ER to Relational / SQL Exercises

Announcements

Feedback sheet: Experiment continues!

Homework 1: Available now!

Scribe notes are quite good! (add to them!)

https://github.com/w4111/scribenotes

Q: Are slides sufficient?

Yes!

Reading is recommended if you find my lectures useless and confusing, or want a review

Entities vs Relationships

Rough guideline

"<*entity*> has <*relationship*> with <*entity*>"

user has instructor relationship with courses

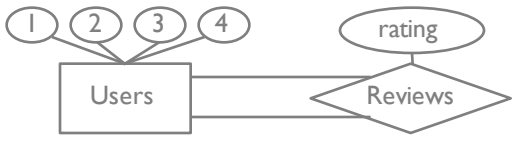
user has friendship with user

user can view profile NO. Unless actually want to log view operations

Ask yourself: is the relationship something you actually want to *store?*

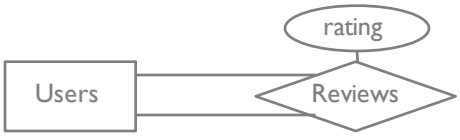
Data vs Logic

ER model stores minimum data to support application. *Does not store logic*

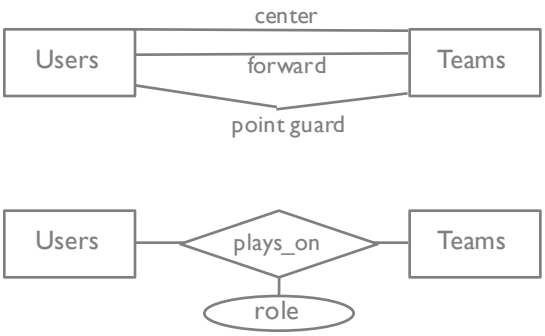


Data vs Logic

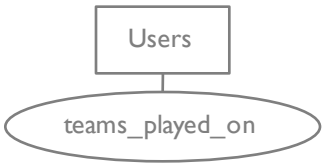
ER model stores minimum data to support application. *Does not store logic*



Multiple Relationships



Lists, Strings…



Thinking about how to store the data, not the data itself.
Violates physical data independence.

Q: Refer to non PK attributes using foreign keys?

ER diagrams: never happens

SQL: Must refer to primary key/unique (cand key)

FOREIGN KEY (a, b) REFERENCES other (x, y)

Q: SQL for 2 relationships, 1 entity

Two relationships = two tables
Relationship with itself: Two attributes

```
CREATE TABLE follows(
  source int,
  destination int,
  PRIMARY KEY (source, destination),
  FOREIGN KEY (source) REFERENCES users
  FOREIGN KEY (destination) REFERENCES users
);
```

Q: ISA hierarchy: children can't overlap

Users
Students
Instructors

3 tables: A user could be in all three
2 tables: Can't be a plain user

No way to express this constraint

Q: Can a primary key be null/blank?

No! PRIMARY KEY(a, b, c) means:

```
  a NOT NULL
  b NOT NULL
  c NOT NULL
```

… but a foreign key can!

```
CREATE TABLE tweet(
  tid int, author_uid REFERENCES users
)
```

Tweet of (0, null) is okay

Q: weak entity be involved in other relationships?

Yes! No limits to relationships with weak entity

Cannot be uniquely identified by its own attributes
Binary "exactly one" relationship with "owner"
Delete owner: delete weak references

Technical definition is rare in reality
Typically assign entities primary key ids

Relational Constraints

Domain constraints
Primary key constraints
Foreign key constraints
Unique constraints
NOT NULL constraints
CHECK constraints

Single column keys

Abbreviated syntax:

```
name   type  PRIMARY KEY
name   type  UNIQUE
name   type  REFERENCES table
```

Single Column Shortcut Notation

```
CREATE TABLE A(
   id int,
   ref int,
   PRIMARY KEY (id),
   FOREIGN KEY (ref) REFERENCES other
);
```

Single Column Shortcut Notation

```
CREATE TABLE A(
   id int PRIMARY KEY,
   ref int REFERENCES other,
   PRIMARY KEY (id),
   FOREIGN KEY (ref) REFERENCES other
);
```

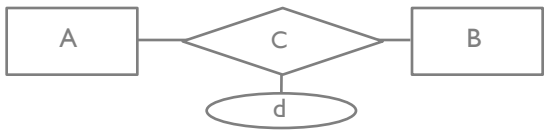
ER → Relational translation

Translate ignoring constraints

Entity: new table with primary key, no foreign key
Relationship: new table;
- Primary key = primary keys of related entities
- Don't forget foreign keys

Aggregation
- Convert inner relationship (aggregated relationship)
- Treat table for aggregated relationship as entity

ISA: tables for each type or just subtypes



A(
aid
)

C(
aid, bid
d
)

B(
bid
)

Add constraints

At most one:
- On relationship, add: UNIQUE(entity key)
- OR
- Combine tables (permit NULL)

Exactly one:
- Combine tables; NOT NULL constraints

At least one:
¯_(ツ)_/¯

Eliminate Redundancy

PRIMARY KEY (a, b) + UNIQUE (b) = PRIMARY KEY (b)

Same primary key? Can combine tables

Perfect translation

For all possible database instances:
- Constraints violated in ER are violated in relational
- Constraints violated in relational are violated in ER

- If ER doesn't violate, neither should relational
- If relational doesn't violate, neither should ER

Some diagrams cannot be perfectly translated (e.g. at least one constraints)

How to check a translation?

For each ER constraint:
1. Find example that violates ER constraint
2. Verify it violates relational version

3. Find example that passes ER constraint
4. Verify it passes relational version

~~"Proof by example": No guarantee, but effective~~

Non-exhaustive examples (with bastardized notation!)

And how to check them

[Instructors] – <Teaches> – [Courses]

Instructors(
iid int primary key,
name text
)

Teaches(
iid int references Instructors,
cid int references Courses,
primary key (iid,cid)
)

Courses(
cid int primary key,
title text
)

iid	name
1	Evan
2	Luis
3	Martha

iid	cid
1	4111
2	4111
3	3827
3	4111
3	3134

cid	title
4111	databases
3827	Systems
3134	Data structures

[Instructors] – <Teaches> ← [Courses]

Instructors(
iid int primary key,
name text
)

Teaches(
iid int references Instructors,
cid int references Courses,
primary key (iid,cid)
)

Courses(
cid int primary key,
title text
)

Q1: What data is no longer valid?
Q2: How do we correct the schema?

iid	name
1	Evan
2	Luis
3	Martha

iid	cid
1	4111
2	4111
3	3827
3	4111
3	3134

cid	title
4111	databases
3827	Systems
3134	Data structures

[Instructors] – <Teaches> ← [Courses]

Instructors(
iid int primary key,
name text
)

Teaches(
iid int references Instructors,
cid int references Courses,
primary key (iid,cid)
UNIQUE (cid)
)

Courses(
cid int primary key,
title text
)

Q1: What data is no longer valid?
Q2: How do we correct the schema?

iid	name
1	Evan
2	Luis
3	Martha

iid	cid
1	4111
2	4111
3	3827
3	4111
3	3134

cid	title
4111	databases
3827	Systems
3134	Data structures

[Instructors] – <Teaches> ← [Courses]

Instructors(
iid int primary key,
name text
)

Teaches(
iid int references Instructors,
cid int references Courses,
primary key (iid,cid)
UNIQUE (cid)
)

Courses(
cid int primary key,
title text
)

Q1: What data is no longer valid?
Q2: How do we correct the schema?

iid	name
1	Evan
2	Luis
3	Martha

iid	cid
1	4111
2	4111
3	3827
3	4111
3	3134

cid	title
4111	databases
3827	Systems
3134	Data structures

[Instructors] – <Teaches> ← [Courses]

Instructors(
iid int primary key,
name text
)

Teaches(
iid int references Instructors,
cid int references Courses,
primary key (iid,cid)
UNIQUE (cid)
)

Courses(
cid int primary key,
title text
)

Is there redundancy in the schema?

iid	name
1	Evan
2	Luis
3	Martha

iid	cid
1	4111
3	3827
3	3134

cid	title
4111	databases
3827	Systems
3134	Data structures

[Instructors] – <Teaches> ← [Courses]

Instructors(
iid int primary key,
name text
)

Teaches(
iid int references Instructors,
cid int references Courses,
primary key (cid)
)

Courses(
cid int primary key,
title text
)

Is there redundancy in the schema?

iid	name
1	Evan
2	Luis
3	Martha

iid	cid
1	4111
3	3827
3	3134

cid	title
4111	databases
3827	Systems
3134	Data structures

[Instructors] – <Teaches> ← [Courses]

```
Instructors(
  iid int primary key,
  name text
)
```

```
Teaches(
  iid int references Instructors,
  cid int references Courses,
  primary key (cid)
)
```

```
Courses(
  cid int primary key,
  title text
)
```

NULLs allowed by default…

iid	name
1	Evan
2	Luis
3	Martha

iid	cid
1	4111
3	3827
null	3134

cid	title
4111	databases
3827	Systems
3134	Data structures

[Instructors] – <Teaches> ← [Courses]

```
Instructors(
  iid int primary key,
  name text
)
```

```
Teaches(
  iid int NOT NULL
    references Instructors,
  cid int references Courses,
  primary key (cid)
)
```

```
Courses(
  cid int primary key,
  title text
)
```

NULLs allowed by default…

iid	name
1	Evan
2	Luis
3	Martha

iid	cid
1	4111
3	3827
null	3134

cid	title
4111	databases
3827	Systems
3134	Data structures

[Instructors] – <Teaches> ← [Courses]

```
Instructors(
  iid int primary key,
  name text
)
```

```
Teaches(
  iid int NOT NULL
    references Instructors,
  cid int references Courses,
  primary key (cid)
)
```

```
Courses(
  cid int primary key,
  title text
)
```

Is there STILL redundancy in the schema?

iid	name
1	Evan
2	Luis
3	Martha

iid	cid
1	4111
3	3827
3	3134

cid	title
4111	databases
3827	Systems
3134	Data structures

[Instructors] – <Teaches> ← [Courses]

```
Instructors(
  iid int primary key,
  name text
)
```

```
Teaches(
  iid int NOT NULL
    references Instructors,
  cid int references Courses,
  primary key (cid)
)
```

```
Courses(
  cid int primary key,
  title text
)
```

```
Courses_Teaches(
  cid int primary key,
  title text,
  iid int references Instructors
)
```

iid	name
1	Evan
2	Luis
3	Martha

iid	cid
1	4111
3	3827
3	3134

cid	title
4111	databases
3827	Systems
3134	Data structures

[Instructors] – <Teaches> ← [Courses]

```
Instructors(
  iid int primary key,
  name text
)
```

```
Courses_Teaches(
  cid int primary key,
  title text,
  iid int references Instructors
)
```

Why is *this* NULL allowed??

iid	name
1	Evan
2	Luis
3	Martha

cid	title	iid
4111	databases	1
3827	Systems	3
3134	Data structures	3
4118	operating systems	null

null required to represent the ER diagram

[Instructors] – <Teaches> ← [Courses]

```
Instructors(
  iid int primary key,
  name text
)
```

```
Courses_Teaches(
  cid int primary key,
  title text,
  iid int references Instructors
)
```

iid	name
1	Evan
2	Luis
3	Martha

cid	title	iid
4111	databases	1
3827	Systems	3
3134	Data structures	3
4118	operating systems	null

[Instructors] – <Teaches> ← [Courses]

```
Instructors(
  iid int primary key,
  name text
)
```

```
Courses_Teaches(
  cid int primary key,
  title text,
  iid int NOT NULL references Instructors
)
```

iid	name
1	Evan
2	Luis
3	Martha

cid	title	iid
4111	databases	1
3827	Systems	3
3134	Data structures	3
4118	operating systems	null

[Instructors] – <Teaches> == [Courses]

```
Instructors(
  iid int primary key,
  name text
)
```

```
Teaches(
  iid int references Instructors,
  cid int references Courses,
  primary key (iid, cid)
)
```

```
Courses(
  cid int primary key,
  title text
)
```

Is there redundancy in the schema?

iid	name
1	Evan
2	Luis
3	Martha

iid	cid
1	4111
2	4111
3	3827
3	4111
3	3134

cid	title
4111	databases
3827	Systems
3134	Data structures

[Instructors] – <Teaches> == [Courses]

```
Instructors(
  iid int primary key,
  name text
)
```

```
Teaches(
  iid int references Instructors,
  cid int references Courses,
  primary key (iid, cid)
)
```

```
Courses(
  cid int primary key,
  title text
)
```

What's the issue?

iid	name
1	Evan
2	Luis
3	Martha

iid	cid
1	4111
2	4111
3	3827
3	4111
3	3134

cid	title
4111	databases
3827	Systems
3134	Data structures
4118	Operating sys

At least one: Not easily representable

Permit many relationships: relationship is a separate table
Enforce a relationship: merge table, NOT NULL constraint

Effectively: a contradiction

(possible with multi-table assertions; not in this course)

[Instructors] ⟶ <Teaches> ⟵ [Courses]

```
Instructors(
  iid int primary key,
  name text
)
```

```
Courses(
  cid int primary key,
  title text
)
```

```
Teaches(
  iid int references Instructors,
  cid int references Courses,
  primary key (iid, cid)
)
```

Q1: What data is no longer valid?
Q2: How do we correct the schema?

iid	name
1	Evan
2	Luis
3	Martha

iid	cid
1	4111
2	4111
3	3827
3	4111
3	3134

cid	title
4111	databases
3827	Systems
3134	Data structures

[Instructors] ⟶ <Teaches> ⟵ [Courses]

```
Instructors(
  iid int primary key,
  name text
)
```

```
Courses(
  cid int primary key,
  title text
)
```

```
Teaches(
  iid int references Instructors,
  cid int references Courses,
  primary key (iid, cid)
)
```

Q1: What data is no longer valid?
Q2: How do we correct the schema?

iid	name
1	Evan
2	Luis
3	Martha

iid	cid
1	4111
2	4111
3	3827
3	4111
3	3134

cid	title
4111	databases
3827	Systems
3134	Data structures

[Instructors] ⟶ <Teaches> ⟵ [Courses]

```
Instructors(
  iid int primary key,
  name text
)

Courses(
  cid int primary key,
  title text
)

Teaches(
  iid int references Instructors,
  cid int references Courses,
  primary key (iid, cid)
  unique (iid),
  unique (cid)
)
```

iid	name
1	Evan
2	Luis
3	Martha

iid	cid
1	4111
2	4111
3	3827
3	4111
3	3134

cid	title
4111	databases
3827	Systems
3134	Data structures

[Instructors] ⟶ <Teaches> ⟵ [Courses]

```
Instructors(
  iid int primary key,
  name text
)

Courses(
  cid int primary key,
  title text
)

Teaches(
  iid int not null references Instructors,
  cid int references Courses,
  primary key (cid),
  unique (iid)
)
```

iid	name
1	Evan
2	Luis
3	Martha

iid	cid
1	4111
2	4111
3	3827
3	4111
3	3134

cid	title
4111	databases
3827	Systems
3134	Data structures

[Instructors] ⟶ <Teaches> ⟵ [Courses]

```
Instructors(
  iid int primary key,
  name text
)

Courses_Teaches(
  cid int primary key,
  title text,
  iid int references Instructors,
  unique (iid)
)
```

iid	name
1	Evan
2	Luis
3	Martha

cid	title	iid
4111	databases	1
3827	Systems	3
3134	Data structures	3
4118	operating systems	null