CS 5530

Database Systems
Spring 2020

Wrap up ER

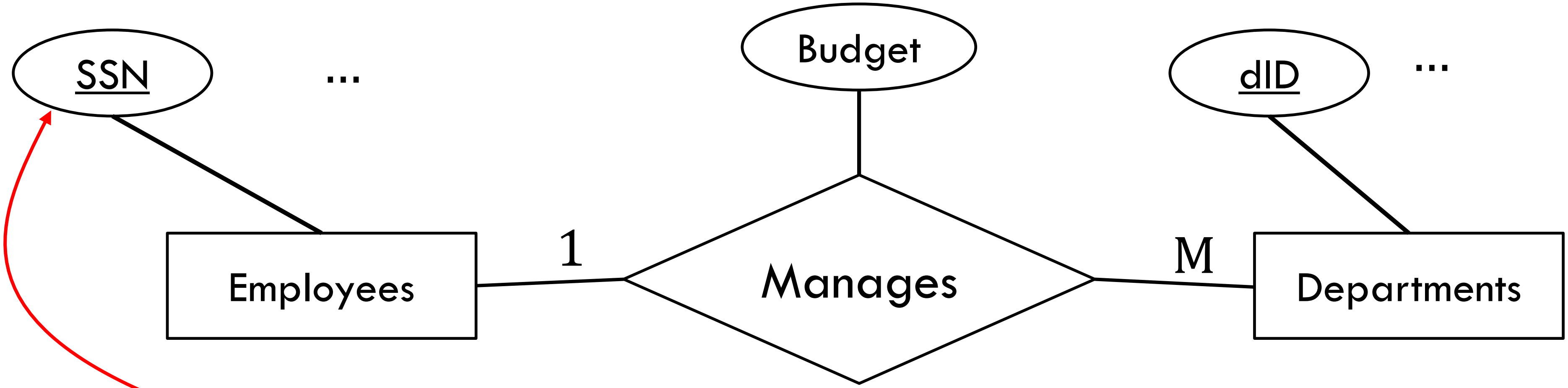
Intro to SQL

Team Formation

- Teamwork starts soon
- Team size = 2
- Find a partner on Piazza
- Declare your team on Canvas quiz

Relationship Set to Schema

- **1-to-Many**

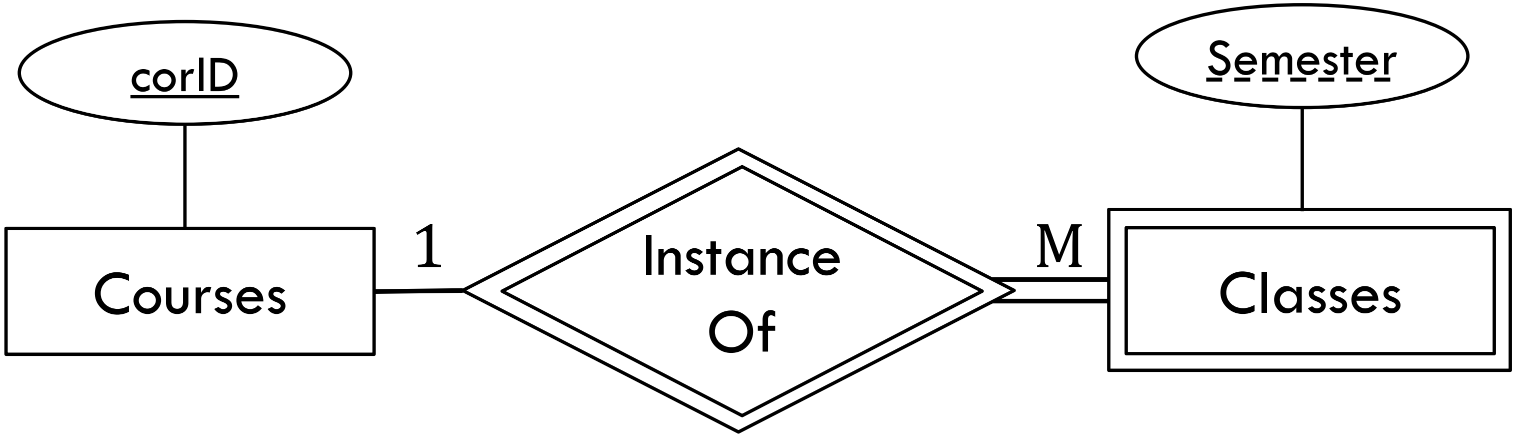


SSN	...
...	...
NN PK	NN

dID	...	Mnger	Budget
...	...		
NN PK	NN	FK	

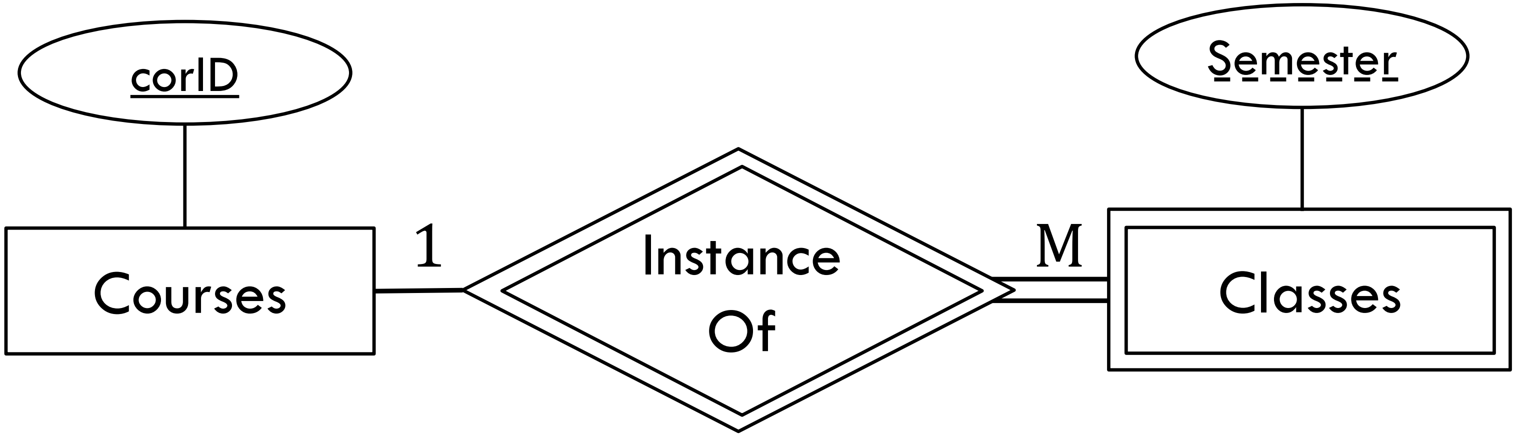
Weak Entity

- `Classes` is a weak entity set



- “Only one `Class` of each `Course` per `Semester`”

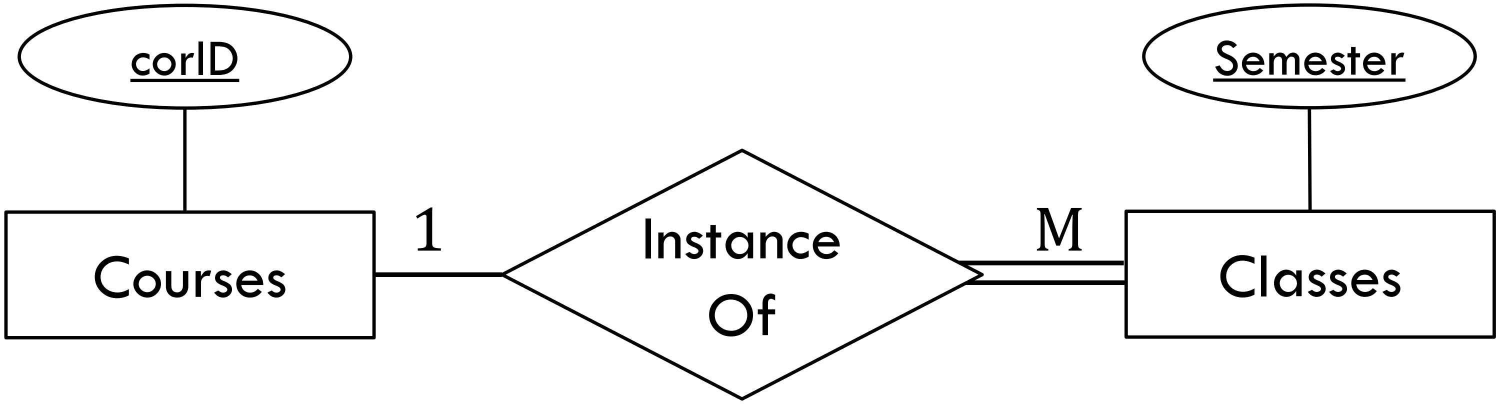
Weak Entity vs…?



corID sem

FK

— PK —



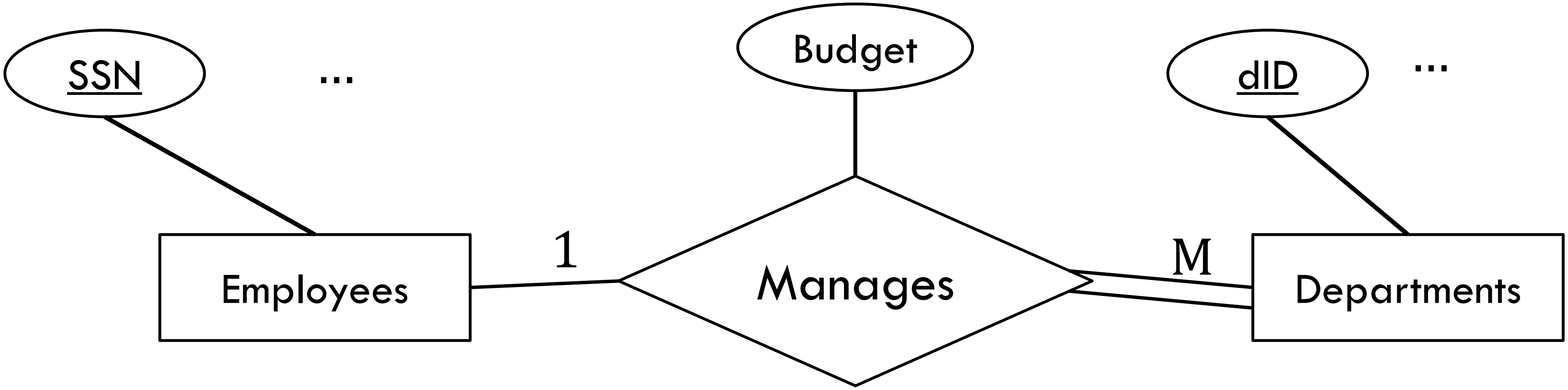
corID Sem

FK

PK

Weak Entity?

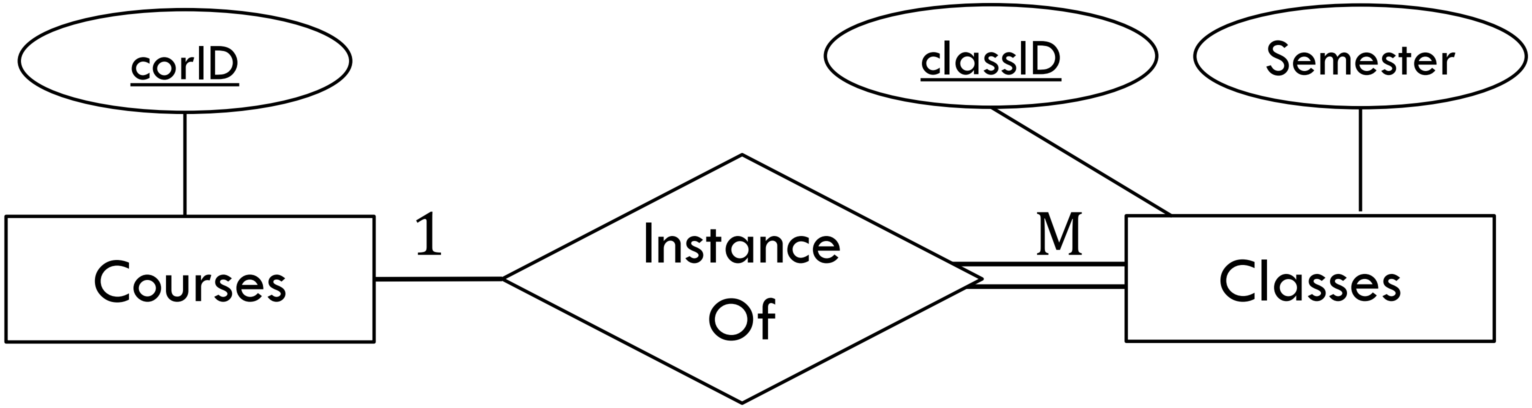
- Why isn't `Departments` a weak entity set?



- `Departments` uniqueness is based on `dID` alone
  - Has nothing to do with SSN

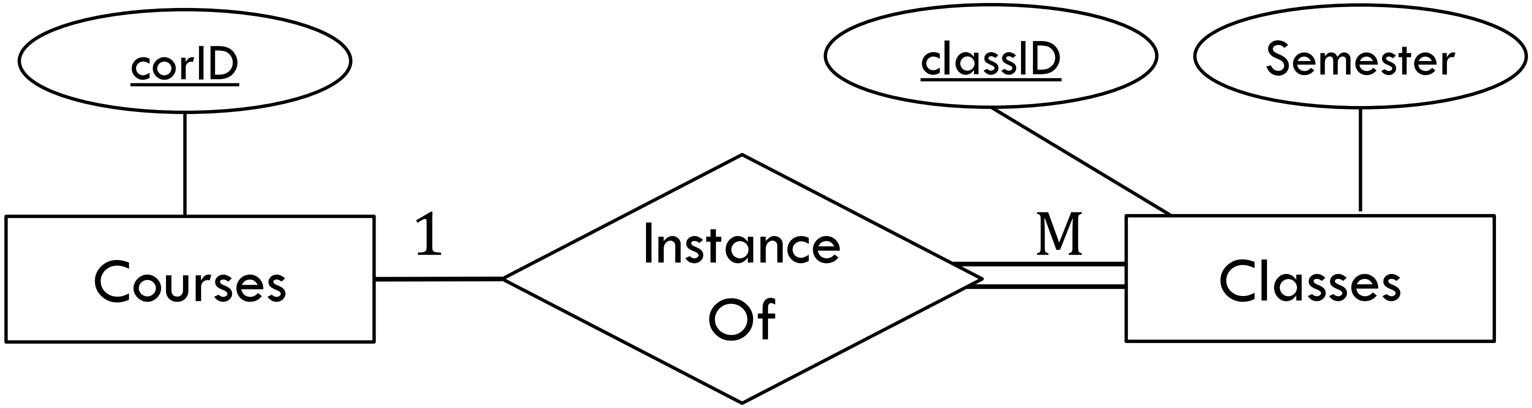
Weak Entity

- Why not just add an ID to `Classes`?



Weak Entity

- Why not just add an ID to `Classes`?

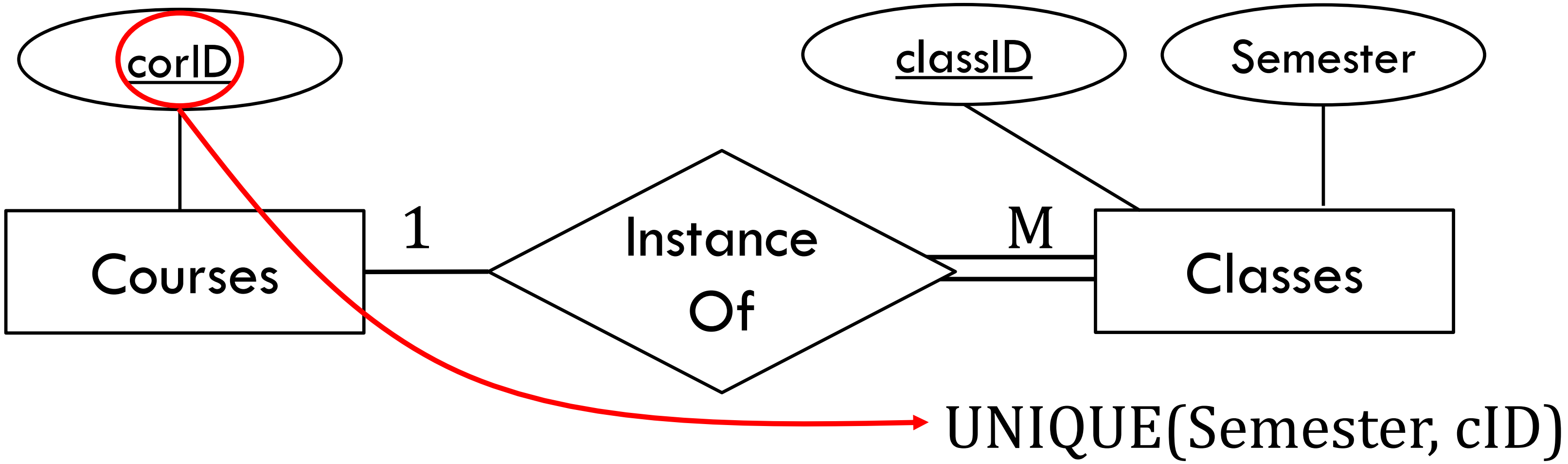


- What this means: “a `Class` belongs to a course and can’t have the same `classID` as any other `Class`”

- What we want: “a `Class` belongs to a `Course` and only one `Class` of each `Course` per `Semester`”

Weak Entity

- Why not just add an ID to `Classes`?

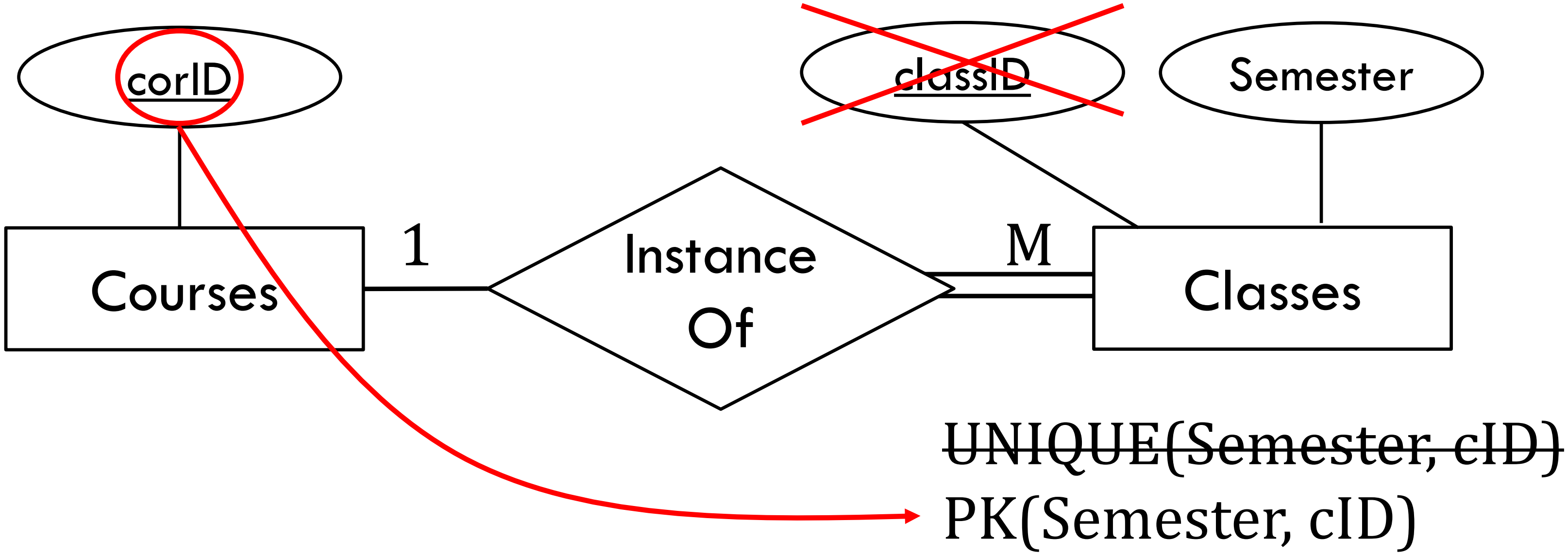


- only one `Class` of each `Course` per `Semester`”

- i.e., `Classes` uniqueness depends on `Courses` key

Weak Entity

- Why not just add an ID to `Classes`?



corID

~~classID~~

Semester

Courses

1

Instance Of

M

Classes

~~UNIQUE(Semester, cID)~~

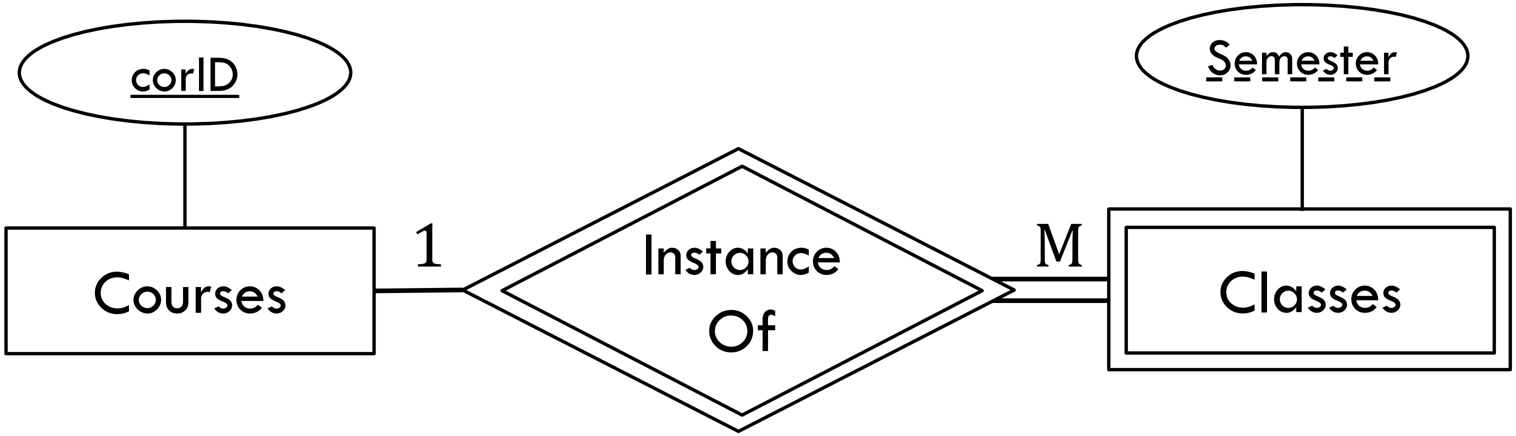
PK(Semester, cID)

- only one `Class` of each `Course` per `Semester`”

- i.e., `Classes` uniqueness depends on `Courses` key

Weak Entity

- Double borders is simply how we represent this



- i.e., `Classes` uniqueness depends on `Courses` key

Weak Entity vs. Participation

- Intuitive meaning

  - Weak Entity – a Class is an instance of *one specific Course*

  - Participation Constraint – a Department must have a Manager, but the Manager can change

Weak Entity vs. Participation

- Intuitive meaning
  - Weak Entity – a Class is an instance of *one specific Course*
  - Participation Constraint – a Department must have a Manager, but the Manager can change
- i.e., a *Class* can not change which *Course* it is tied to

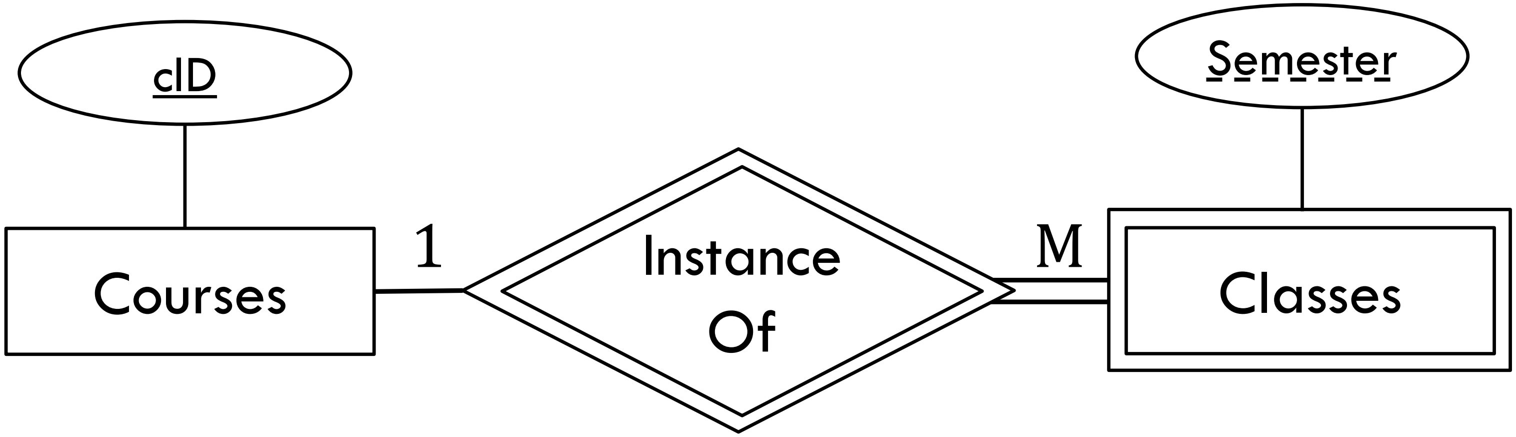
Weak Entity vs. Participation

- Formal meaning

  - Weak Entity – its uniqueness depends on a foreign key (plus its own)

  - Participation Constraint – does not say anything about uniqueness

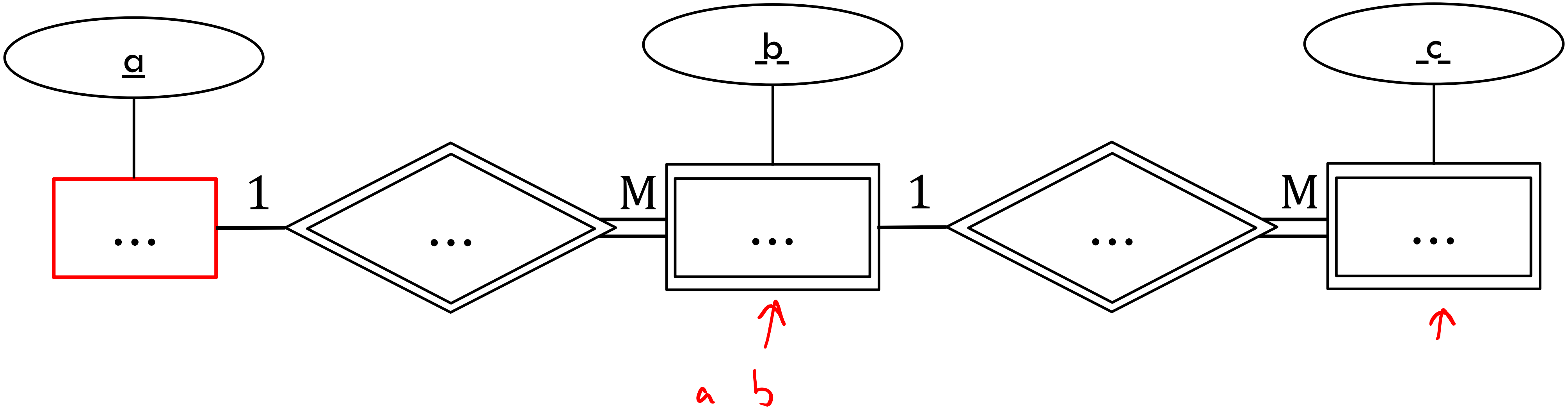
Weak Entity Key



`key = {Semester, cID}`

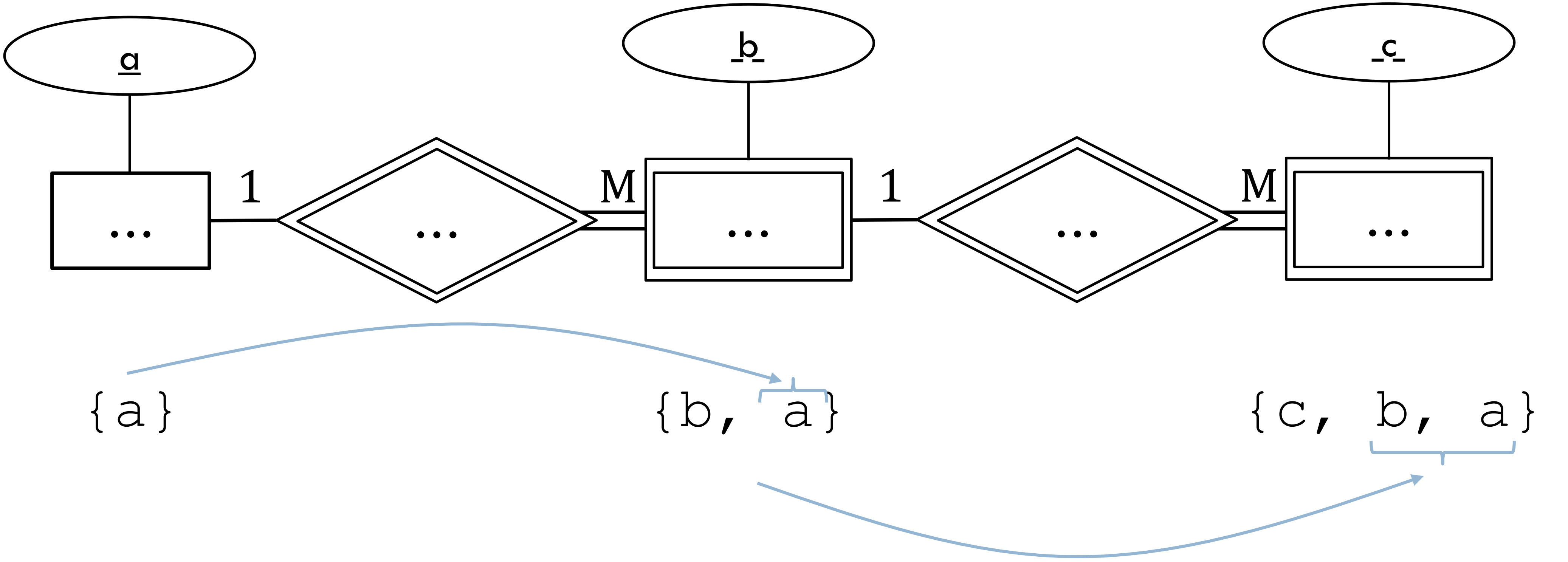
Weak Entity Chain

- A weak entity can be supported by another weak entity
    - As long as the chain is anchored by a strong entity



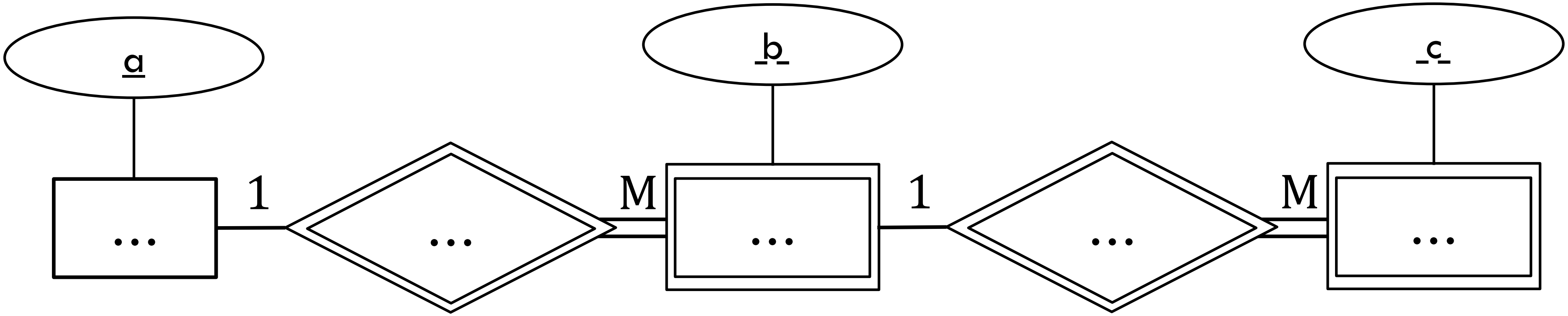
Weak Entity Chain

- Keys get progressively more complex down the chain



Weak Entity Chain

- Keys get progressively more complex down the chain

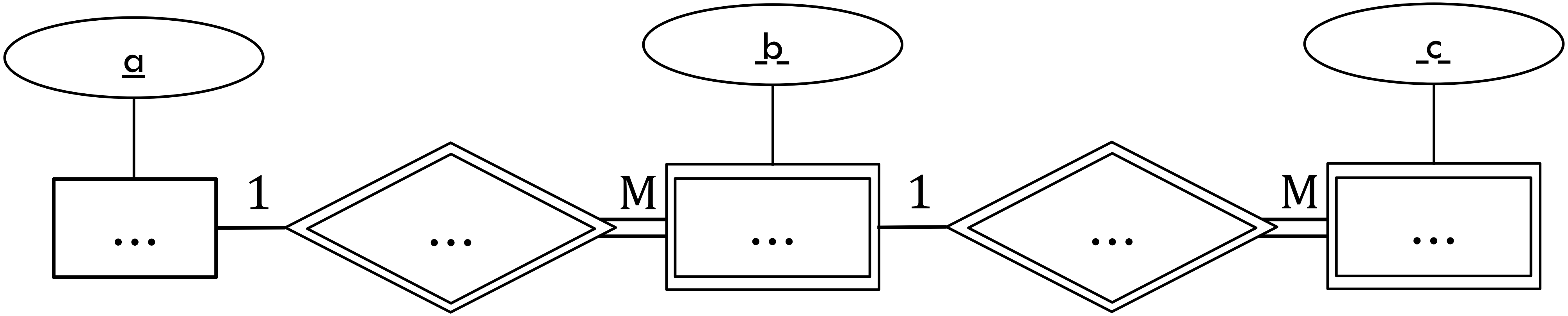


{a} {b, a} {c, b, a}

- This only becomes a concern in actual database
  - In ER it doesn’t matter

Translating Weak Entity Chain

- Primary keys should be simple



a
…

PK

b	a
…	…

--PK--

c	b	a
…	…	…

----PK----

- Worry about this later…

ER Logic?

- SQL tables can’t always represent constraints in an ER diagram

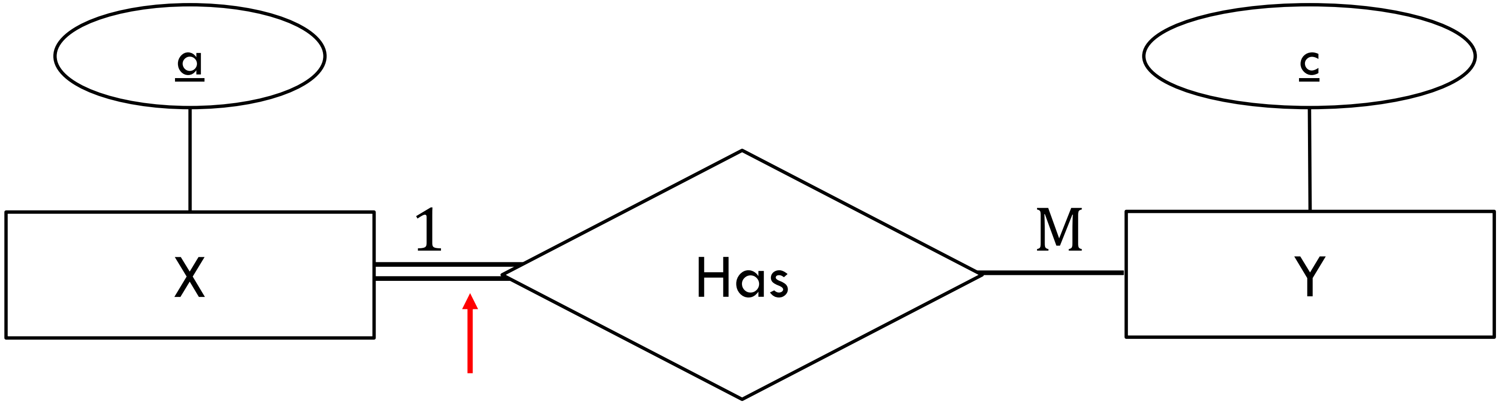
- This is OK!

  - We match the tables as closely as possible

  - Don’t compromise on the ER diagram

ER Logic?

- How would we capture this in SQL tables?



X

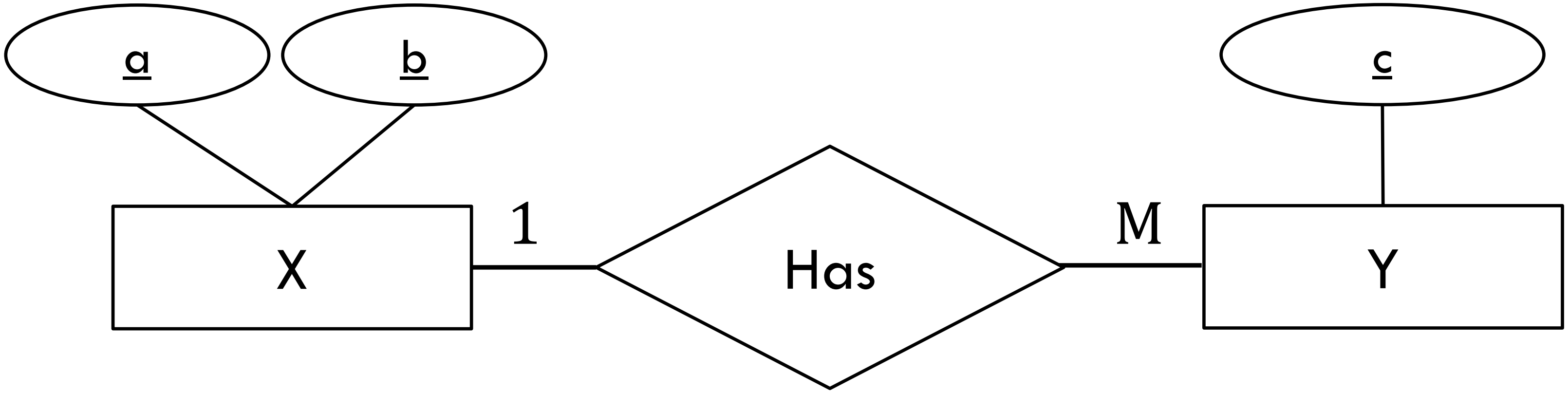
a
PK

Y

a	c
FK	PK

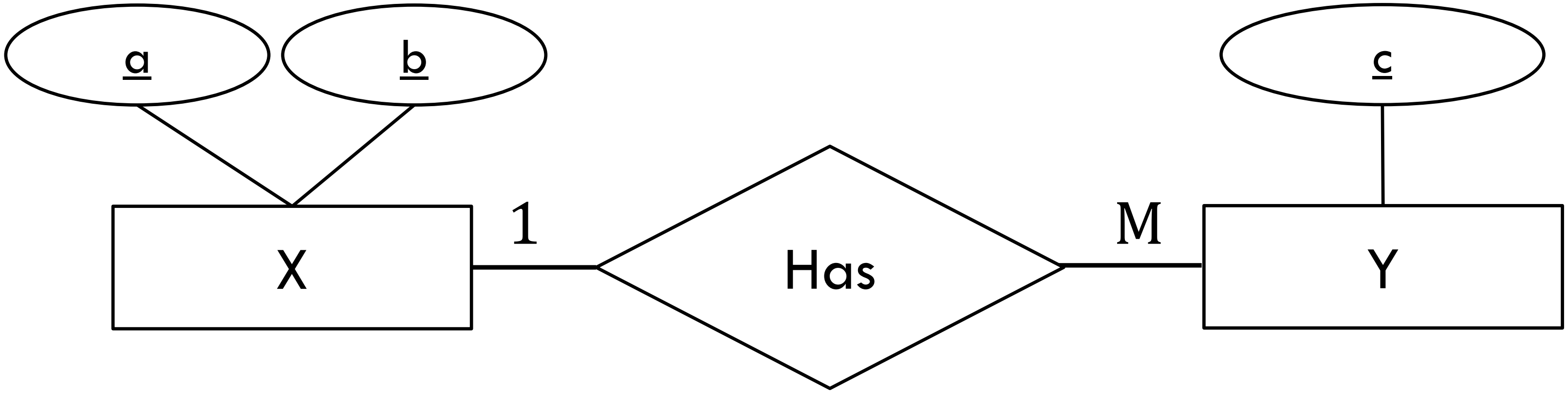
Practice

- What are the foreign key(s) for Y?



Practice

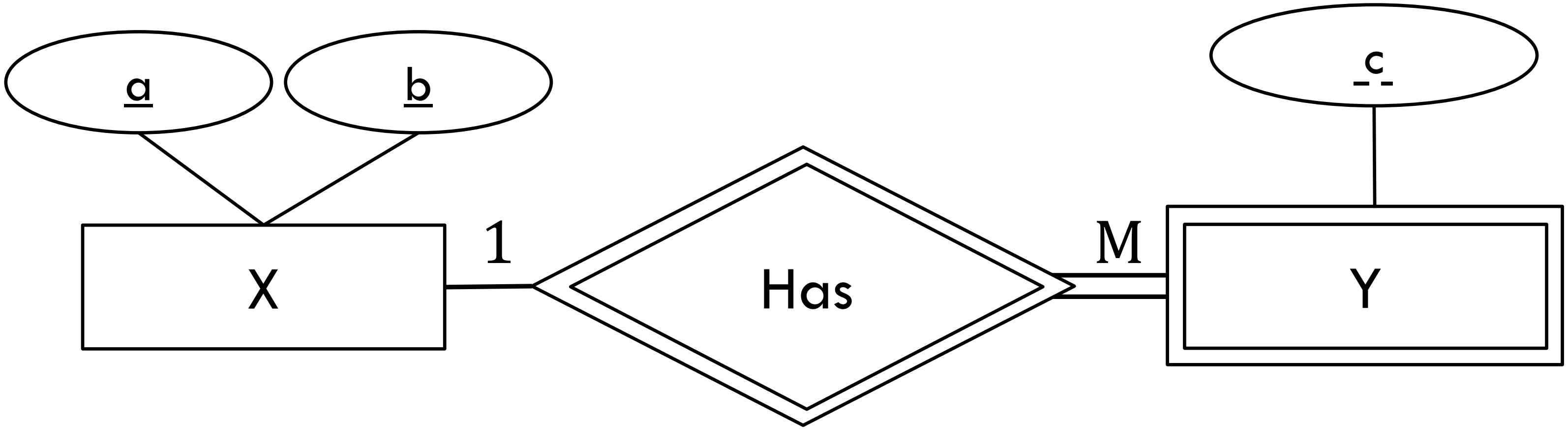
- What are the foreign key(s) for Y?



```
CREATE TABLE Y(
   c … NOT NILL,
   a …,
   b …,
   FOREIGN KEY (a, b) REFERENCES X(a, b),
   PRIMARY KEY (c));
```

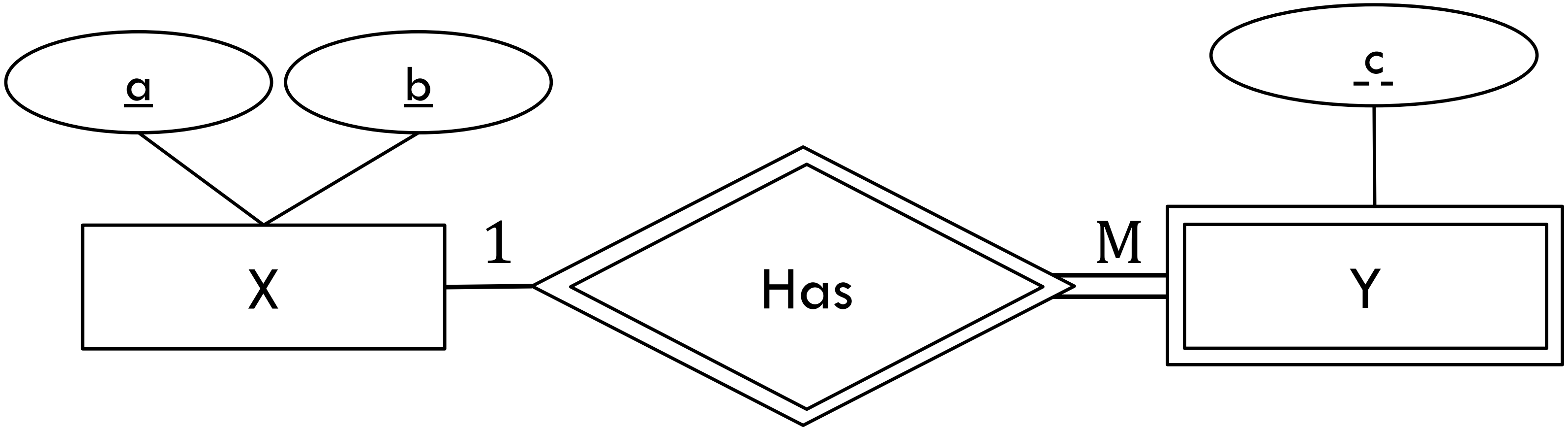
Practice

- What is the foreign + primary key for Y?



Practice

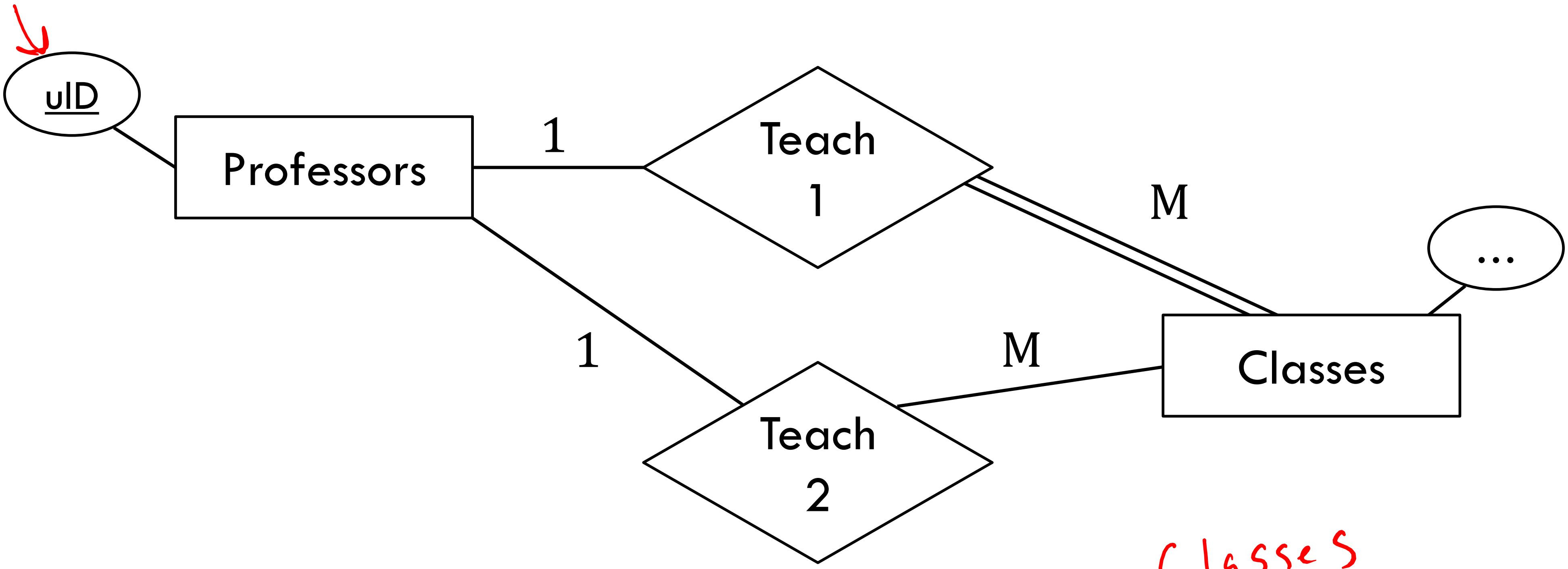
- What is the foreign + primary key for `Y`?



```
CREATE TABLE Y(
   c … NOT NULL,
   a … NOT NULL,
   b … NOT NULL,
   FOREIGN KEY (a, b) REFERENCES X(a, b),
   PRIMARY KEY (a, b, c));
```

Practice

- What are the foreign key(s) for `Classes`?



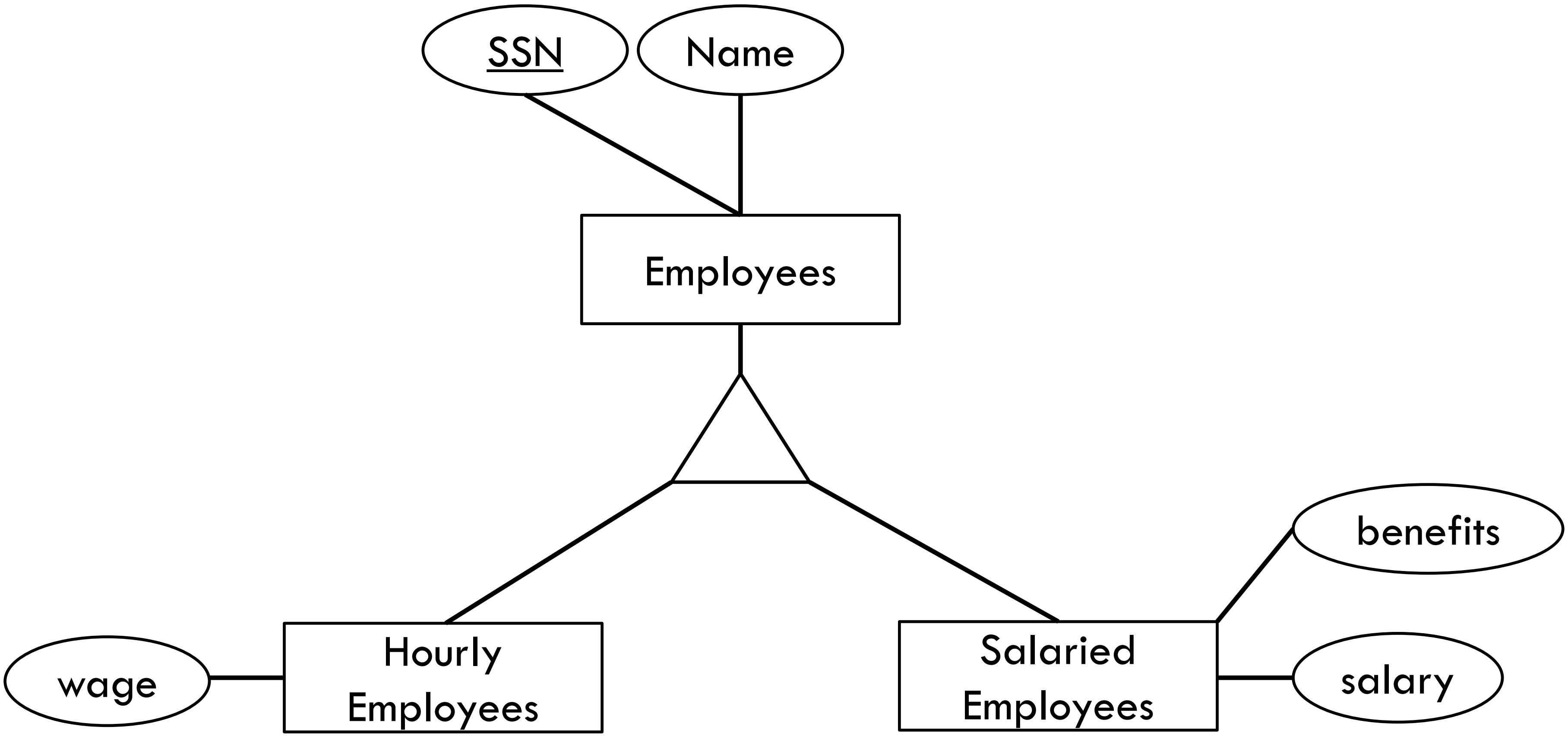
Classes

T1
FK
NN

T2
FK

IS-A to Schemas

- How do we translate this to schemas?

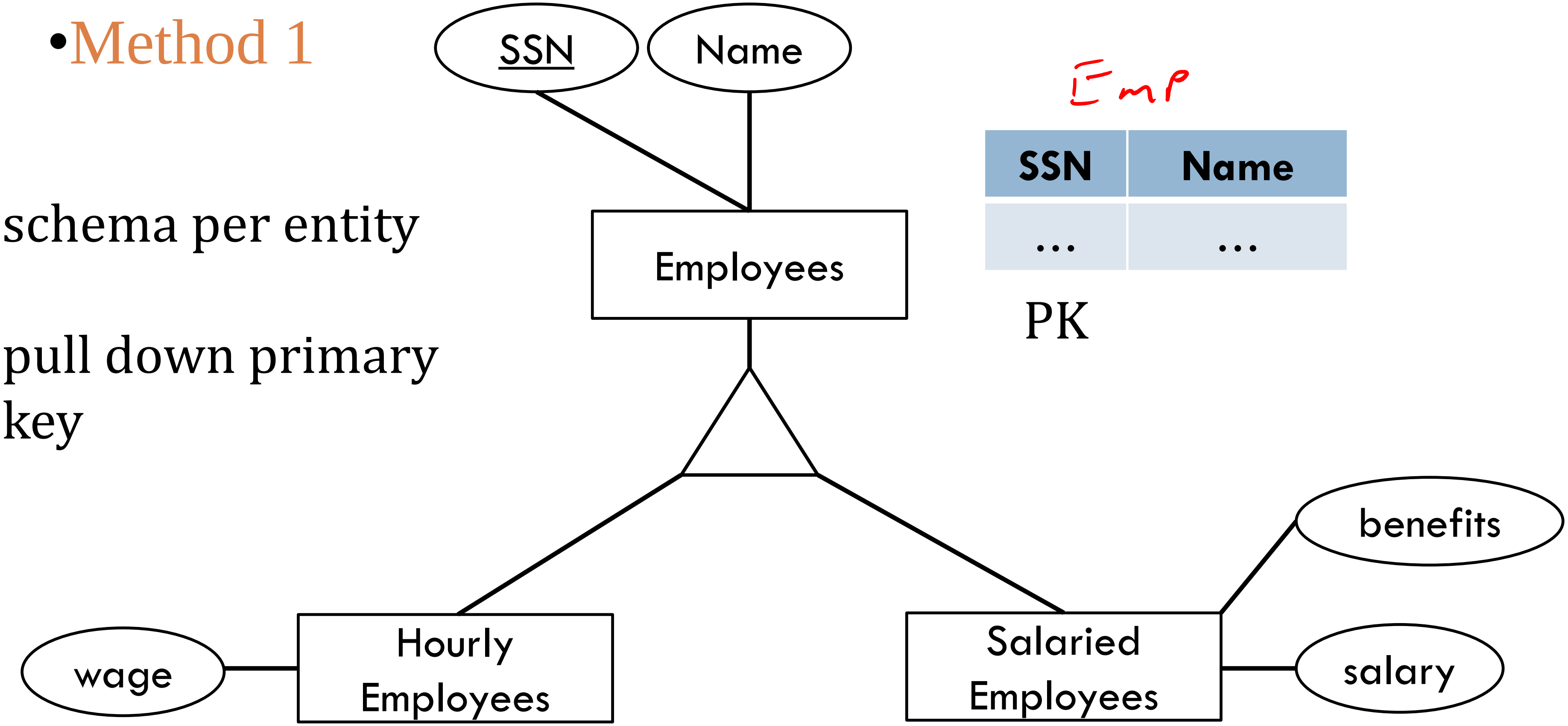


IS-A to Schemas

- Method 1

schema per entity

pull down primary key



Emp

SSN	Name
…	…

PK

SSN	Wage
…	…

PK
FK

SSN	Benefits	Salary
…	…	

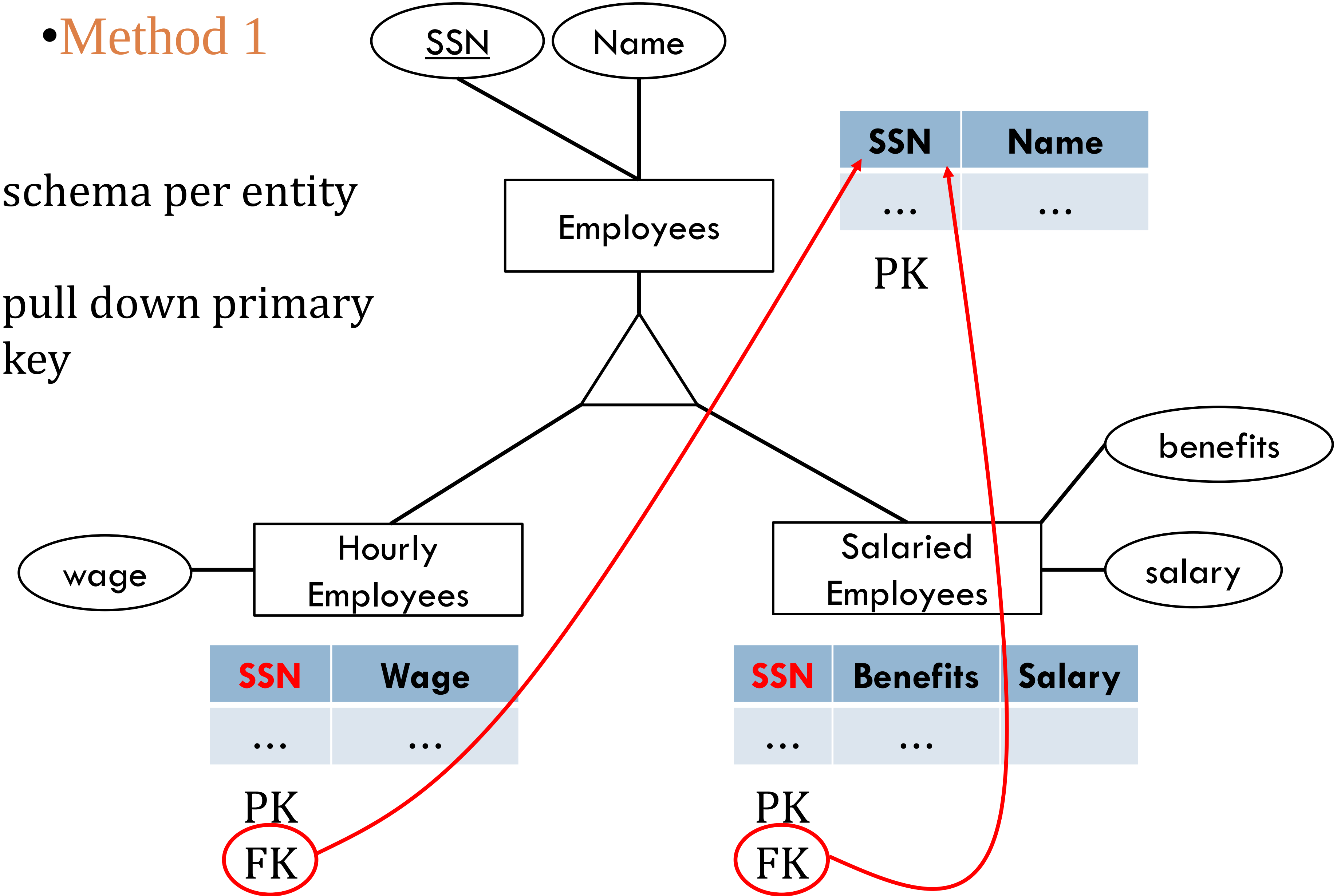
PK
FK

IS-A to Schemas

- Method 1

schema per entity

pull down primary key



SSN	Name
…	…

PK

SSN	Wage
…	…

PK

FK

SSN	Benefits	Salary
…	…	

PK

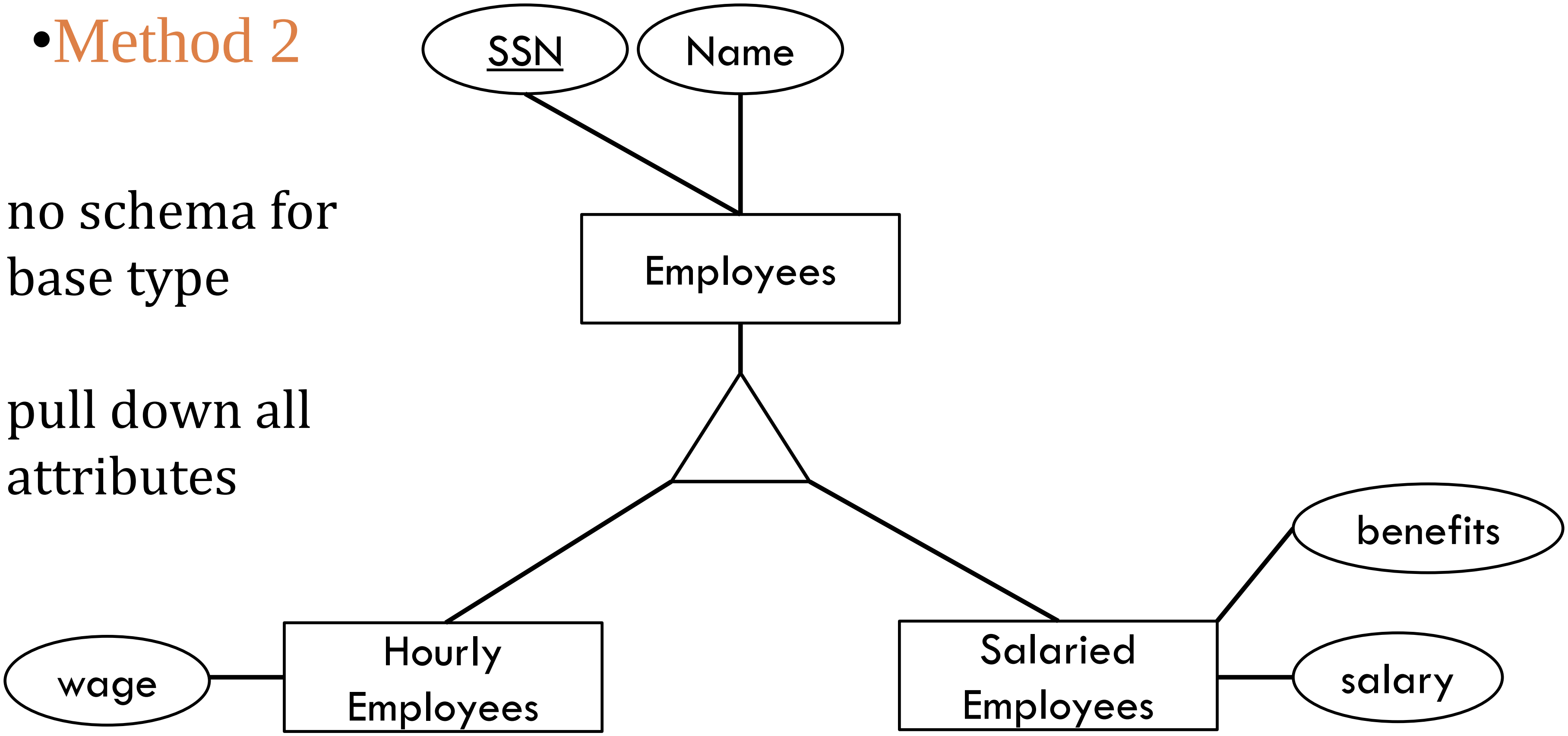
FK

IS-A to Schemas

- Method 2

no schema for base type

pull down all attributes



SSN	Name	Wage
…		…

PK

SSN	Name	Benefits	Salary
…		…	

PK

IS-A to Schemas

- Method 1:
  - If an entity can be two different derived types
  - If the base type has relationships

- Method 2:
  - If the base type is “abstract”

A Brief History of DBMS

- 1970s – commercial database systems becoming popular
    - Many different languages - impractical

A Brief History of DBMS

- 1970s – commercial database systems becoming popular
  - Many different languages - impractical
- 1969 – Relational Model (Edgar Codd)

A Brief History of DBMS

- 1970s – commercial database systems becoming popular
  - Many different languages - impractical
- 1969 – Relational Model (Edgar Codd)
- SQL – one of the first languages to implement the relational model – “Structured Query Language”

A Brief History of DBMS

- 1970s – commercial database systems becoming popular
    - Many different languages - impractical
- 1969 – Relational Model (Edgar Codd)
- SQL – one of the first languages to implement the relational model – “Structured Query Language”
- ANSI standardized SQL in 1986
    - Revised many times, most recently in 2016

A Brief History of DBMS

- 2010s
  - *Tons* of new entrants, mostly non-relational
  - Firebase
  - MongoDB
  - Oracle NoSQL
  - Redis
  - …

DBMS vs. Query Language

- SQL is the language (like C++)
- DBMS is the implementation (like gnu/g++)

DBMS vs. Query Language

- SQL is the language (like C++)
- DBMS is the implementation (like gnu/g++)
- Many SQL implementations
  - MySQL / MariaDB
  - PostgreSQL
  - SQL Server
  - Oracle

DBMS vs. Query Language

- SQL is the language (like C++)
- DBMS is the implementation (like gnu/g++)
- Many SQL implementations
  - MySQL
  - PostgreSQL
  - SQL Server
  - Oracle

Varying behavior, despite the standard…

SQL

- “SQL” vs. “sequel”?

SQL

- “SQL” vs. “sequel”?
- Originally called “Structured English QUEry Language”
- Forced to change name – no longer an acronym

SQL

- "SQL" vs. "sequel"?
- Originally called "Structured English QUEry Language"
- Forced to change name – no longer an acronym
  - I prefer "SQL" pronunciation

SQL

- SQL is a combination of languages:
  - DDL: Data Definition Language
    - create/modify tables and settings
  - DML: Data Manipulation Language
    - create/modify/delete tuples
    - search for tuples

SQL

- Data Definition Language (DDL)

```
create table <name> (
   <column1Name> <type> <properties>,
   <column2Name> <type> <properties>,
   …
   <table properties>
);

drop table <name>;

alter table <name>
   add …
   drop column …
```

DDL – Create Tables

```
CREATE TABLE table_name (
  col_name type [DEFAULT default_expr]
  [col_constraint [, ...]] |
  table_constraint [, ...]);
```

DDL – Column Constraints

- Column constraints:

```
col_constraint =
NOT NULL | NULL | UNIQUE | PRIMARY KEY
| CHECK (expression) | AUTO_INCREMENT |
REFERENCES reftable [ ( refcolumn ) ] [
ON DELETE action ] [ ON UPDATE action ]
```

DDL – Column Constraints

- Column constraints:

```
col_constraint =
NOT NULL | NULL | UNIQUE | PRIMARY KEY
| CHECK (expression) | AUTO_INCREMENT |
REFERENCES reftable [ ( refcolumn ) ] [
ON DELETE action ] [ ON UPDATE action ]
```

- Silently ignored by MySQL

DDL – Column Constraints

- Column constraints:

```
col_constraint =
NOT NULL | NULL | UNIQUE | PRIMARY KEY
| AUTO_INCREMENT
```

DDL – Table Constraints

- Table constraints:

```
table_constraint =
[CONSTRAINT constraint_name]
UNIQUE(col_name [, ...]) |
PRIMARY KEY(col_name [, ...]) |
FOREIGN KEY(column_name [, ...])
REFERENCES reftable(refcolumn [, ... ])]
   [ON DELETE action]
   [ON UPDATE action]
```

DML

- Data Manipulation Language (DML)

```
select …

insert into …

delete from …
```

…

select

```
SELECT     [DISTINCT]  target-list
FROM       relation-list
[WHERE     qualification]
[ORDER BY  column] [DESC]
[LIMIT     number]
```

select

```
SELECT     [DISTINCT]  target-list
FROM       relation-list
[WHERE     qualification]
[ORDER BY  column] [DESC]
[LIMIT     number]
```

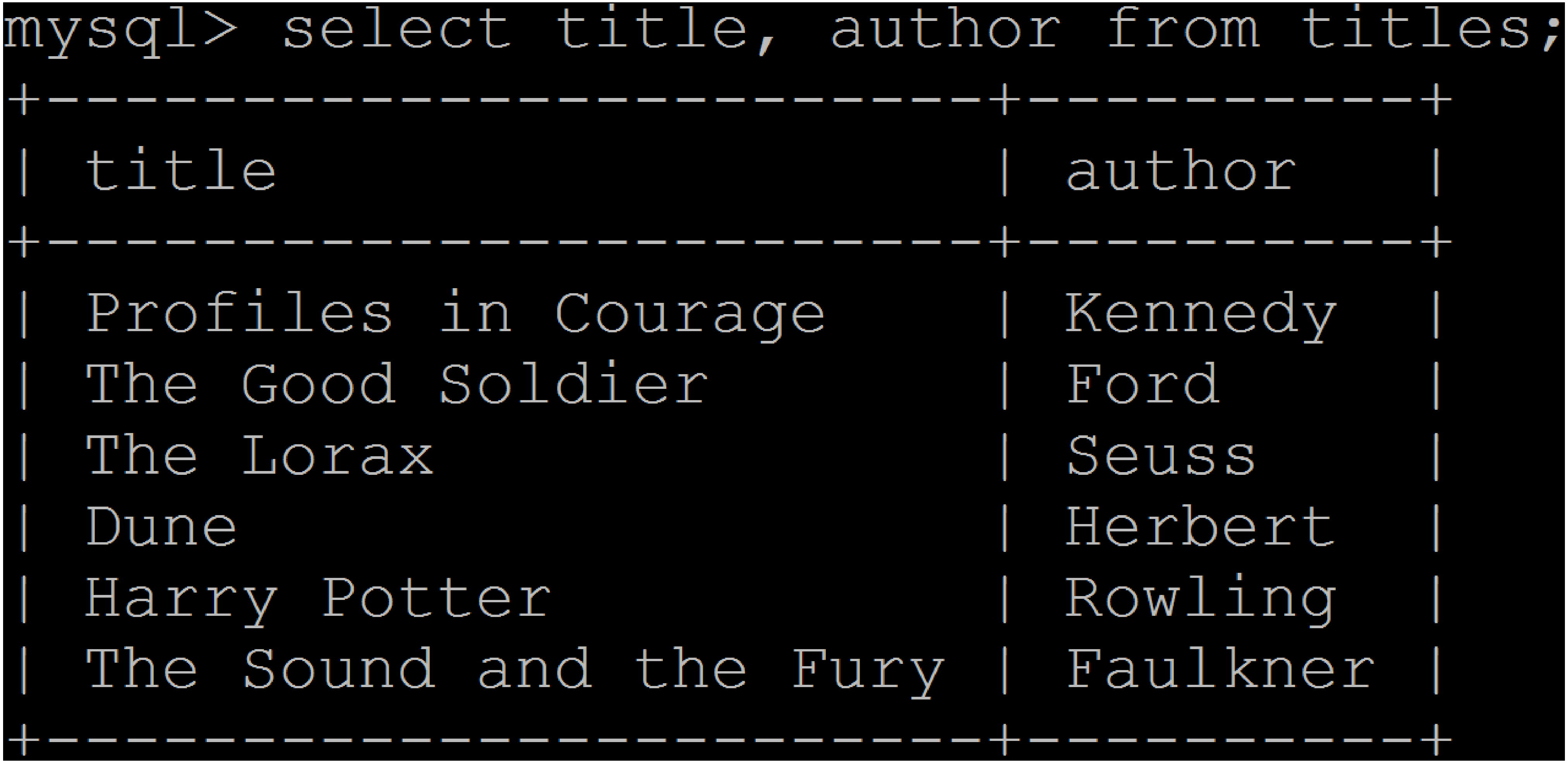
- `target-list` and `relation-list` are comma separated

- `qualification` is a Boolean expression

- Demo

Retrieve Columns From Table

- Select specific columns by name:

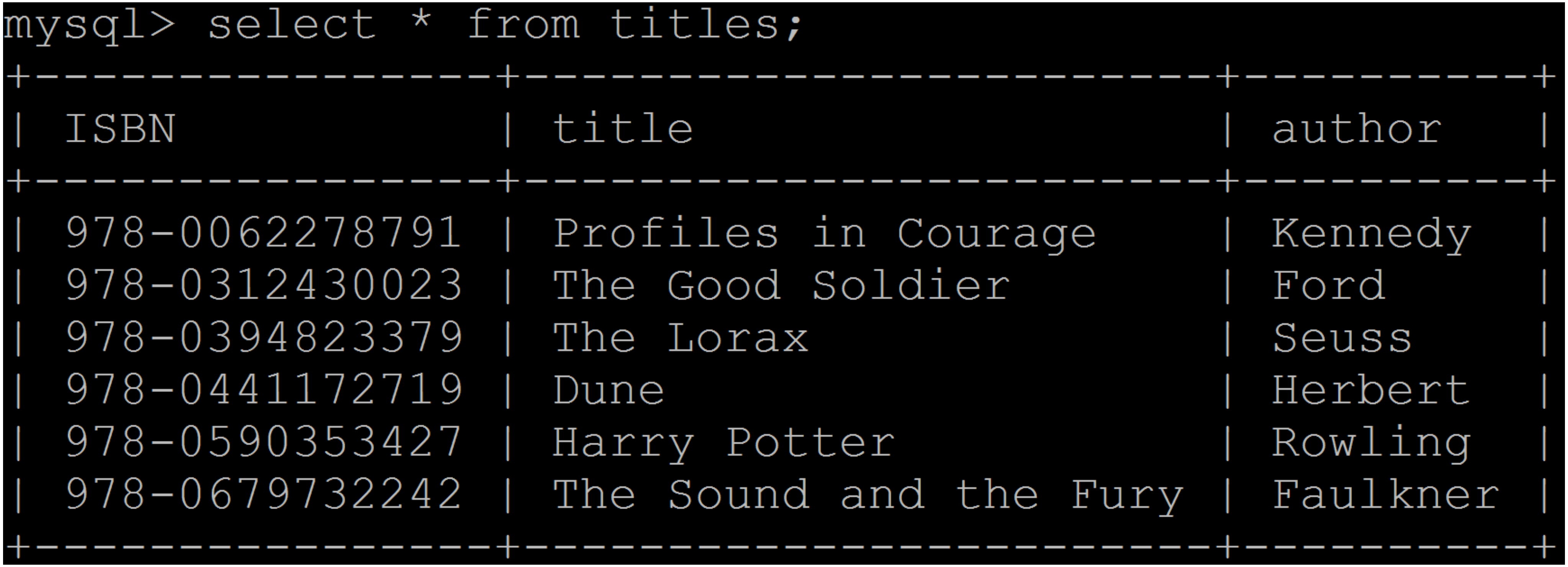


```
mysql> select title, author from titles;
+------------------------+----------+
| title                  | author   |
+------------------------+----------+
| Profiles in Courage    | Kennedy  |
| The Good Soldier       | Ford     |
| The Lorax              | Seuss    |
| Dune                   | Herbert  |
| Harry Potter           | Rowling  |
| The Sound and the Fury | Faulkner |
+------------------------+----------+
```

Retrieve Entire Table

- `select <columns> from <table>`
- `<columns>` respects wildcards: `*`

Retrieve Entire Table



```
mysql> select * from titles;
+----------------+------------------------+----------+
| ISBN           | title                  | author   |
+----------------+------------------------+----------+
| 978-0062278791 | Profiles in Courage    | Kennedy  |
| 978-0312430023 | The Good Soldier       | Ford     |
| 978-0394823379 | The Lorax              | Seuss    |
| 978-0441172719 | Dune                   | Herbert  |
| 978-0590353427 | Harry Potter           | Rowling  |
| 978-0679732242 | The Sound and the Fury | Faulkner |
+----------------+------------------------+----------+
```

Multiple Tables

- What if we want info defined by a relationship to another table?
  - e.g., what is Joe’s phone number?

Phones

CardNum	Phone
1	555-5555
2	666-6666
3	777-7777
4	888-8888
4	999-9999

Patrons

Name	CardNum
Joe	1
Ann	2
Ben	3
Dan	4

Relations

- Remember that we are dealing with relations
  - schema + instance
- `select` commands input a relation and output a relation

Multiple Tables

- `select` inputs a relation and outputs a relation
- We need a new relation that combines these two

Phones

CardNum	Phone
1	555-5555
2	666-6666
3	777-7777
4	888-8888
4	999-9999

Patrons

Name	CardNum
Joe	1
Ann	2
Ben	3
Dan	4

Combine Relations

Patrons

CN	Name
3	a
5	b

Phones

CN	PN
3	x
4	y
5	z

Pa. CN	Name	Ph. CN	PN
3	a	3	x
3	a	4	y
3	a	5	z
5	b	3	x
5	b	4	y
5	b	5	z

Join Tables

- `<table> join <table>` creates a temporary table
- `Phones join Patrons` – full cross product of rows

Phones

CardNum	Phone
1	555-5555
2	666-6666
3	777-7777
4	888-8888
4	999-9999

Patrons

Name	CardNum
Joe	1
Ann	2
Ben	3
Dan	4

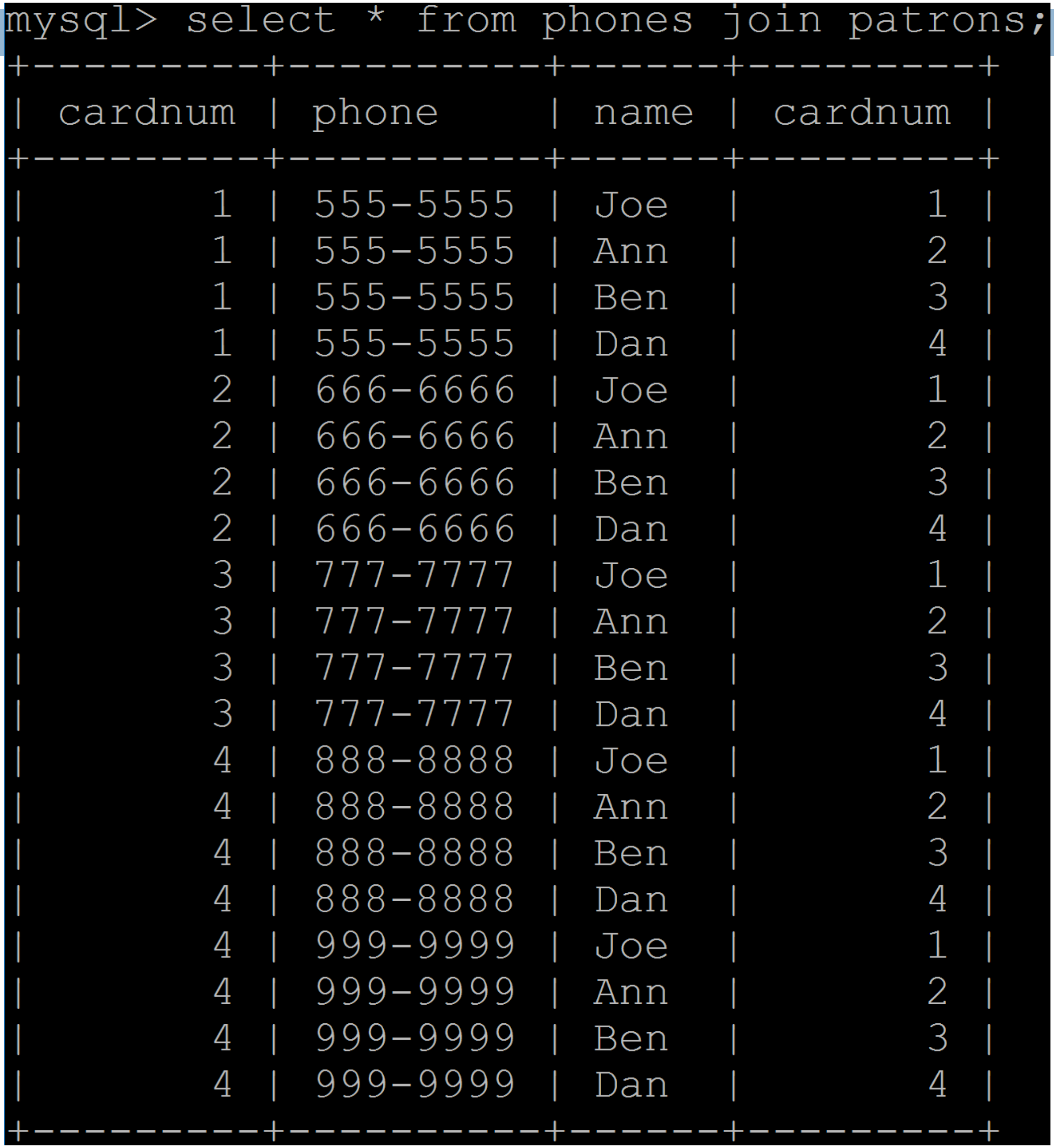
Join Tables

Phones

CardNum	Phone
1	555-5555
2	666-6666
3	777-7777
4	888-8888
4	999-9999

Patrons

Name	CardNum
Joe	1
Ann	2
Ben	3
Dan	4



```
mysql> select * from phones join patrons;
+---------+----------+------+---------+
| cardnum | phone    | name | cardnum |
+---------+----------+------+---------+
|       1 | 555-5555 | Joe  |       1 |
|       1 | 555-5555 | Ann  |       2 |
|       1 | 555-5555 | Ben  |       3 |
|       1 | 555-5555 | Dan  |       4 |
|       2 | 666-6666 | Joe  |       1 |
|       2 | 666-6666 | Ann  |       2 |
|       2 | 666-6666 | Ben  |       3 |
|       2 | 666-6666 | Dan  |       4 |
|       3 | 777-7777 | Joe  |       1 |
|       3 | 777-7777 | Ann  |       2 |
|       3 | 777-7777 | Ben  |       3 |
|       3 | 777-7777 | Dan  |       4 |
|       4 | 888-8888 | Joe  |       1 |
|       4 | 888-8888 | Ann  |       2 |
|       4 | 888-8888 | Ben  |       3 |
|       4 | 888-8888 | Dan  |       4 |
|       4 | 999-9999 | Joe  |       1 |
|       4 | 999-9999 | Ann  |       2 |
|       4 | 999-9999 | Ben  |       3 |
|       4 | 999-9999 | Dan  |       4 |
+---------+----------+------+---------+
```

Syntax

- SQL has lots of ways of doing things
- `select * from Phones join Patrons;`
- `select * from Phones, Patrons;`

Implies
`join`

Syntax

- SQL has lots of ways of doing things
- `select * from Phones join Patrons;`
- `select * from Phones, Patrons;`
- `select * from Phones inner join Patrons;`

Syntax

- SQL has lots of ways of doing things
- `select * from Phones join Patrons;`
- `select * from Phones, Patrons;`
- `select * from Phones inner join Patrons;`

◯ = *one* temporary relation

Syntax

- Relational model terminology:
  - join

  or

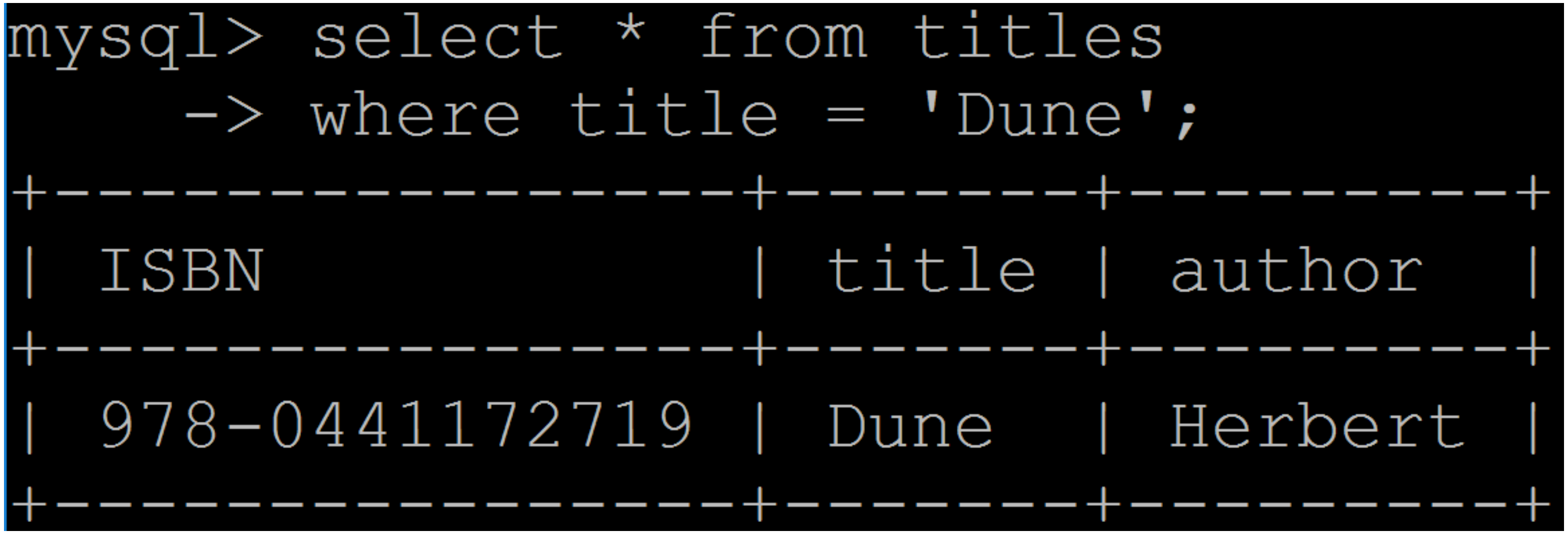
  - cross product

where

- Apply a conditional filter to any select
- `SELECT … FROM … WHERE <condition>`
- e.g.

```
WHERE CardNum > 2
WHERE Title = ‘Dune’
```

where



```
mysql> select * from titles
    -> where title = 'Dune';
+----------------+--------+---------+
| ISBN           | title  | author  |
+----------------+--------+---------+
| 978-0441172719 | Dune   | Herbert |
+----------------+--------+---------+
```

where

- This applies to joined tables too!

```
SELECT * FROM Patrons JOIN Phones
WHERE Patrons.CardNum=Phones.CardNum;
```

Join

- SQL:
  - `… JOIN … WHERE …`
- Relational model:
  - theta join
  - (join with condition)

Conditions

- `SELECT <columns> FROM <table>`
  `WHERE <condition>`

- `<condition>` can be complex (combined with AND, OR, etc)

- Conditions comprised of comparison(s):
  - <, >, <=, >=, =, !=

- e.g.
  - `WHERE Name='Joe' AND  Age >= 20`

Logical Operations

- AND
- OR
- NOT

```
SELECT * FROM Patrons JOIN Phones
WHERE Patrons.CardNum = Phones.CardNum
AND Patrons.Name = ‘Joe’;
```

JOIN ON

- Adds a filter to the join
- `<table> JOIN <table> ON <condition>`

JOIN ON

- Adds a filter to the join
- `<table> JOIN <table> ON <condition>`
- `<condition>`
    - Usually want to compare a column from each table
    - `Phones.CardNum = Patrons.CardNum`

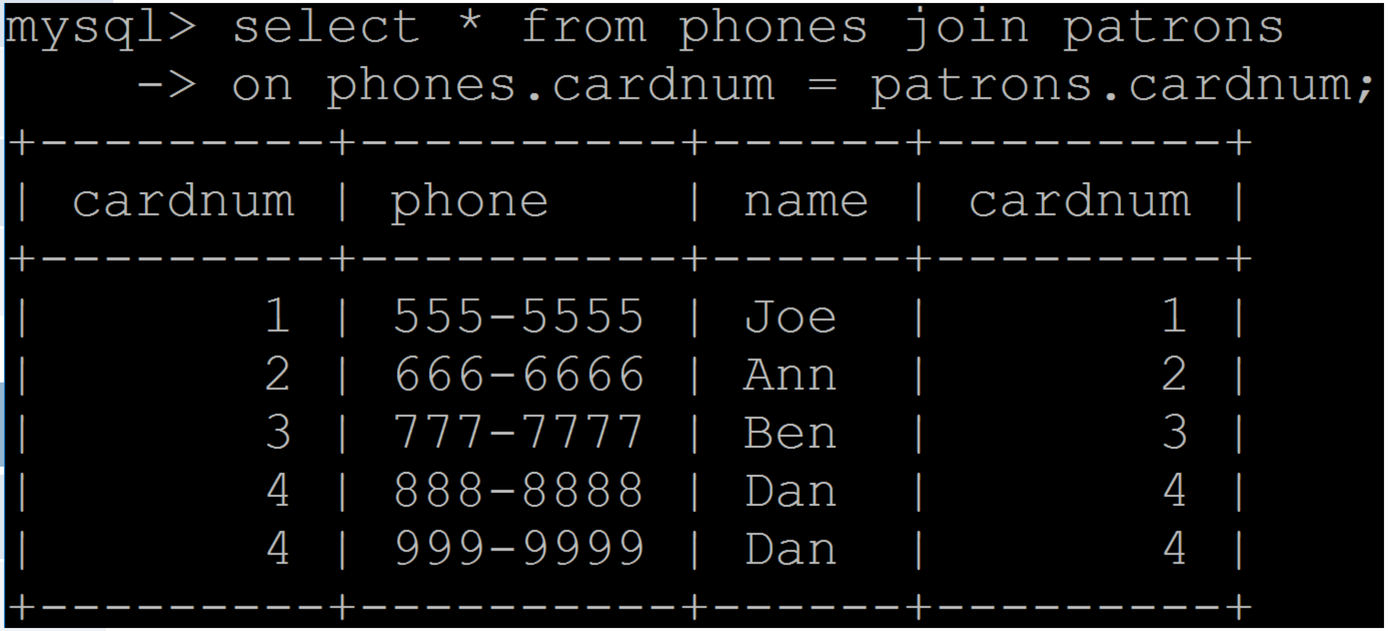
JOIN ON

Patrons

Name	CardNum
Joe	1
Ann	2
Ben	3
Dan	4

Phones

CardNum	Phone
1	555-5555
2	666-6666
3	777-7777
4	888-8888
4	999-9999



```
mysql> select * from phones join patrons
    -> on phones.cardnum = patrons.cardnum;
+---------+----------+------+---------+
| cardnum | phone    | name | cardnum |
+---------+----------+------+---------+
|       1 | 555-5555 | Joe  |       1 |
|       2 | 666-6666 | Ann  |       2 |
|       3 | 777-7777 | Ben  |       3 |
|       4 | 888-8888 | Dan  |       4 |
|       4 | 999-9999 | Dan  |       4 |
+---------+----------+------+---------+
```

Syntax

- I will usually prefer `WHERE` syntax

```
Patrons JOIN Phones ON
  Patrons.CardNum = Phones.CardNum
```

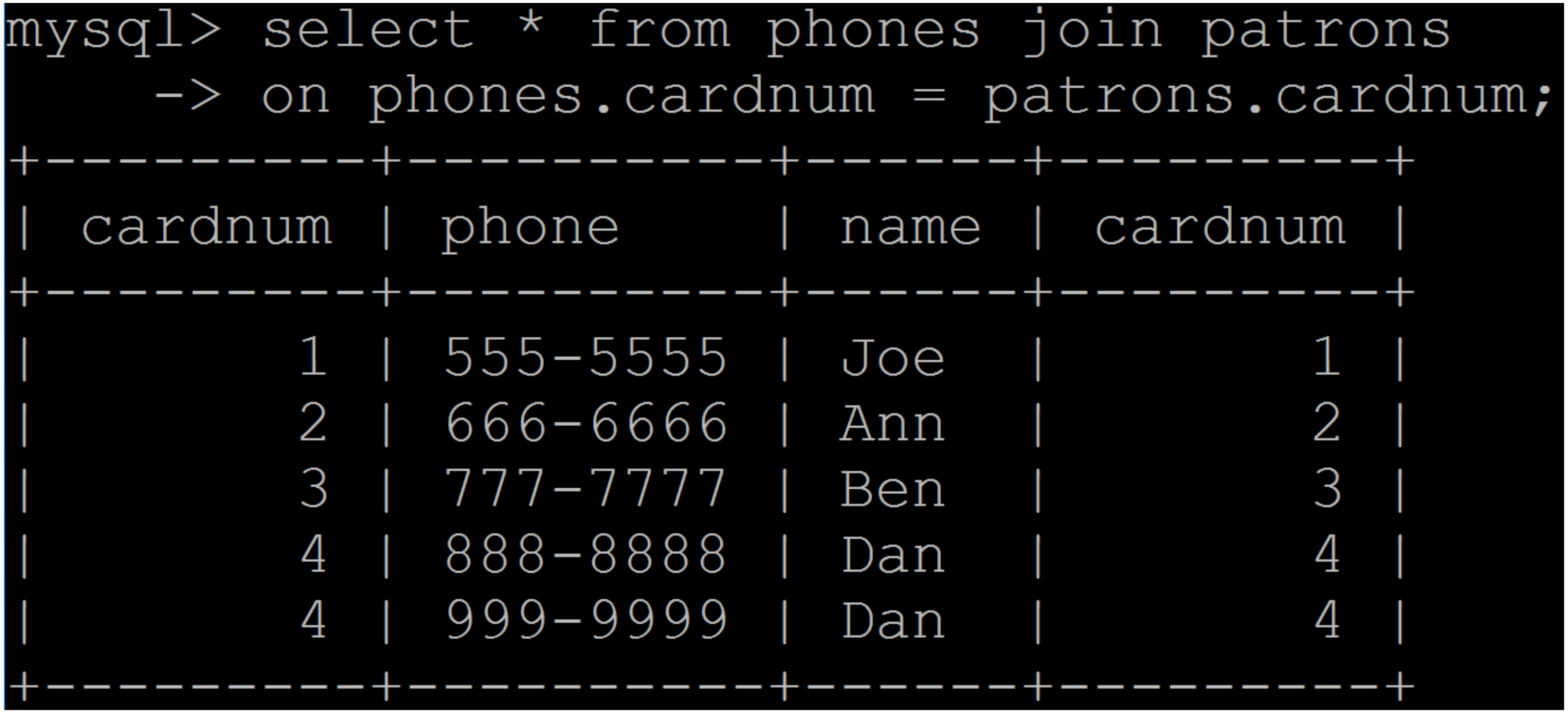
- vs

```
Patrons JOIN Phones WHERE
  Patrons.CardNum = Phones.CardNum
```

- Sometimes `ON` syntax is necessary (later)

Sidetrack: newlines

- Command is not complete until semicolon
- Use whitespace to break up long queries



```
mysql> select * from phones join patrons
    -> on phones.cardnum = patrons.cardnum;
+---------+----------+------+---------+
| cardnum | phone    | name | cardnum |
+---------+----------+------+---------+
|       1 | 555-5555 | Joe  |       1 |
|       2 | 666-6666 | Ann  |       2 |
|       3 | 777-7777 | Ben  |       3 |
|       4 | 888-8888 | Dan  |       4 |
|       4 | 999-9999 | Dan  |       4 |
+---------+----------+------+---------+
```

NATURAL JOIN

- Joins on the columns two tables have in common

- `SELECT * FROM Phones NATURAL JOIN Patrons;`

↓ Phones

CardNum	Phone
1	555-5555
2	666-6666
3	777-7777
4	888-8888
4	999-9999

Patrons ✓

Name	CardNum
Joe	1
Ann	2
Ben	3
Dan	4

NATURAL JOIN

- SQL
  - `… NATURAL JOIN …`
- Relational model:
  - natural join

Quiz

- Query for getting cardnum, phone number(s), and name for “Dan”?

- Reminder:

```
select
from
join
where
```

Patrons

Name	CardNum
Joe	1
Ann	2
Ben	3
Dan	4

Phones

CardNum	Phone
1	555-5555
2	666-6666
3	777-7777
4	888-8888
4	999-9999

Quiz

- Get Titles of first 3 books in inventory (by serial)

```
select
from
join
where
```

Inventory

Serial	ISBN
1001	978-0590353427
1002	978-0590353427
1003	978-0679732242
1004	978-0394823379
1005	978-0394823379
1006	978-0062278791

Titles

ISBN	Title	Author
978-0590353427	Harry Potter	Rowling
978-0679732242	The Sound and the Fury	Faulkner
978-0394823379	The Lorax	Seuss
978-0062278791	Profiles in Courage	Kennedy
978-0441172719	Dune	Herbert

Quiz

- Get Titles of first 3 books in inventory (by Title)

```
select
from
join
where
```

Inventory

Serial	ISBN
1001	978-0590353427
1002	978-0590353427
1003	978-0679732242
1004	978-0394823379
1005	978-0394823379
1006	978-0062278791

Titles

ISBN	Title	Author
978-0590353427	Harry Potter	Rowling
978-0679732242	The Sound and the Fury	Faulkner
978-0394823379	The Lorax	Seuss
978-0062278791	Profiles in Courage	Kennedy
978-0441172719	Dune	Herbert

Warning

- This is called query by instance, and it is bad
  - `WHERE Serial <= 1003`
- What if the Serials don’t start at 1001?
- What if there are gaps in the Serials?

Inventory

Serial	ISBN
1001	978-0590353427
1002	978-0590353427
1003	978-0679732242
1004	978-0394823379
1005	978-0394823379
1006	978-0062278791

Better

```
… ORDER BY Serial LIMIT 3
```

Inventory

Serial	ISBN
1001	978-0590353427
1002	978-0590353427
1003	978-0679732242
1004	978-0394823379
1005	978-0394823379
1006	978-0062278791

Multiple Join

- Join operates on any two relations

$$\underbrace{\texttt{Patrons JOIN CheckedOut}}$$

- Result of a join is itself a table that can be joined

```
(Patrons JOIN CheckedOut) JOIN Inventory
```

Multiple Join

- Join operates on any two relations

$$\underbrace{\texttt{Patrons JOIN CheckedOut}}$$

- Result of a join is itself a table that can be joined

```
(Patrons JOIN CheckedOut) JOIN Inventory
```

- Parentheses not needed

Exercise

Patrons

Name	CardNum
Joe	1
Ann	2
Ben	3
Dan	4

Inventory

Serial	ISBN
1001	978-0590353427
1002	978-0590353427
1003	978-0679732242
1004	978-0394823379
1005	978-0394823379

CheckedOut

CardNum	Serial
1	1001
1	1004
4	1005
4	1004

Phones

CardNum	Phone
1	555-5555
2	666-6666
4	999-9999

Query to get **titles and authors** of books checked out by Joe?

Titles

ISBN	Title	Author
978-0590353427	Harry Potter	Rowling
978-0679732242	The Sound and the Fury	Faulkner
978-0394823379	The Lorax	Seuss
978-0062278791	Profiles in Courage	Kennedy
978-0441172719	Dune	Herbert

Exercise

Patrons

Name	CardNum
Joe	1
Ann	2
Ben	3
Dan	4

Inventory

Serial	ISBN
1001	978-0590353427
1002	978-0590353427
1003	978-0679732242
1004	978-0394823379
1005	978-0394823379

CheckedOut

CardNum	Serial
1	1001
1	1004
4	1005
4	1004

Phones

CardNum	Phone
1	555-5555
2	666-6666
4	999-9999

Query to get **phone number** of person holding Harry Potter?

Titles

ISBN	Title	Author
978-0590353427	Harry Potter	Rowling
978-0679732242	The Sound and the Fury	Faulkner
978-0394823379	The Lorax	Seuss
978-0062278791	Profiles in Courage	Kennedy
978-0441172719	Dune	Herbert