account cannot be NULL

superkey		0 %
referential integrity constraint	3 respondents	50 %
domain constraint		0 %
schematic constraint		0 %
foreign key	3 respondents	50 %
entity integrity constraint		0 %
primary key		0 %
none of these		0 %

Accounts.acctNo | Customers.account->Accounts.acctNo | **Customers.account in Customers.account->Accounts.acctNo** | Customers.firstName, Customers.lastName, Customers.idNo | Accounts.acctNo, Accounts.type | Accounts.balance is a number | Customer.account cannot be NULL

superkey	1 respondent	17 %
referential integrity constraint	4 respondents	67 %
domain constraint		0 %
schematic constraint		0 %
foreign key	1 respondent	17 %
entity integrity constraint		0 %
primary key		0 %
none of these		0 %

The Relational Model: Terminology

```
Accounts(acctNo, type, balance)
Customers(firstName, lastName, idNo, account)
```

Accounts.acctNo | Customers.account->Accounts.acctNo | Customers.account in Customers.account->Accounts.acctNo | **Customers.firstName, Customers.lastName, Customers.idNo** | Accounts.acctNo, Accounts.type | Accounts.balance is a number | Customer.account cannot be NULL

superkey	1 respondent	17 %
referential integrity constraint	1 respondent	17 %
domain constraint	2 respondents	33 %
schematic constraint		0 %
foreign key		0 %
entity integrity constraint		0 %
primary key		0 %
none of these	2 respondents	33 %

Accounts.acctNo | Customers.account->Accounts.acctNo | Customers.account in Customers.account->Accounts.acctNo | Customers.firstName, Customers.lastName, Customers.idNo | **Accounts.acctNo, Accounts.type** | Accounts.balance is a number | Customer.account cannot be NULL

superkey	2 respondents	33 %
referential integrity constraint		0 %
domain constraint	2 respondents	33 %
schematic constraint	1 respondent	17 %
foreign key		0 %
entity integrity constraint		0 %
primary key		0 %
none of these	1 respondent	17 %

The Relational Model: Terminology

```
Accounts(acctNo, type, balance)
Customers(firstName, lastName, idNo, account)
```

Accounts.acctNo | Customers.account->Accounts.acctNo | Customers.account in Customers.account->Accounts.acctNo | Customers.firstName, Customers.lastName, Customers.idNo | Accounts.acctNo, Accounts.type | **Accounts.balance is a number** | Customer.account cannot be NULL

superkey		0 %
referential integrity constraint		0 %
domain constraint	3 respondents	50 %
schematic constraint		0 %
foreign key		0 %
entity integrity constraint	3 respondents	50 %
primary key		0 %
none of these		0 %

Accounts.acctNo | Customers.account->Accounts.acctNo | Customers.account in Customers.account->Accounts.acctNo | Customers.firstName, Customers.lastName, Customers.idNo | Accounts.acctNo, Accounts.type | Accounts.balance is a number | **Customer.account cannot be NULL**

superkey		0 %
referential integrity constraint		0 %
domain constraint		0 %
schematic constraint	3 respondents	50 %
foreign key		0 %
entity integrity constraint	3 respondents	50 %
primary key		0 %
none of these		0 %

The Relational Model: Structure

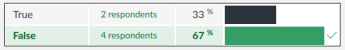
True or false?

The following are different relations.

acctNo	type	balance
12345	savings	12000
23456	checking	1000
34567	savings	25

acctNo	type	balance
23456	checking	1000
12345	savings	12000
34567	savings	25

- false – tuples in a relation are not ordered.
  (Relations are sets of tuples.)

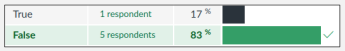


The following are different relations.

acctNo	type	balance
12345	savings	12000
23456	checking	1000
34567	savings	25

acctNo	balance	type
12345	12000	savings
23456	1000	checking
34567	25	savings

- false – attributes in a relation are not ordered.
  (Though for practical purposes we consider them in the order listed in the schema.)



The Relational Model: Structure

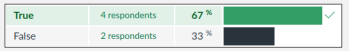
True or false?

The following are different relations.

acctNo	type	balance
12345	savings	12000
23456	checking	1000
34567	savings	25

acctNo	type	balance
12345	savings	12000
23456	checking	1000
34567	savings	25
45678	checking	2050

- true – relations are sets of tuples, and these have different tuples.
  (But the relation schemas are the same.)



The Relational Model: Constraints

True or false?

Given the relation schema

```
Accounts(acctNo,type,balance)
```

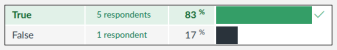
the following is a legal relation.

acctNo	type	balance
12345	savings	12000
23456	checking	1000
34567	savings	12000

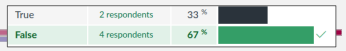
the following is a legal relation.

acctNo	type	balance
12345	savings	12000
12345	checking	1000
34567	savings	25

- true – it is possible for two tuples in the same relation to have the same value for an attribute as long as that attribute (or group of attributes) doesn't constitute a key.



- false – duplicate values are not allowed for a key attribute.



The Relational Model: Constraints

True or false?

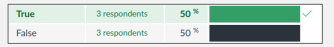
Given the relation schema

```
Accounts(acctNo,type,balance)
```

the following is a legal relation.

acctNo	type	balance
12345	savings	12000
12345	checking	1000
34567	savings	25

- true – it is possible for two tuples in the same relation to have the same value for an attribute as long as that attribute (or group of attributes) doesn't constitute a key.



The Relational Model: Constraints

Given the relation schema

`Accounts(acctNo,type,balance)`

the following is a legal relation.

acctNo	type	balance
12345	savings	12000
23456	NULL	1000
34567	savings	25

- false – entity integrity constraint means that key attributes cannot have NULL values

True	2 respondents	33 %
False	4 respondents	67 % ✓

Given the relation schema

`Accounts(acctNo,type,balance)`

the following is a legal relation.

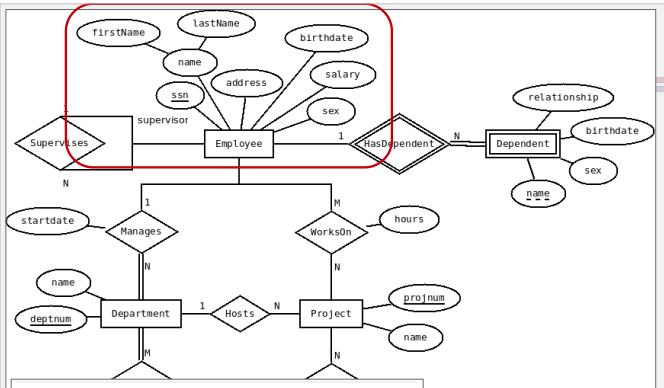
acctNo	type	balance
12345	savings	12000
23456	checking	1000
34567	savings	NULL

- true – for non-key attributes, any NOT NULL constraint must be specified separately

True ✓	3 respondents	50 %
False	3 respondents	50 %

ER → Relational: Entity Types

1. entity type E → relation with all of the simple attributes of E
   - key of E → primary key



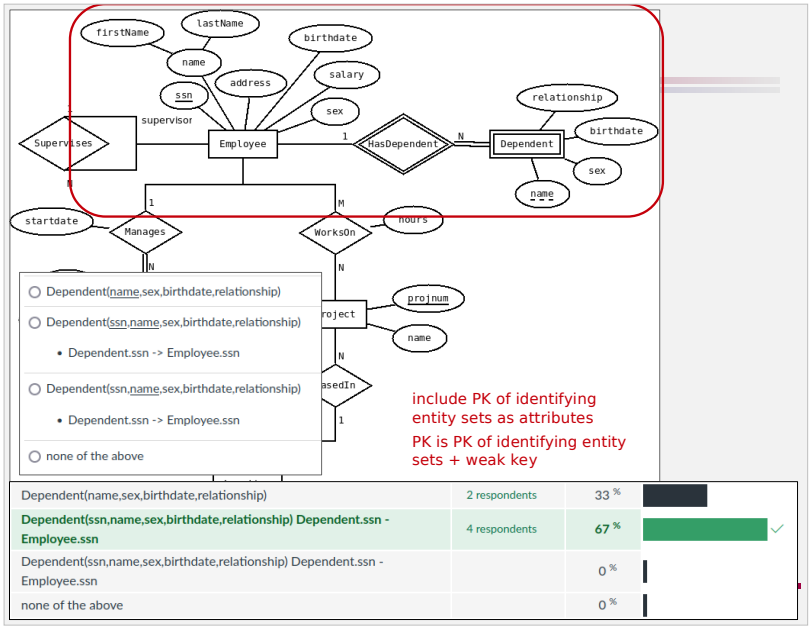
- Employee(ssn,name,firstName,lastName,address,birthdate,salary,sex)
- Employee(ssn,name,address,birthdate,salary,sex)
- Employee(ssn,firstName,lastName,address,birthdate,salary,sex)
- none of the above

`firstName, lastName` are the simple attributes that make up the composite `name`

Employee(ssn,name,firstName,lastName,address,birthdate,salary,sex)		0 %
Employee(ssn,name,address,birthdate,salary,sex)	3 respondents	50 %
Employee(ssn,firstName,lastName,address,birthdate,salary,sex) ✓	3 respondents	**50 %**
none of the above		0 %

ER → Relational: Weak Entity Types

2. weak entity type W → relation with all of the simple attributes of W + all of the simple attributes of identifying relationships R + all key attributes of identifying entity sets E
   - partial key of W + key attributes of E → primary key
   - attributes stemming from key attributes of E are foreign keys referring to E's relation



ER → Relational: Binary Relationship Types

3. binary 1:1 relationship type
   - if both entity types have total participation → use merged relation
   - if one entity type has total participation → use foreign key
   - if neither entity type has total participation → use foreign key or cross-reference
4. binary 1:N relationship type → use foreign key (with the relationship on the N side) or cross-reference
5. binary M:N relationship type → use cross-reference

merged relation approach – combine the relations for the two participating entity sets into one, along with the simple attributes of the relationship

foreign key approach – add the simple attributes of the relationship to the relation for one of the participating entity sets, along with the primary key of the relation for the other participating entity set

cross-reference approach – create a new relation with the simple attributes of the relationship and the primary keys of the relations for the participating entity sets

ER → Relational: Binary Relationship Types

Use the choice of primary key to enforce cardinality constraints.

S(s1,s2)
T(t1,t2)

- merged relation approach

  supports N:1 with repetition of T
  supports 1:N with repetition of S
  supports M:N with repetition of both – or
  1:1 (no repetition)

  ST(s1,s2,t1,t2)
  ST(s1,s2,t1,t2)
  ST(s1,s2,t1,t2)

- foreign key approach

  supports N:1
  supports 1:N with repetition of S
  supports M:N with repetition of S

  S(s1,s2,t1)
  S(s1,s2,t1)
  S(s1,s2,t1)
  } T(t1,t2)

- cross-reference approach

  supports N:1
  supports 1:N
  supports M:N

  ST(s1,t1)
  ST(s1,t1)
  ST(s1,t1)
  } S(s1,s2) T(t1,t2)

ER → Relational: Binary Relationship Types

Participation.

S(s1,s2)
T(t1,t2)

- merged relation approach

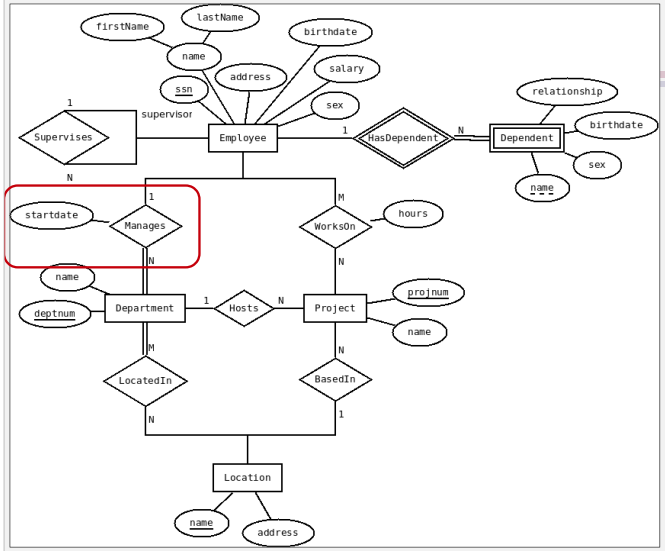
requires total participation of both **ST(s1,s2,t1,t2)**

- foreign key approach

partial participation of S results in NULL values **S(s1,s2,t1)** T(t1,t2)
cannot require total participation of T

- cross-reference approach

no NULLs – ST only contains entries if there is an instance of the relationship type **ST(s1,t1)** S(s1,s2) T(t1,t2)
cannot require total participation of S or T



Step 4 of the ER-to-relational mapping process means that Manages will end up as: (choose all that are appropriate - the constraints from the ER diagram are captured without drawbacks like redundant information or unnecessary NULL values)

- ☐ A single relation for Employee, Department, and Manages:
  Employee(ssn,empname,address,birthdate,salary,sex,deptnum,deptname,startdate)
- ☐ A single relation for Employee, Department, and Manages:
  Department(deptnum,deptname,ssn,empname,address,birthdate,salary,sex,startdate)
  - Department.ssn NOT NULL
- ☐ Department(deptnum,deptname,ssn,startdate)
  - Department.ssn -> Employee.ssn
- ☐ Department(deptnum,deptname,ssn,startdate)
  - Department.ssn -> Employee.ssn
  - Department.ssn NOT NULL
- ☐ Employee(ssn,firstName,lastName,address,birthdate,salary,sex,deptnum,startdate)
  - Employee.deptnum -> Department.deptnum
- ☐ Employee(ssn,firstName,lastName,address,birthdate,salary,sex,deptnum,startdate)
  - Employee.deptnum -> Department.deptnum
  - Employee.deptnum NOT NULL

merged relation is not suitable
– cannot accommodate employees who aren't managers

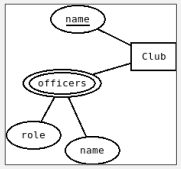
foreign key approach
– relationship goes with N side (Department)
– at most 1 manager is enforced by PK being PK of N side (Department)
– total participation of Department means no NULLs for ssn, startdate

cross-reference approach
– at most 1 manager is enforced by PK being PK of N side (Department)
– does not enforce total participation of Department

- ☐ Manages(ssn,deptnum,startdate)
  - Manages.ssn -> Employee.ssn
  - Manages.deptnum -> Department.deptnum
- ☐ Manages(ssn,deptnum,startdate)
  - Manages.ssn -> Employee.ssn
  - Manages.deptnum -> Department.deptnum
  - Manages.deptnum NOT NULL
- ☐ Manages(ssn,deptnum,startdate)
  - Manages.ssn -> Employee.ssn
  - Manages.deptnum -> Department.deptnum
  - Manages.ssn NOT NULL

ER → Relational: Multivalued Attributes

6. multivalued attribute A on entity type E → relation with attribute A and primary key of E's relation
   all attributes → primary key



- ☐ Club(clubname,role,name)
- ☐ Club(name,officers)
- ☐ Club(name)
  Officers(clubname,role,name)
  - Officers.clubname -> Club.name

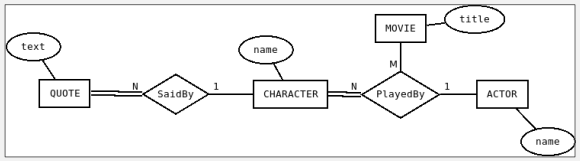
entity set → relation with all of the simple attributes of the entity set
multivalued attribute → separate relation with the attribute and the PK of the relation for the entity set

Club(clubname,role,name)		0 %
Club(name,officers)	2 respondents	33 %
Club(name)Officers(clubname,role,name) Officers.clubname - Club.name	4 respondents	**67 %** ✓

ER → Relational: n-Ary Relationship Types

7. n-ary relationship type → extension of cross-reference approach for binary relationship types
   - i.e. separate relation for the relationship, with attributes corresponding to the simple attributes of the relationship and the PK of all of the participating entity sets
     - can adjust PK to capture some cardinality constraints
   - can't capture total participation

Interpreting Cardinality Constraints



[one instance of stuff on the other side] <relationship> *n* [entity type]

(C1,M1,A1)
(C1,M2,A2)
~~(C1,M1,A2)~~