Entity Relationship Diagrams

DB Practice CSC 330 Spring 2022

README

- This is a notebook for learning and practicing modeling relational databases using Entity Relationship Diagrams, keys and bridge tables
- Though we won't get more deeply into database design, understanding relationships, you need to understand this to understand `JOIN` commands when querying relational databases
- The notebook also contains two practice exercises for you to complete on your own in a self-guided lab session.
- **Modeling** is an essential IT design skill. You can model:
  - Technical processes (e.g. with UML)
  - Human processes (e.g. with BPMN)
  - Algorithms (e.g. with pseudocode or flowcharts)
  - Entity relationships (e.g. with ERDs)

ERD notations

- Relational database design relies on table relationships
- This is especially important for `JOIN` operations
- There are two dominant notations for ERD[1]

Chen

- **Notation Example**

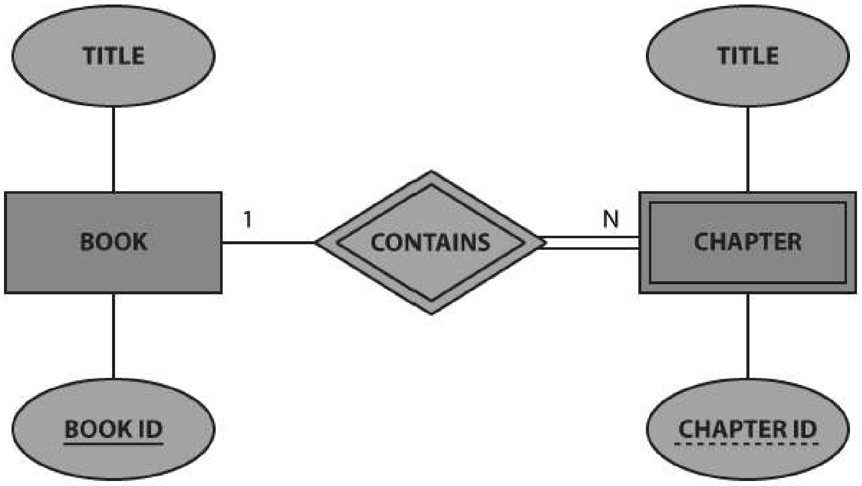


Figure 1: ERD Chen notation example (Source: Dybka, 2014)

- `BOOK` and `CHAPTER` are two entities (tables)
- A `BOOK` fully `CONTAINS` 1…N `CHAPTER` entities
- The `CHAPTER` is a fully dependent child of `BOOK`
- The `CHAPTER` totally participates in the `BOOK`
- The `BOOK` has attributes `TITLE` and `BOOK_ID`
- `BOOK_ID` is a *Primary Key* (`PK`) of `BOOK`
- The `CHAPTER` has attributes `TITLE` and `CHAPTER_ID`
- The `CHAPTER_ID` is a *Primary Key* (`PK`) of `CHAPTER`

Crow's foot--

- **Notation summary**
  - A box represents an **entity**, e.g. `book`
  - An entity has **attributes**, e.g. `book_id`, `book_title` etc.

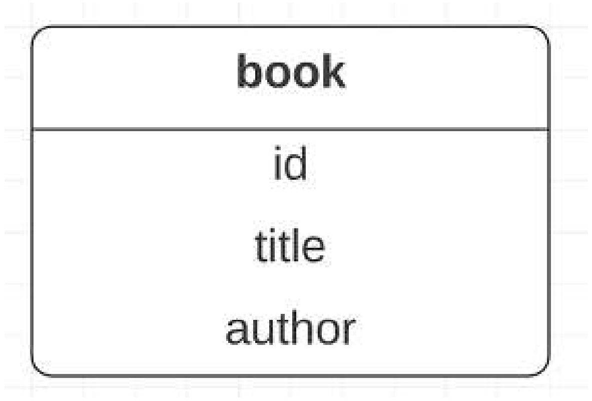


Figure 2: Entity with attributes

  - The attributes can be overloaded with additional properties like *Primary Key*, *Foreign Key*, and they have types like *integer*, *text* etc.

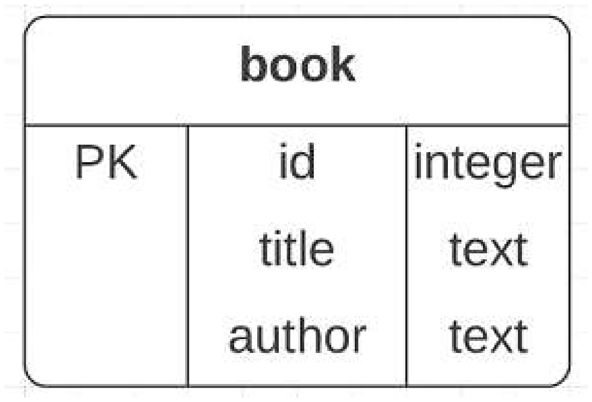


Figure 3: Entity with overloaded attributes

  - Lines between entities represent a (binary) **relationship**

- Relationships have two indicators: **maximum** (aka multiplicity) and **minimum**
- Multiplicity = *maximum* number of associations between the entities

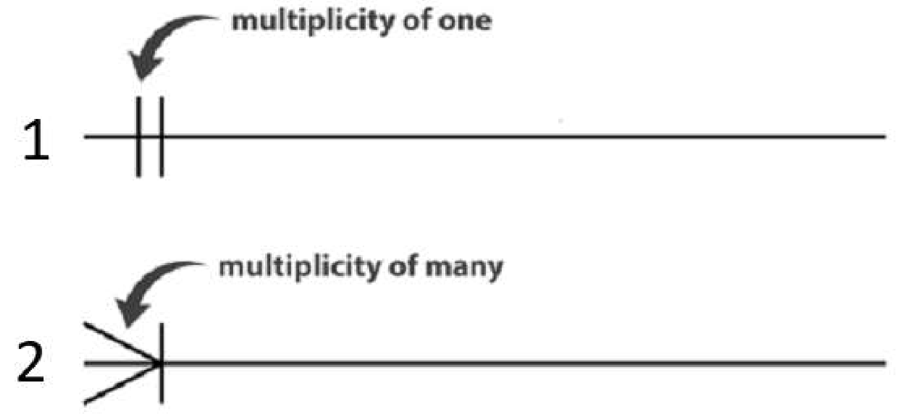


Figure 4: Multiplicity (Source: Dybka, 2016)

- Example 1: "A book has one and only one ISBN."
- Example 2: "A book has many chapters."

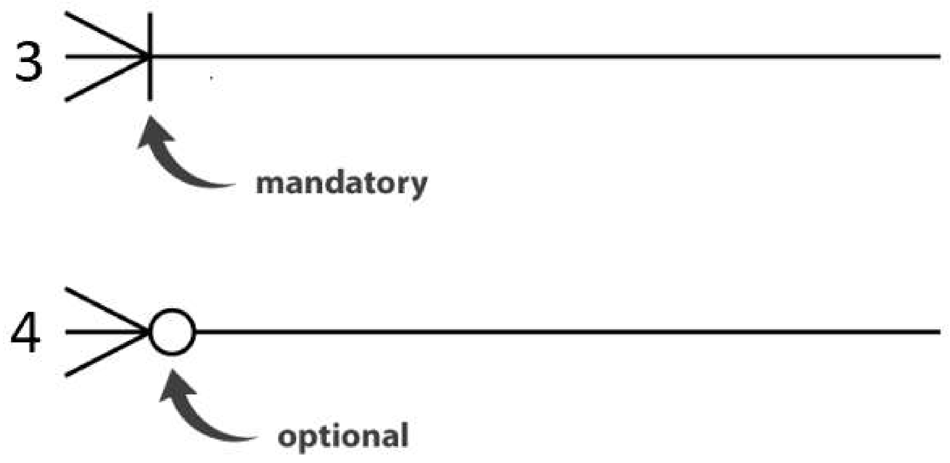


Figure 5: Minimum (Source: Dybka, 2016)

- Example 3: "A book has at least one chapter."
- Example 4: "A book has no or many new editions."

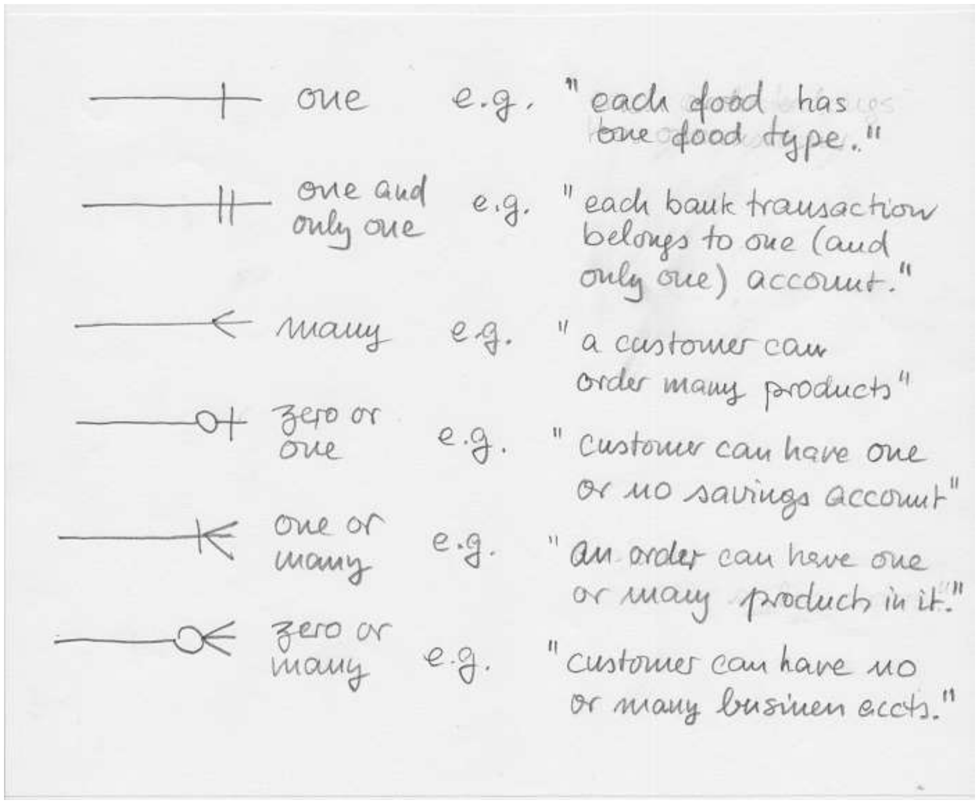

one e.g. "each food has one food type."
one and only one e.g. "each bank transaction belongs to one (and only one) account."
many e.g. "a customer can order many products"
zero or one e.g. "Customer can have one or no savings account"
one or many e.g. "an order can have one or many products in it."
zero or many e.g. "customer can have no or many business accts."

Figure 6: ERD crow's foot notation summary

- Realistic example: bank tables

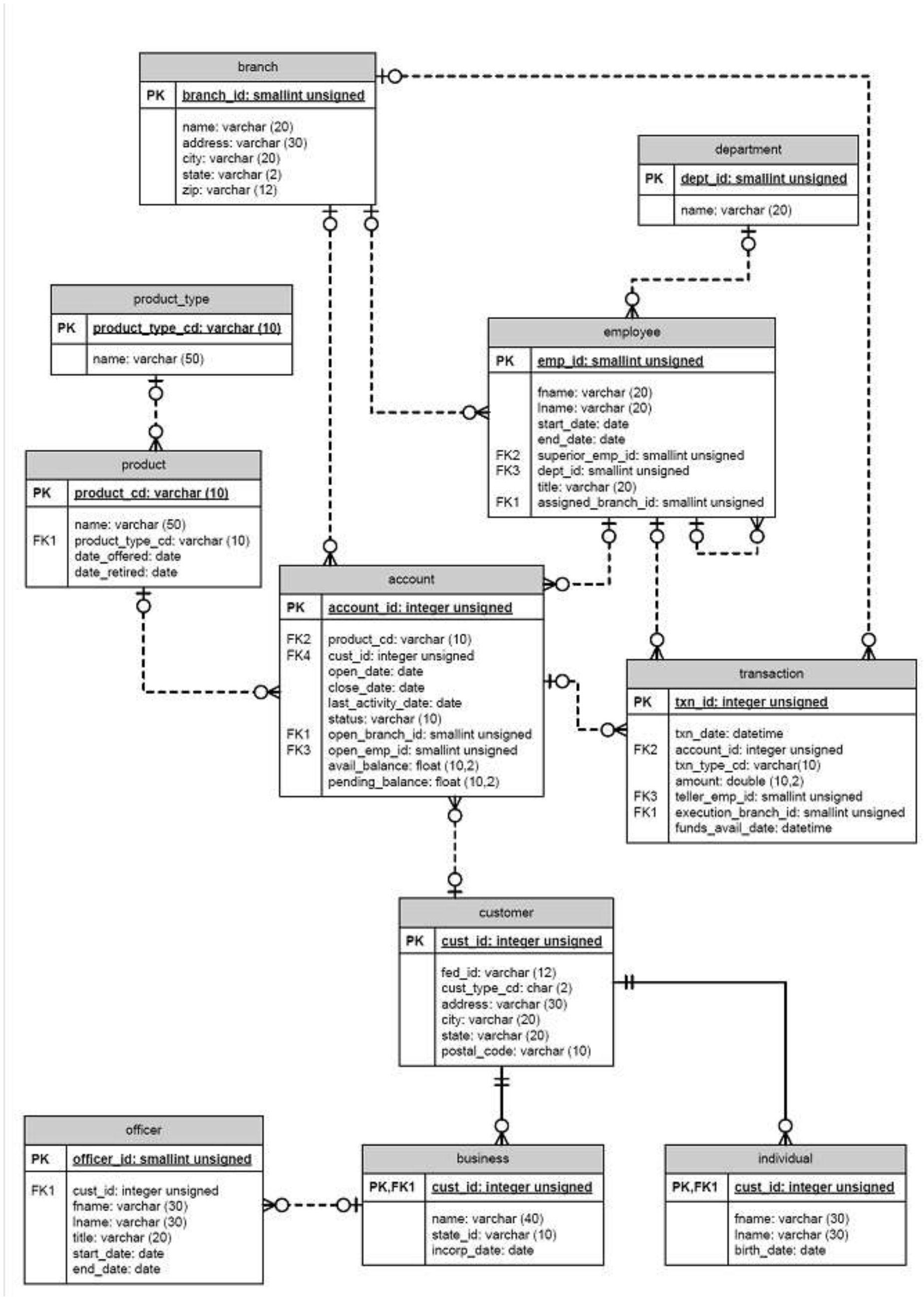

branch
PK branch_id: smallint unsigned
name: varchar (20)
address: varchar (30)
city: varchar (20)
state: varchar (2)
zip: varchar (12)
department
PK dept_id: smallint unsigned
name: varchar (20)
product_type
PK product_type_cd: varchar (10)
name: varchar (50)
employee
PK emp_id: smallint unsigned
fname: varchar (20)
lname: varchar (20)
start_date: date
end_date: date
FK2 superior_emp_id: smallint unsigned
FK3 dept_id: smallint unsigned
title: varchar (20)
FK1 assigned_branch_id: smallint unsigned
product
PK product_cd: varchar (10)
name: varchar (50)
FK1 product_type_cd: varchar (10)
date_offered: date
date_retired: date
account
PK account_id: integer unsigned
FK2 product_cd: varchar (10)
FK4 cust_id: integer unsigned
open_date: date
close_date: date
last_activity_date: date
status: varchar (10)
FK1 open_branch_id: smallint unsigned
FK3 open_emp_id: smallint unsigned
avail_balance: float (10,2)
pending_balance: float (10,2)
transaction
PK txn_id: integer unsigned
txn_date: datetime
FK2 account_id: integer unsigned
txn_type_cd: varchar(10)
amount: double (10,2)
FK3 teller_emp_id: smallint unsigned
FK1 execution_branch_id: smallint unsigned
funds_avail_date: datetime
customer
PK cust_id: integer unsigned
fed_id: varchar (12)
cust_type_cd: char (2)
address: varchar (30)
city: varchar (20)
state: varchar (20)
postal_code: varchar (10)
officer
PK officer_id: smallint unsigned
FK1 cust_id: integer unsigned
fname: varchar (30)
lname: varchar (30)
title: varchar (20)
start_date: date
end_date: date
business
PK,FK1 cust_id: integer unsigned
name: varchar (40)
state_id: varchar (10)
incorp_date: date
individual
PK,FK1 cust_id: integer unsigned
fname: varchar (30)
lname: varchar (30)
birth_date: date

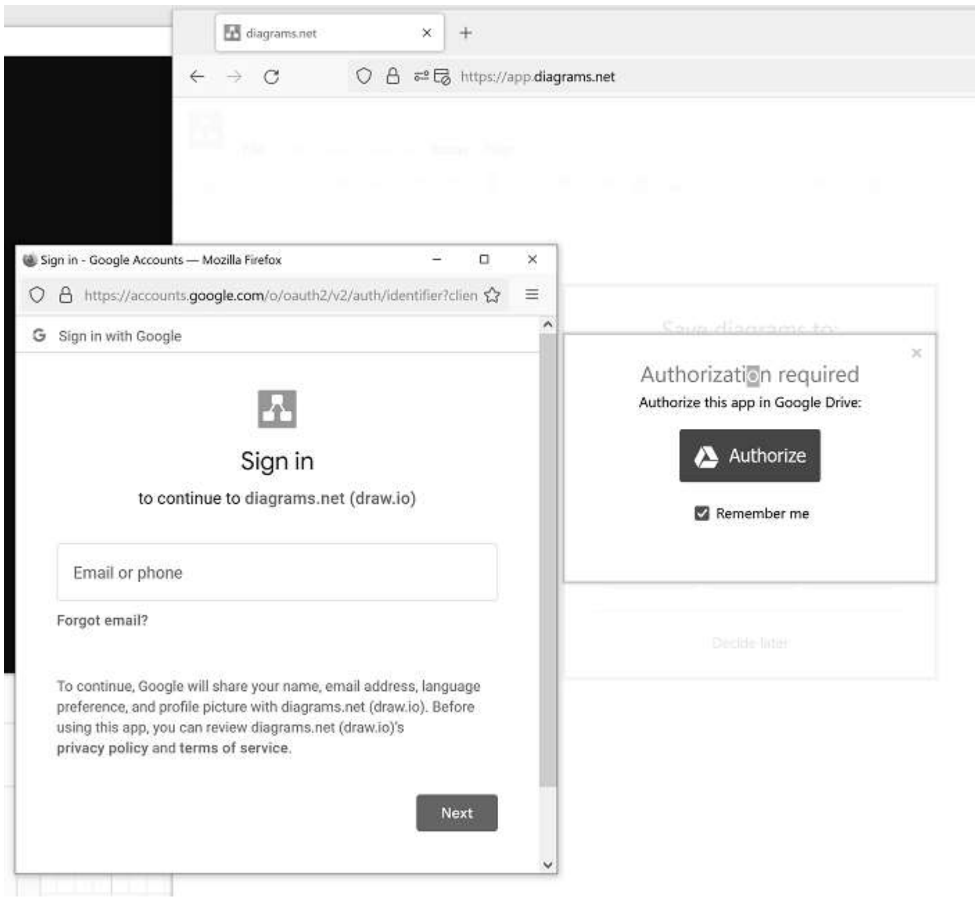
Figure 7: ERD of a bank (Source: Beaulieu, 2008)

Food database revisited

Create diagram with `draw.io`

- [ ] Open draw.io - you can choose where to save your diagrams
- [ ]

  If you choose Google Drive, you need to sign in to authorize as shown in the image.



to continue to diagrams.net (draw.io)

To continue, Google will share your name, email address, language preference, and profile picture with diagrams.net (draw.io). Before using this app, you can review diagrams.net (draw.io)'s privacy policy and terms of service.

Authorization required

Authorize this app in Google Drive:

Figure 8: draw.io authorization dialog

- [ ] When you've authorized the storage place, you can create a new diagram. Don't bother with the templates.
- [ ] This short video shows how to create an entity and relationships between them using the crow's foot notation.

Create diagram

- [ ] *Zoom to Width* using the menu (left bottom of screen)
- [ ] Highlight and delete the diagrams on the screen
- [ ] Drag a table template onto the drawing board
- [ ]

  Draw the 4 entity diagrams for the food database

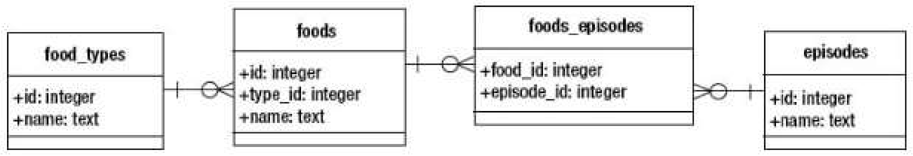


Figure 9: ERD of the food database (Source: Allen/Owens, 2010)

- `[ ]`

  Write down the relationships between `food_types` and `food`. Remember that you're not trying to describe all possible relationships but only the relationship for the purpose of this database with its narrow meaning: types of foods shown in episodes of the Seinfeld TV show.

  #+begin_quote

`food_types` to `food`: each food type has zero or many instances

of food in the show.

`food` to `food_types`: each food on the show is exactly one type

of food.

#+end_quote

Practice: customer orders

- `[X]` Develop an ER diagram with three tables: `customer`, `order`, and `product`
- `[X]` Identify suitable attributes
- `[X]` The diagram should allow for relationships like
  - "A customer submits an order"
  - "An order contains a product"
  - "An order belongs to a customer"
  - "A product is part of an order."
- `[X]` Use draw.io to draw the diagram
- `[X]` Draw relationships with the correct minimum/maximum
- `[ ]` Give your diagram a title
- `[ ]`

  Share the link to your solution with me via Email

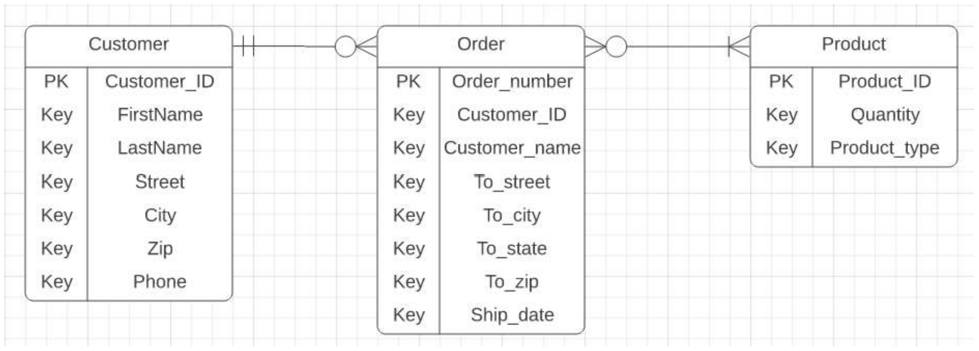


Figure 10: Customer orders products ERD

Practice exercise (due Thursday, 7 April 4 pm)

- `[ ]` Develop an ER diagram with relationships between **four** tables: `Student`, `Course`, `Lecturer`, and `Class` (or meeting).
- `[ ]` Identify suitable attributes: make sure that they are
  - measurable (you can think of a suitable data type)
  - atomic (`address` is composite, `street`, `street number` are atomic)
  - attributes (like `name`) and not entities (like `registrar`) or relationships (like `attendance`) themselves
- `[ ]` The diagram should allow for relationships like
  - "A student is enroled in a course."
  - "A class is attended by students."
  - "An lecturer offers a course."
  - "A class is taught by an lecturer."
  - etc.
- `[ ]` Use draw.io to draw the final ERD (see demo video)
- `[ ]` Draw relationships with the correct minimum/maximum
- `[ ]` Put `ERD assignment by [Your name] - Pledged` in the title of your diagram
- `[ ]` Upload a screenshot of your solution to Schoology

Sample solution

- All relationships are subject to **business rules**. E.g. at least one student may have to be enroled in a course, or a lecturer may only be allowed to give a certain number of lectures - or reversely, a lecture may have to have at least one lecturer assigned to it.

- Because business rules change, the relationship structure of the database may also change. The schema, reflected in the entity relationship diagram, reflects this (source diagram at ponyorm).

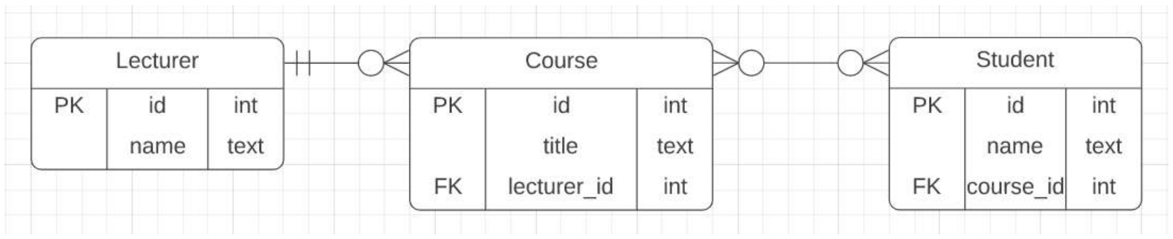


Figure 11: Sample solution (part) for students db

- Important to note:
  - Each lecturer can teach many courses (or none)
  - Each course can only be taught by exactly one lecturer
  - Each course can have many students (or none)
  - Each student can have many courses (or none)

Keys and bridge tables

ERD and relational model

- The ER model maps the relational database model:

ER MODEL (ORM)	DATABASE	EXAMPLE
Entity	Table	Lecturer

ER MODEL (ORM)	DATABASE	EXAMPLE
Entity attribute	Tuple/record/row	("2", "Birkenkrahe")
Attribute	Column	Lecturer.name
Attribute type	Column data type	integer/int

PK Definition

- A **primary key** (`PK`) is an attribute (or column) that uniquely identifies every record in a certain table.
- We already marked the potential PKs in the figure 10.
- Primary key rules:
    1. **UNIQUE** (across the database, i.e. all tables)
    2. **unchangeable** (while the table exists)
    3. **NOT NULL** (when data are inserted)

Key candidate identification

- In any table, the tuple of potential primary keys form the **candidate key**.

  Example table:

id	fname	lname	street	city	zip	phone
30014	John	Smith	1014 Main St	Batesville	72501	870-307-4245
30067	John	Smith	2300 College Rd	Batesville	72501	501-444-4287
30333	Jane	Doe	1014 Main St	Conway	72004	877-223-4445

- Names, phones, addresses are not *unique* or *unchangeable*
- Only `id` is designed to fulfil the PK rules
- Can be "randomly" assigned (are there problems with that?)[2]

Example: web sites

- If a website does not let you change your username, it likely uses your `username` as a primary key (*unchangeable*)
- If you're told that a username is already taken, it likely uses your `username` as a primary key (*unique*)
- A website will force you to enter certain information for database reasons if the information is used to create your PK (must be `NOT NULL`)

Foreign Keys

- Foreign keys are primary keys in other (linked) tables
- Example from the customer example:

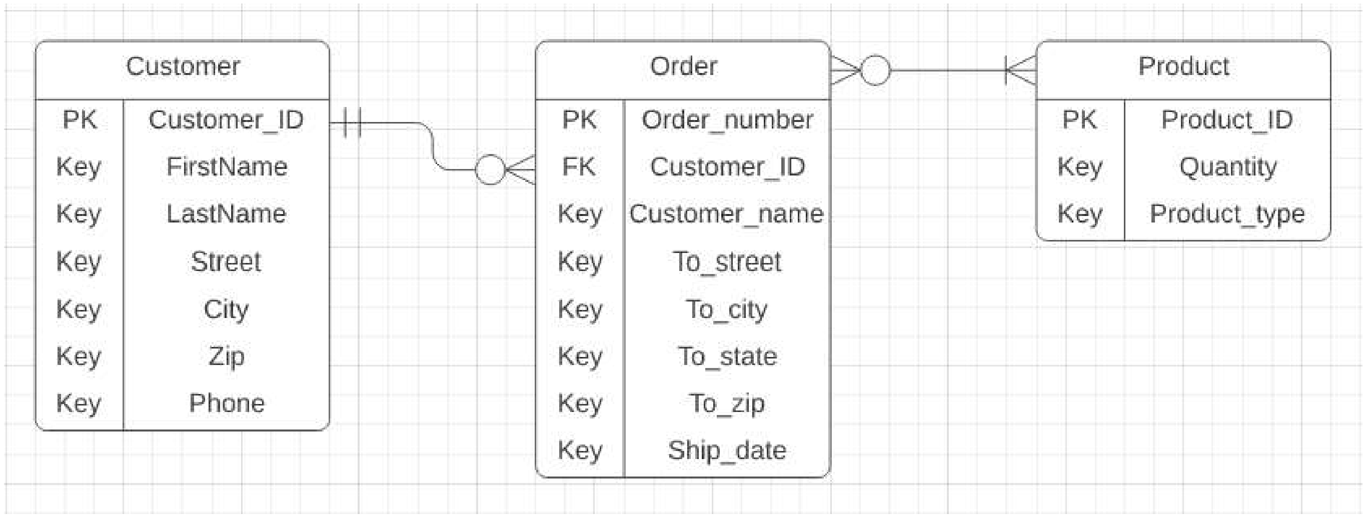
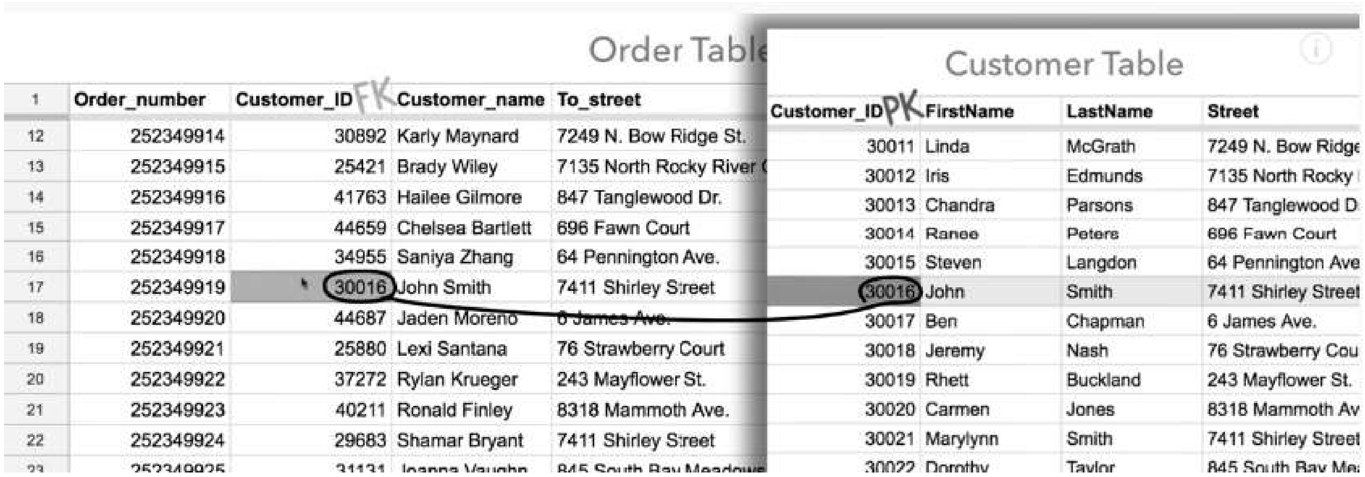


Figure 12: PK and FK in Customer and Order (ERD)


Order Table

1	Order_number	Customer_ID FK	Customer_name	To_street
12	252349914	30892	Karly Maynard	7249 N. Bow Ridge St.
13	252349915	25421	Brady Wiley	7135 North Rocky River
14	252349916	41763	Hailee Gilmore	847 Tanglewood Dr.
15	252349917	44659	Chelsea Bartlett	696 Fawn Court
16	252349918	34955	Saniya Zhang	64 Pennington Ave.
17	252349919	30016	John Smith	7411 Shirley Street
18	252349920	44687	Jaden Moreno	6 James Ave.
19	252349921	25880	Lexi Santana	76 Strawberry Court
20	252349922	37272	Rylan Krueger	243 Mayflower St.
21	252349923	40211	Ronald Finley	8318 Mammoth Ave.
22	252349924	29683	Shamar Bryant	7411 Shirley Street
23	252349925	31131	Joanna Vaughn	845 South Bay Meadows

Customer Table

Customer_ID PK	FirstName	LastName	Street
30011	Linda	McGrath	7249 N. Bow Ridge
30012	Iris	Edmunds	7135 North Rocky
30013	Chandra	Parsons	847 Tanglewood D
30014	Ranee	Peters	696 Fawn Court
30015	Steven	Langdon	64 Pennington Ave
30016	John	Smith	7411 Shirley Street
30017	Ben	Chapman	6 James Ave.
30018	Jeremy	Nash	76 Strawberry Cou
30019	Rhett	Buckland	243 Mayflower St.
30020	Carmen	Jones	8318 Mammoth Av
30021	Marylynn	Smith	7411 Shirley Street
30022	Dorothy	Taylor	845 South Bay Me

Figure 13: PK and FK in Customer and Order (tables)

- Foreign keys do not need to be unique (a customer could make another order) and there can be multiple foreign keys in one entity - if we want to link information across multiple tables.
- Example: `product_id` in a customer order. Now the order table has two foreign keys, `customer_id` and `product_id`.

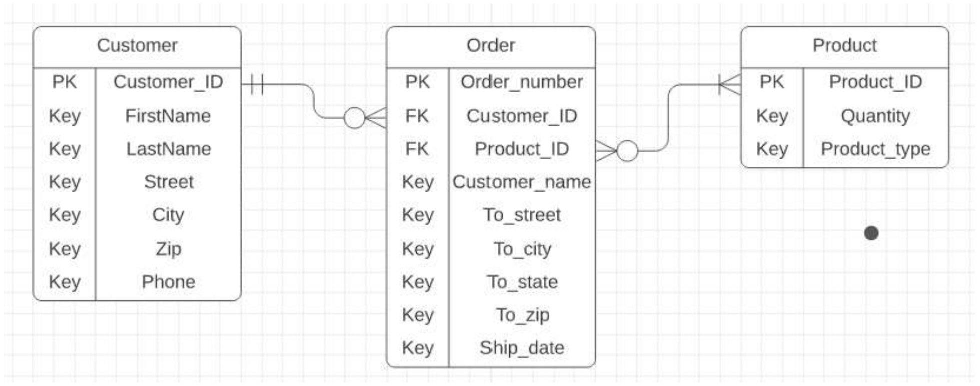
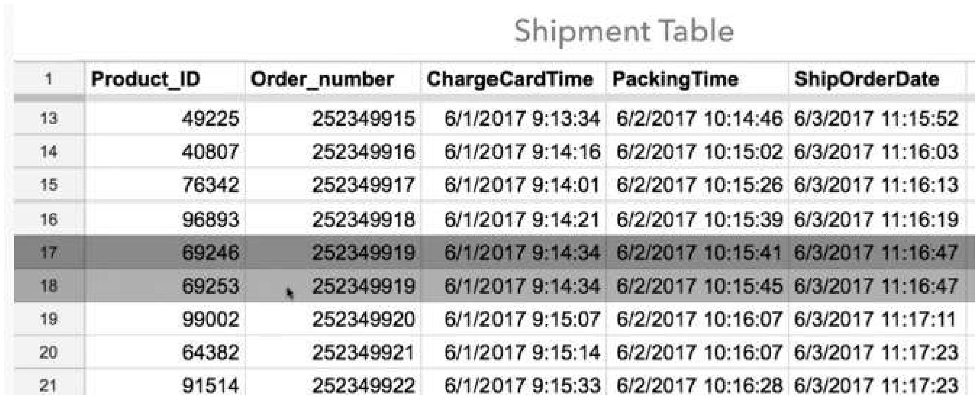


Figure 14: PK and FK in Order and Product (ERD)

Composite primary key

- Let's say, an order is sent in two different shipments.



Shipment Table

1	Product_ID	Order_number	ChargeCardTime	PackingTime	ShipOrderDate
13	49225	252349915	6/1/2017 9:13:34	6/2/2017 10:14:46	6/3/2017 11:15:52
14	40807	252349916	6/1/2017 9:14:16	6/2/2017 10:15:02	6/3/2017 11:16:03
15	76342	252349917	6/1/2017 9:14:01	6/2/2017 10:15:26	6/3/2017 11:16:13
16	96893	252349918	6/1/2017 9:14:21	6/2/2017 10:15:39	6/3/2017 11:16:19
17	69246	252349919	6/1/2017 9:14:34	6/2/2017 10:15:41	6/3/2017 11:16:47
18	69253	252349919	6/1/2017 9:14:34	6/2/2017 10:15:45	6/3/2017 11:16:47
19	99002	252349920	6/1/2017 9:15:07	6/2/2017 10:16:07	6/3/2017 11:17:11
20	64382	252349921	6/1/2017 9:15:14	6/2/2017 10:16:07	6/3/2017 11:17:23
21	91514	252349922	6/1/2017 9:15:33	6/2/2017 10:16:28	6/3/2017 11:17:23

Figure 15: Two orders in the shipment table

- We need a composite primary key, because none of the individual IDs will satisfy our rules - none are unique:
  - Someone else might order the same product (`Product_ID`)
  - Products might be in the same order (`Order_ID`)
  - Time and date could coincide
- The pair `(Product_ID, Order_ID)` is unique for the shipment - it's a valid **composite primary key**.
- Rules:
  - use the fewest number of attributes possible
  - attributes should be unchangeable
- Alternative: add a `Shipping_ID`.

Bridge tables

- If two tables (like `Student` and `Course` in figure 11) are connected by man-to-many relationships, you need a bridge table to remove ambiguities.
- Otherwise, many details are not accessible: e.g. without the Order table, you would not know how many products a customer bought, or when he made individual purchases.

- With the order table, each time a product is purchased, there is a record in the order table about when and how many products.
- The easiest way to generate a bridge table is by creating an intermediate table with only two columns that together form the composite key.
- For example, for the `student-course` relationship, this could be a table `enrolment` with the primary key `(student_id, course_id)`. Such a table could record all kinds of enrolment data.

Object relations mapper

- Ponyorm is a (free) example, and Lucidchart and other apps also offer this option - translate an ERD model into a database.
- The other way around is also possible and common: visualizing an existing database. Recommended for SQLite: DBeaver (open source).

References

- Birkenkrahe (April 5, 2022). Drawing ERD in draw.io [video]. URL: youtu.be/gCranxLqZDI.
- Dybka (August 2, 2014). Chen Notation [blog]. URL: vertabelo.com.
- Dybka (August 31, 2016). Crow's Foot Notation [blog]. URL: vertabelo.com.
- Lucidchart (2017). Entity Relationship Diagram (ERD) Tutorial Part 1 [video]. URL: youtu.be/QpdhBUYk7Kk.

Footnotes:

[1] Notation reflects priorities, e.g. readibility vs. detail. It is surprising that there aren't more popular notations! Notation must faithfully represent the modeling standard. You can in fact become famous with notation - Feynman diagrams are an example: they are a diagrammatic language for complicated integrals that represent elementary particle interactions.

[2] Yes - (1) computers cannot generate true random numbers; (2) the basis for the number may over time get exhausted - this happened e.g. with IPv4 addresses (see "IPv4 address exhaustion").

Author: Marcus Birkenkrahe

Created: 2022-04-12 Tue 12:06