Data Models and Databases 101

Berkeley Copwatch / WITNESS workshop
28 April 2018

Objectives

- Become familiar with what databases are, why they can be useful, and the steps to designing an effective database.
- Come to agreed understanding of purpose of next phase of BC database, what we want it to enable us to do.
- Collaboratively design new data model, schema, and data dictionary, building upon work that has already been done.

What is a database?

Datum or **data element**: a piece of factual information.

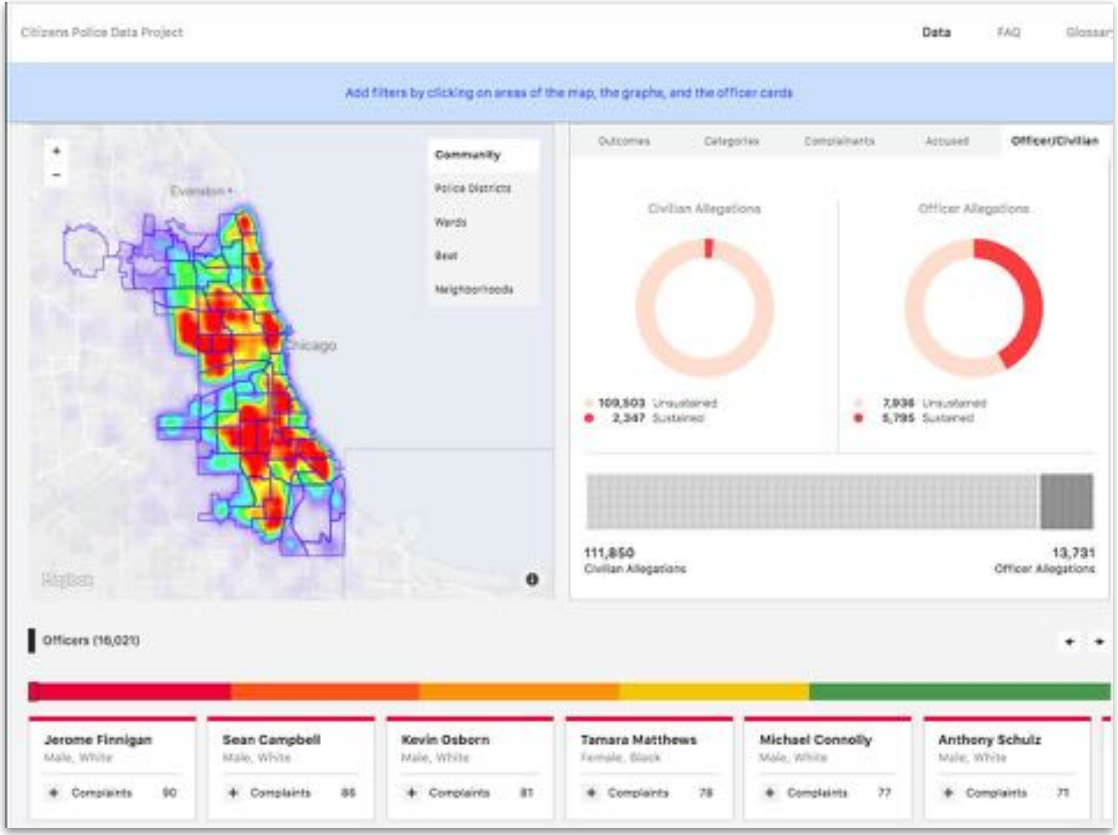
Database: a collection of related data that represents some aspect of the real world, and that is designed and populated to serve a specific purpose.

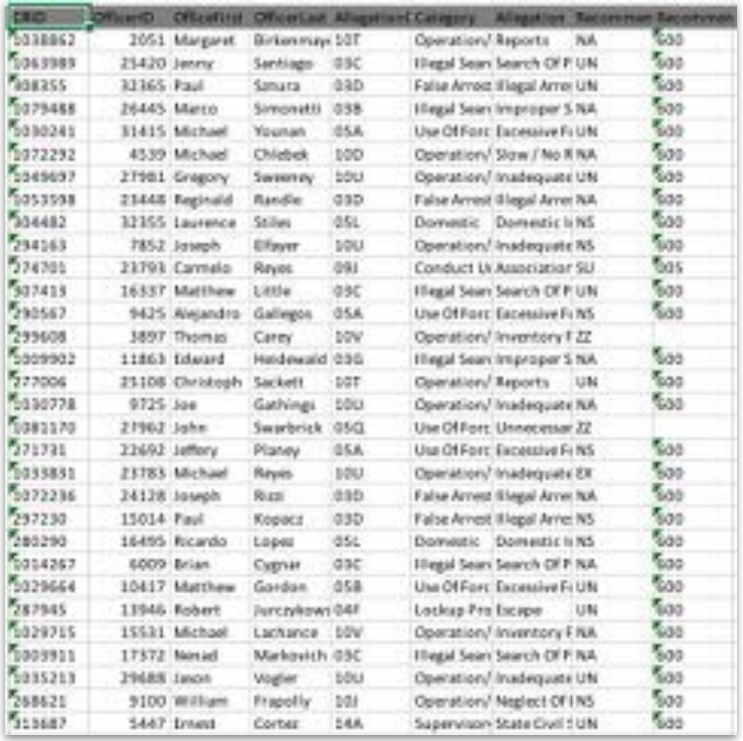
Typically implemented using **database management software** that allows users to retrieve selected/filtered data, sort, count or calculate data.

How can a database be useful to you?

- An accessible and centralized place to store information you want to hold on to.
- Structured data can harness computing power to sort, match, link, calculate, aggregate information, and see it in different ways.
- Structured data is easier to share and invite collaboration on.
- Enabling strategic use and exchange of information.

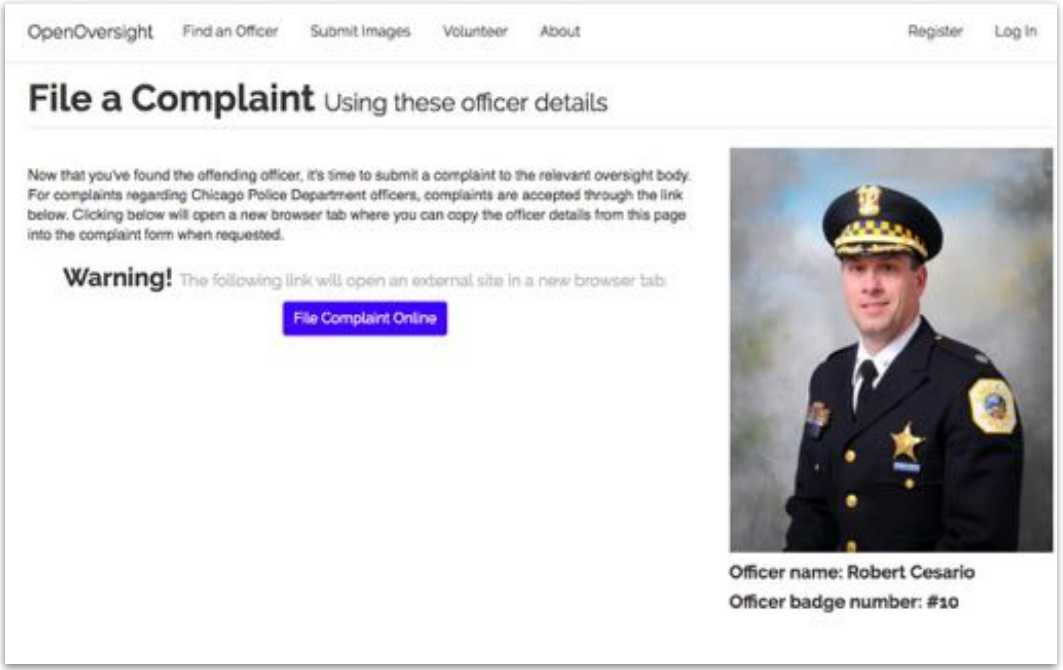
Some examples of databases





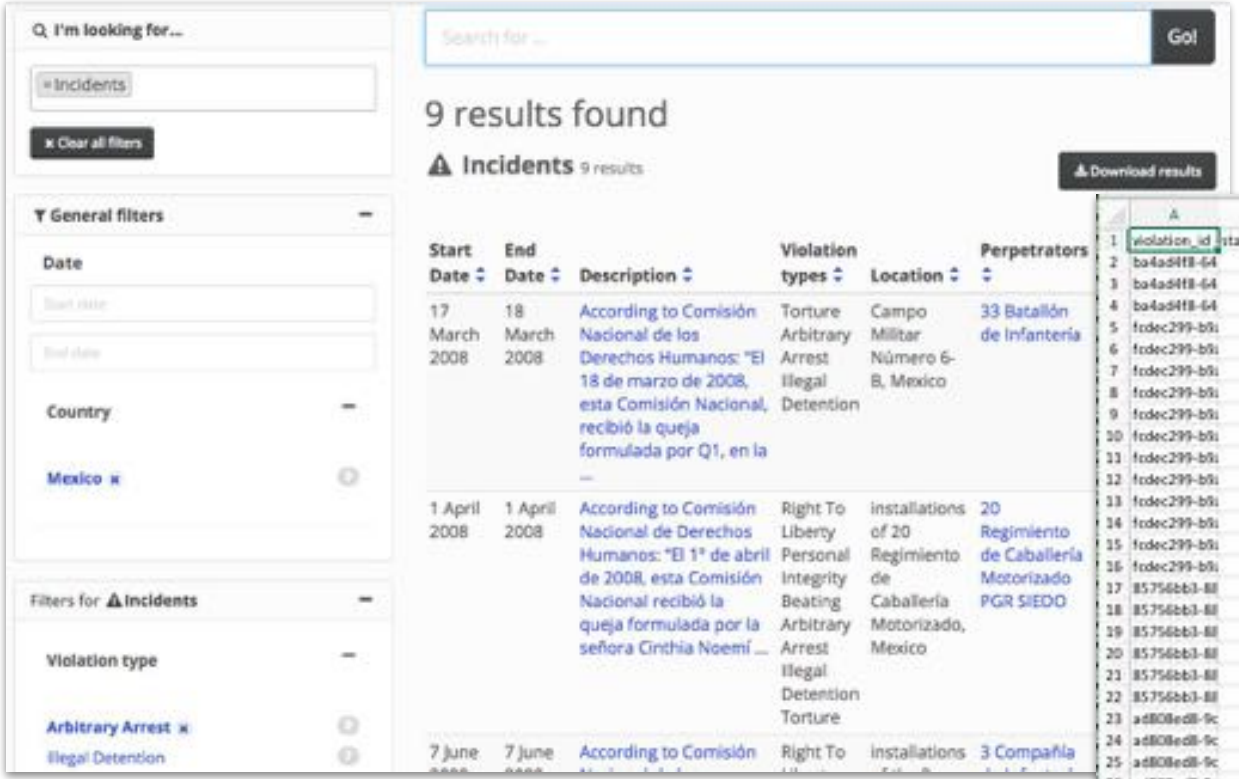
CRID	OfficerID	OfficerFirst	OfficerLast	Allegation	Category	Allegation	Recommen	Recommen
1038862	2051	Margaret	Birkenmaye	10T	Operation/	Reports	NA	00
1063989	25420	Jenny	Santiago	09C	Illegal Sear	Search Of P	UN	00
1038355	32365	Paul	Sanura	03D	False Arrest	Illegal Arre	UN	00
1079488	26445	Marco	Simonetti	03B	Illegal Sear	Improper S	NA	00
1030241	31415	Michael	Younan	05A	Use Of Forc	Excessive Fi	UN	00
1072292	4539	Michael	Chlebek	10D	Operation/	Slow / No R	NA	00
1049697	27981	Gregory	Sweeney	10U	Operation/	Inadequate	UN	00
1053598	23448	Reginald	Randle	03D	False Arrest	Illegal Arre	NA	00
1034482	32155	Laurence	Stiles	05L	Domestic	Domestic In	NS	00
1094163	7852	Joseph	Effayer	10U	Operation/	Inadequate	NS	00
1074701	21791	Carmelo	Reyes	09J	Conduct Un	Association	SU	00
1037419	16537	Matthew	Little	09C	Illegal Sear	Search Of P	UN	00
1090567	9425	Alejandro	Gallegos	05A	Use Of Forc	Excessive Fi	NS	00
1093608	3897	Thomas	Carey	10V	Operation/	Inventory F	ZZ	
1009902	11863	Edward	Heidewald	03G	Illegal Sear	Improper S	NA	00
1077006	25108	Christoph	Sackett	10T	Operation/	Reports	UN	00
1030778	9725	Joe	Gathings	10U	Operation/	Inadequate	NA	00
1083170	27962	John	Swarbrick	05Q	Use Of Forc	Unnecessar	ZZ	
1071731	22692	Jeffery	Planey	05A	Use Of Forc	Excessive Fi	NS	00
1033851	23783	Michael	Reyes	10U	Operation/	Inadequate	ER	00
1072236	24128	Joseph	Rizzi	03D	False Arrest	Illegal Arre	NA	00
1037230	15014	Paul	Kopacz	03D	False Arrest	Illegal Arre	NS	00
1080290	16495	Ricardo	Lopez	05L	Domestic	Domestic In	NS	00
1014267	6009	Brian	Cygnar	09C	Illegal Sear	Search Of P	NA	00
1029664	10417	Matthew	Gordon	05B	Use Of Forc	Excessive Fi	UN	00
1087945	13946	Robert	Jurczykows	04F	Lockup Pro	Escape	UN	00
1029715	15531	Michael	Lachance	10V	Operation/	Inventory F	NA	00
1003911	17572	Nenad	Markovith	09C	Illegal Sear	Search Of P	NA	00
1035213	29688	Jason	Vogler	10U	Operation/	Inadequate	UN	00
1068621	9100	William	Frapolly	10J	Operation/	Neglect Of I	NS	00
1136887	5447	Ernest	Cortez	14A	Supervisor	State Civil I	UN	00

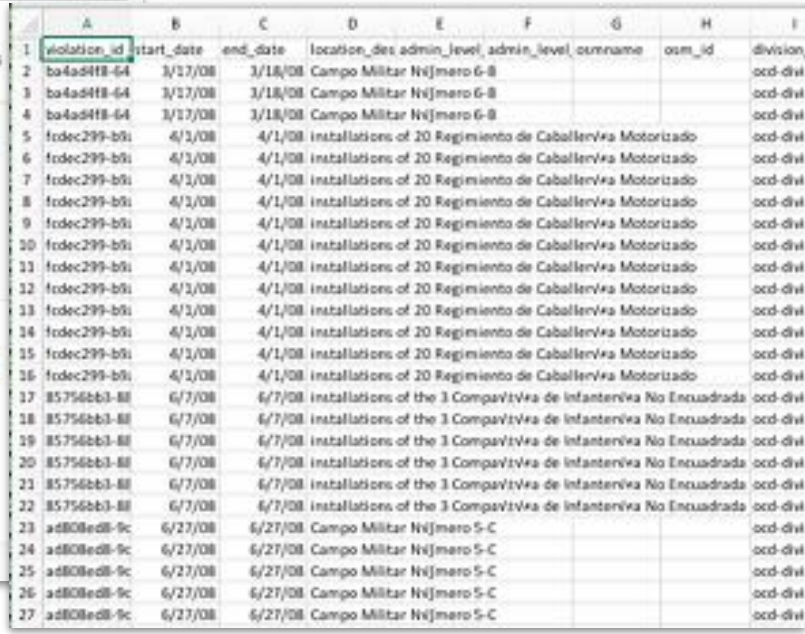
Invisible Institute, Citizen Police Data Project




```python
class Department(db.Model):
    __tablename__ = 'departments'
    id = db.Column(db.Integer, primary_key=True)
    name = db.Column(db.String(255), index=True, unique=True, nullable=False)
    short_name = db.Column(db.String(100), unique=False, nullable=False)

    def __repr__(self):
        return '<Department ID {}: {}>'.format(self.id, self.name)


class Officer(db.Model):
    __tablename__ = 'officers'

    id = db.Column(db.Integer, primary_key=True)
    last_name = db.Column(db.String(120), index=True, unique=False)
    first_name = db.Column(db.String(120), index=True, unique=False)
    middle_initial = db.Column(db.String(120), unique=False, nullable=True)
    race = db.Column(db.String(120), index=True, unique=False)
    gender = db.Column(db.String(120), index=True, unique=False)
    employment_date = db.Column(db.DateTime, index=True, unique=False, nullable=True)
    birth_year = db.Column(db.Integer, index=True, unique=False, nullable=True)
    assignments = db.relationship('Assignment', backref='officer', lazy='dynamic')
    face = db.relationship('Face', backref='officer', lazy='dynamic')
    department_id = db.Column(db.Integer, db.ForeignKey('departments.id'))
    department = db.relationship('Department', backref='officers')

    def __repr__(self):
        return '<Officer ID {}: {} {} {}>'.format(self.id,
                                                  self.first_name,
                                                  self.middle_initial,
                                                  self.last_name)
```

Lucy Parsons Lab, Open Oversight





Security Force Monitor, “Who was in Command” database

Form completed by: Your Phone Number

Berkeley COPWATCH

Case Interview Form (for 2nd hand accounts)

File Creation Date 4/12/2008

Document I.D. C04-076

Information on Complainant

First Name Last Name

Street Address

City State Zip

Phones

Relationship to Victim

Information on Victim

First Name Last Name

Street Address

City State Zip Age

Phone Gender

Race/Ethnicity Victim's sexual orientation

Immigration status Date of birth

Criminal history related to incident

Information on the Officers

badge#	officer f. name	officer l. name	race	gender	dept.	car #

Nature of Complaint

Police violence Harrassment Wrongful arrest Justice system complaint Policy co

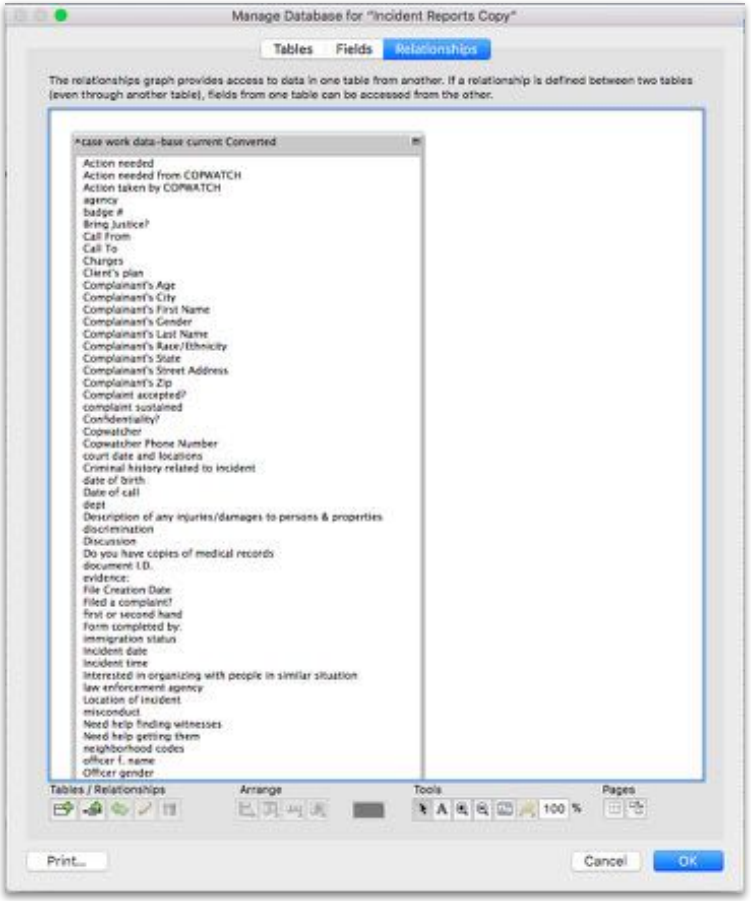
Other

Information on the incident

Location of incident

Incident date Incident time

What happened?



Berkeley Copwatch, internal database

Database vs. archive

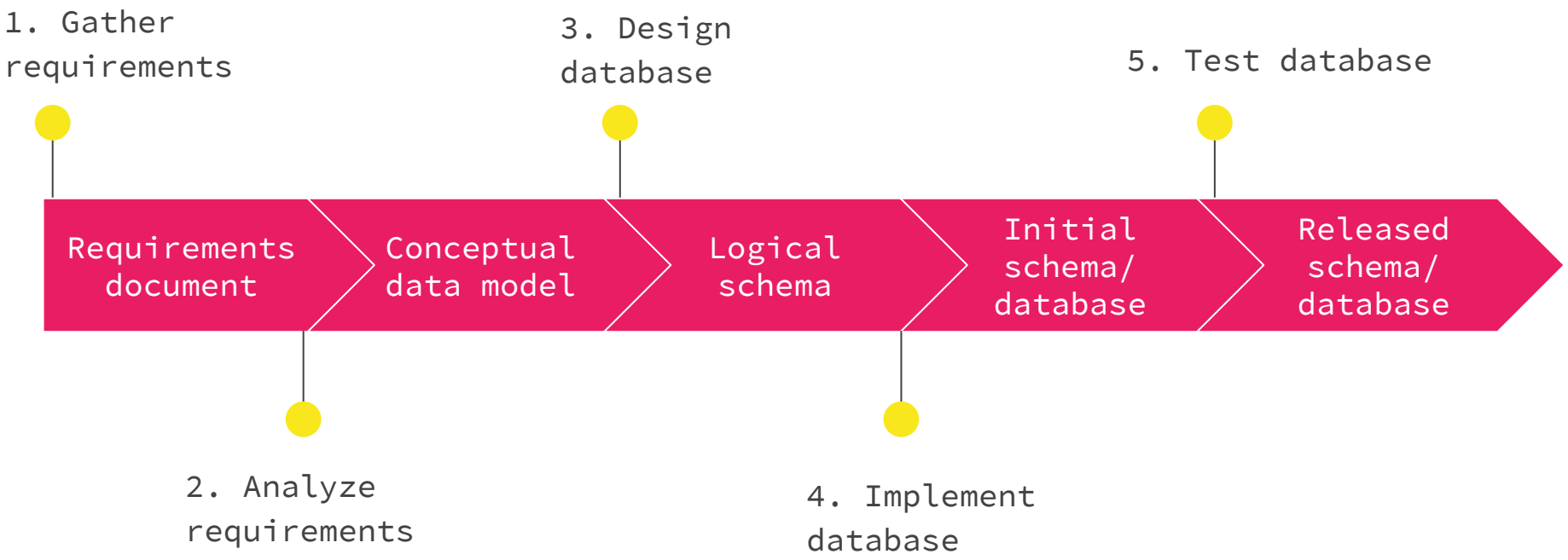
- **Archive:** An organization (systems & people) that preserves and provides access to information objects (e.g. files, videos, documents, artworks, etc).
- **Database:** Compiled and structured data that serves some purpose.
- Archives often have databases, to serve purpose of search and retrieval of objects. Archive databases contain a type of data called **“metadata”** - i.e. data that describes information object (e.g. data about a video, or data about a document).

Database vs. archive, Berkeley Copwatch

- BC has both a database and a video archive, that mostly function separately.
- Do we want them to be more integrated? The database can contain data about policing *and* metadata about videos.

  If so, let's make sure to include in data modeling, up next...

Database development process



1. Gathering requirements

Creating a common understanding of why, who we are creating this database for, what those users need to be able to do.

Databases provide responses to queries.

- **What questions do we want to be able to answer using the data?**

Example: “Who Was In Command” database

What is WhoWasInCommand?

WhoWasInCommand answers key questions about the structure, behaviour and people in charge of security forces like the police and army:

- Who is in charge of the specialized anti-riot police unit?
- What army unit has jurisdiction over what areas and for how long?
- Where did this commander previously serve, and where did they go next?
- When was a particular police unit based in a specific city?
- What allegations have civil society groups made against a unit or commander?

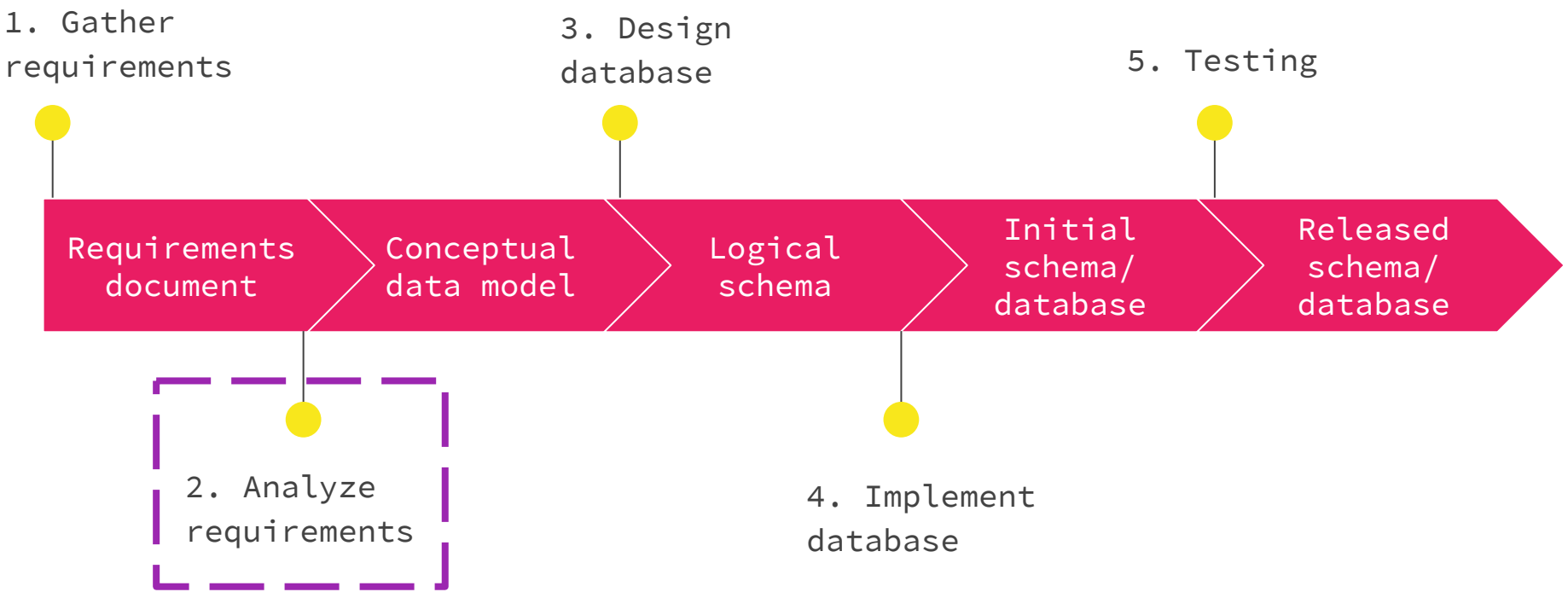
WhoWasInCommand presents data from thousands of public sources to help human rights researchers, investigative journalists and anyone who wants security forces to be more accountable.

Learn more →

Berkeley Copwatch requirements: discussion

- What should the purpose of this next phase of the Berkeley Copwatch database be?
- Review preliminary set of questions. Is this the information we want to know? Any additions / revisions?

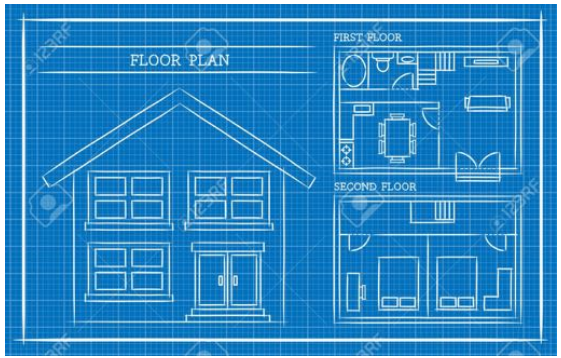
Database development process



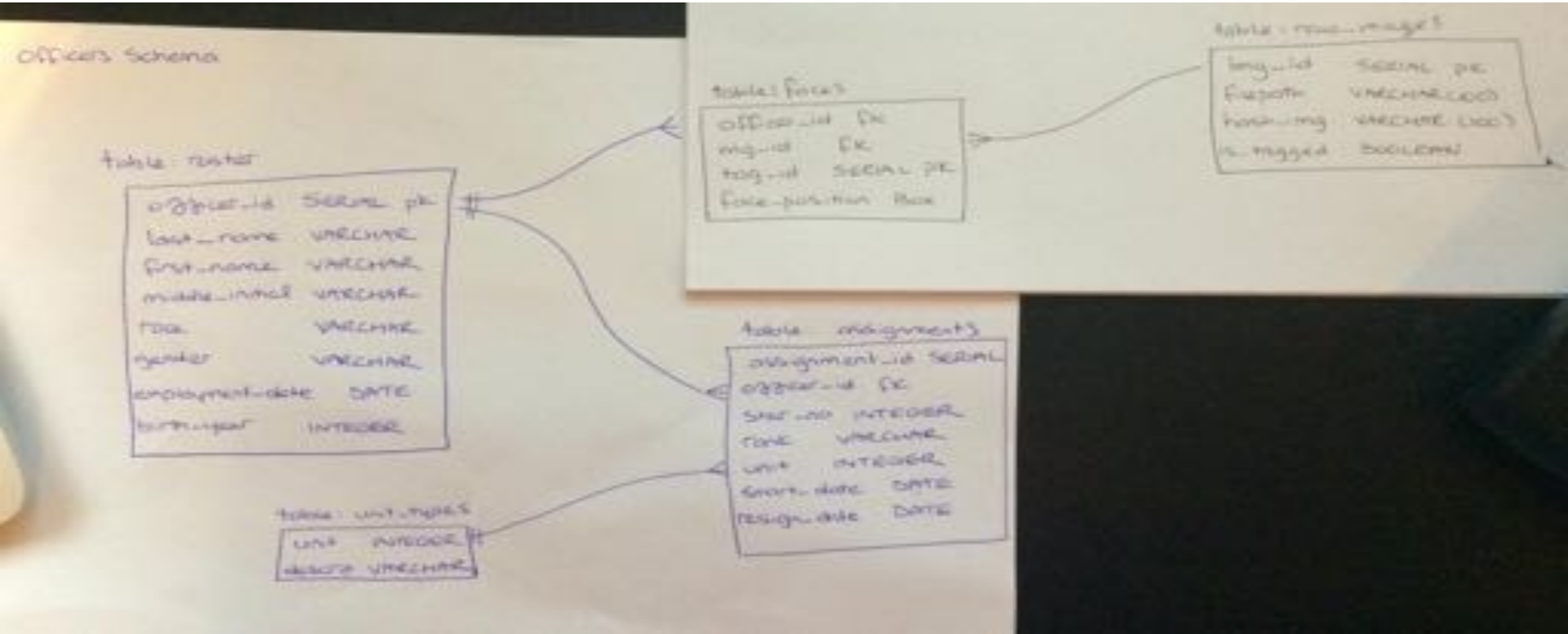
2. Analyzing requirements → Data model

Examining requirements and determining what data needs to be in the database, how data items need to be structured, in order to answer your questions.

→ High-level “blueprint” that outlines your data “house”.



Example: (early) data model from Open Oversight



Same (updated) data model, represented in another way

```
class Department(db.Model):
    __tablename__ = 'departments'
    id = db.Column(db.Integer, primary_key=True)
    name = db.Column(db.String(255), index=True, unique=True, nullable=False)
    short_name = db.Column(db.String(100), unique=False, nullable=False)

    def __repr__(self):
        return '<Department ID {}: {}>'.format(self.id, self.name)


class Officer(db.Model):
    __tablename__ = 'officers'

    id = db.Column(db.Integer, primary_key=True)
    last_name = db.Column(db.String(120), index=True, unique=False)
    first_name = db.Column(db.String(120), index=True, unique=False)
    middle_initial = db.Column(db.String(120), unique=False, nullable=True)
    race = db.Column(db.String(120), index=True, unique=False)
    gender = db.Column(db.String(120), index=True, unique=False)
    employment_date = db.Column(db.DateTime, index=True, unique=False, nullable=True)
    birth_year = db.Column(db.Integer, index=True, unique=False, nullable=True)
    assignments = db.relationship('Assignment', backref='officer', lazy='dynamic')
    face = db.relationship('Face', backref='officer', lazy='dynamic')
    department_id = db.Column(db.Integer, db.ForeignKey('departments.id'))
    department = db.relationship('Department', backref='officers')
```

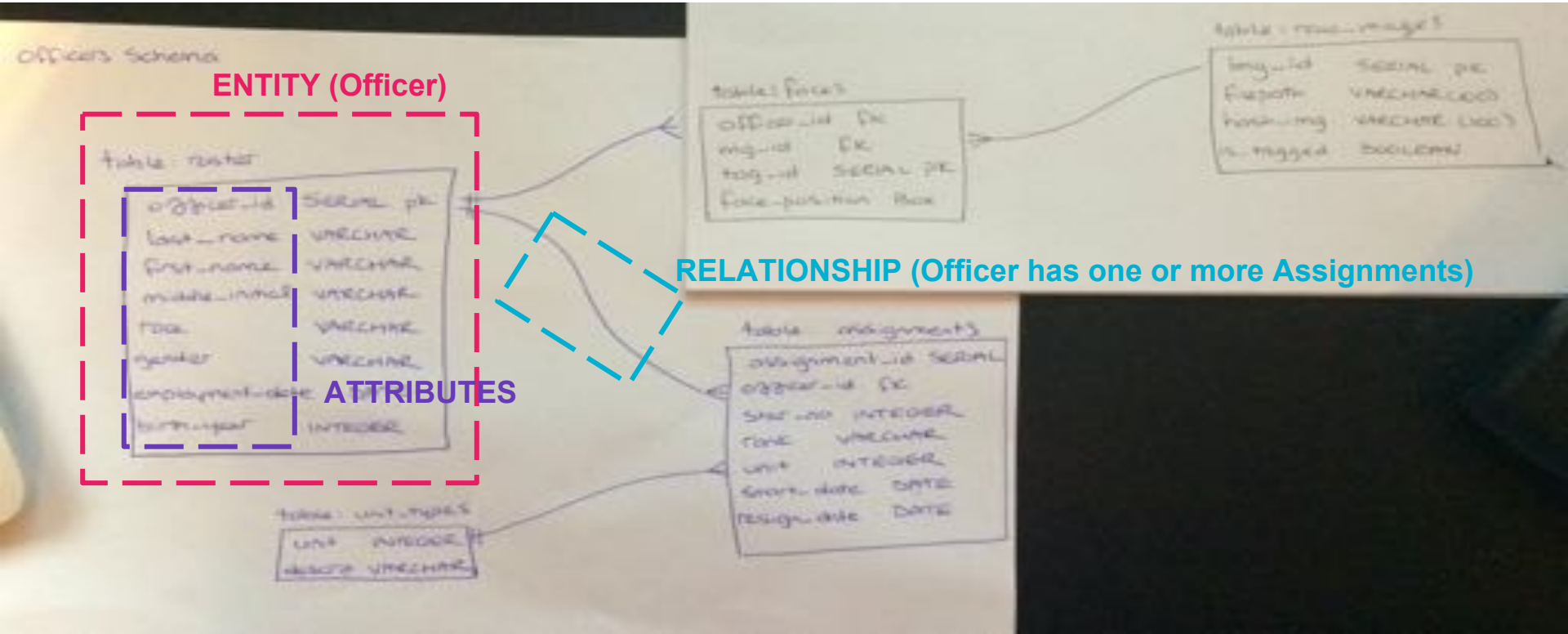
Components of a data model

Entities: The real-life things (people, objects, events, etc) that the database contains information *about.* E.g. officers, events, videos.

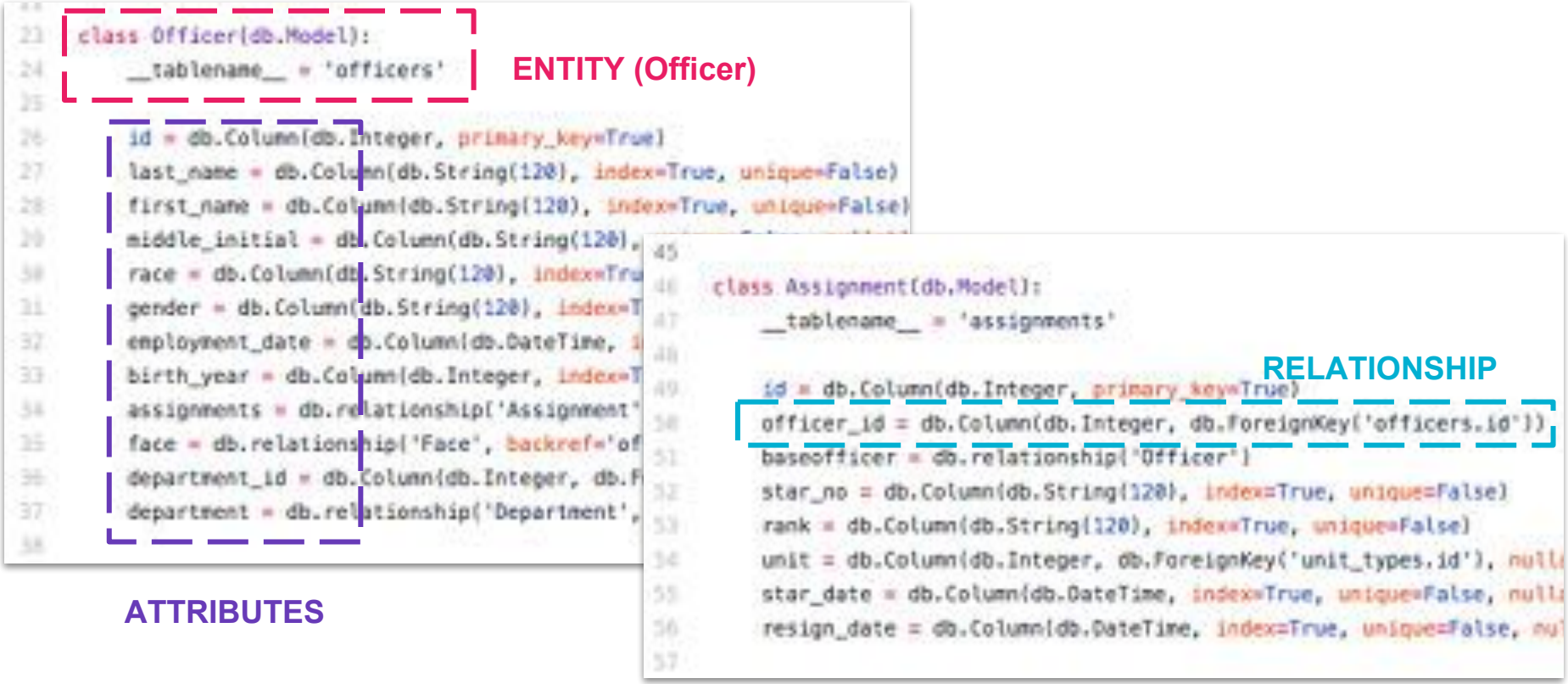
Attributes: Aspects of the entities, i.e. the fields within each entity. E.g. Officer: First name, last name, shield number.

Relationships: How different entities are associated with each other. E.g. Video *documents* an Event.

Example: Open Oversight

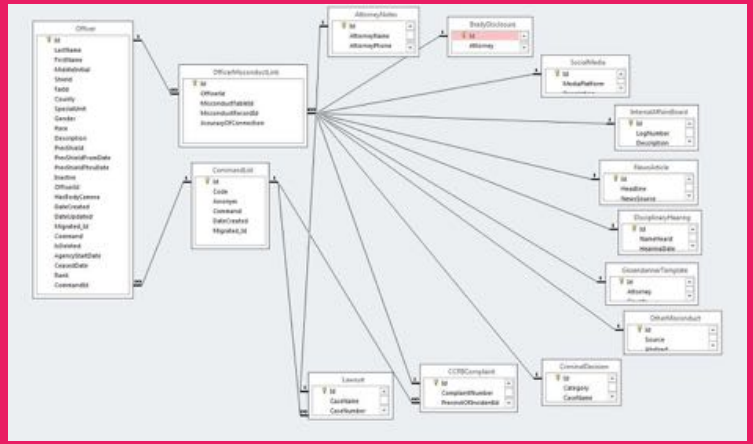


Example: Open Oversight



Let’s Data Model

Based on the questions we want to answer



What we’re going to do next:

- Identify entities.
- Identify attributes for those entities.
- Identify relationships between entities.

Identifying entities

Looking back at our questions, **what are the real-life things (people, objects, events) that we are asking questions *about*?**

Example: WhoWasInCommand's questions:

- Who is in charge of the specialized anti-riot police unit?
- What army unit has jurisdiction over what areas and for how long?
- Where did this commander previously serve, and where did they go next?
- When was a particular police unit based in a specific city?
- What allegations have civil society groups made against a unit or commander?

Example: Entities in WhoWasInCommand data model

Security Force Monitor researches and creates data about three things (or *entities*) related to security forces around the world:

- Organizations are official state or state-sanctioned organizations responsible for the internal or external security for a country, including police, army, navy, air force and other security bodies. Organizations refer to any any part of the hierarchy of a security force, ranging from a national defense ministry, to a police unit based in a small town. Organizations can also be groupings of organizations that occur, such as "operations", "joint task forces" or peacekeeping missions. The Monitor collects data about an organization's name, aliases, location, geographical areas of operation and relationships with other organizations.
- Persons are natural persons who are affiliated with, or hold positions of command over a specific organization at a particular point in time. The Monitor creates a dossier for each person, which includes their name, aliases, rank, title, role and the different organizations which they are affiliated with.
- Events are publicly-documented allegations of human rights violations committed by security forces. These include extrajudicial killings, rape, torture and other forms of violence. The Monitor does not make allegations itself, but rather compiles allegations made by governmental bodies, human rights organizations and other civil society actors. For each Event, the Monitor includes a description from the source, date(s), specific location(s), its perpetrators and the type of human rights violation.

Exercise: Identifying entities

From our requirements/questions, what/who are we trying to answer questions *about?*

- Start with the explicit subjects & objects of questions.
- Entities might not be explicit, e.g. “Do BPD officers profile by race?” -- what are we counting in order to answer this question?
- Consider if some things can be generalized into a broader entity (e.g. “searches” and “raids” as types of “Events”).

Clarify: What does having (separate) entities mean?

EVERYTHING

Incident Date	Incident Location	Victim Name	Victim Age	Victim Gender	Complainant Name	Complainant Address

VS.

INCIDENT

Incident Date	Incident Location

VICTIM

Victim Name	Victim Age	Victim Gender

COMPLAINANT

Complainant Name	Complainant Address

Why have entities / relationships?: Examples

- Can create a record for Victim without necessarily creating a Incident record first. Or can delete an Incident record without losing data about Victim.
- If updating Complainant Address for complainant with multiple complaints, only need to do it once. Less chance of inconsistency/error.
- Can associate multiple Victims to a single Incident without creating sorting / counting problems.

Identifying attributes

- Next, what are the properties/aspects of each entity that will help us answer our questions?

  E.g. “Person” entity might include: name, race, gender, immigration status, health status, etc. etc.

Example: Legal Aid database Officer entity

(Note: as you can see, it's not 100% clear what some of these attributes mean. That's why we need a **data dictionary** -- coming up later!)

Exercise: Identifying attributes

Working in pairs(?), take one of the entities we've identified, and brainstorm a list of attributes on the worksheets, which we'll post on the wall.

Refer to:

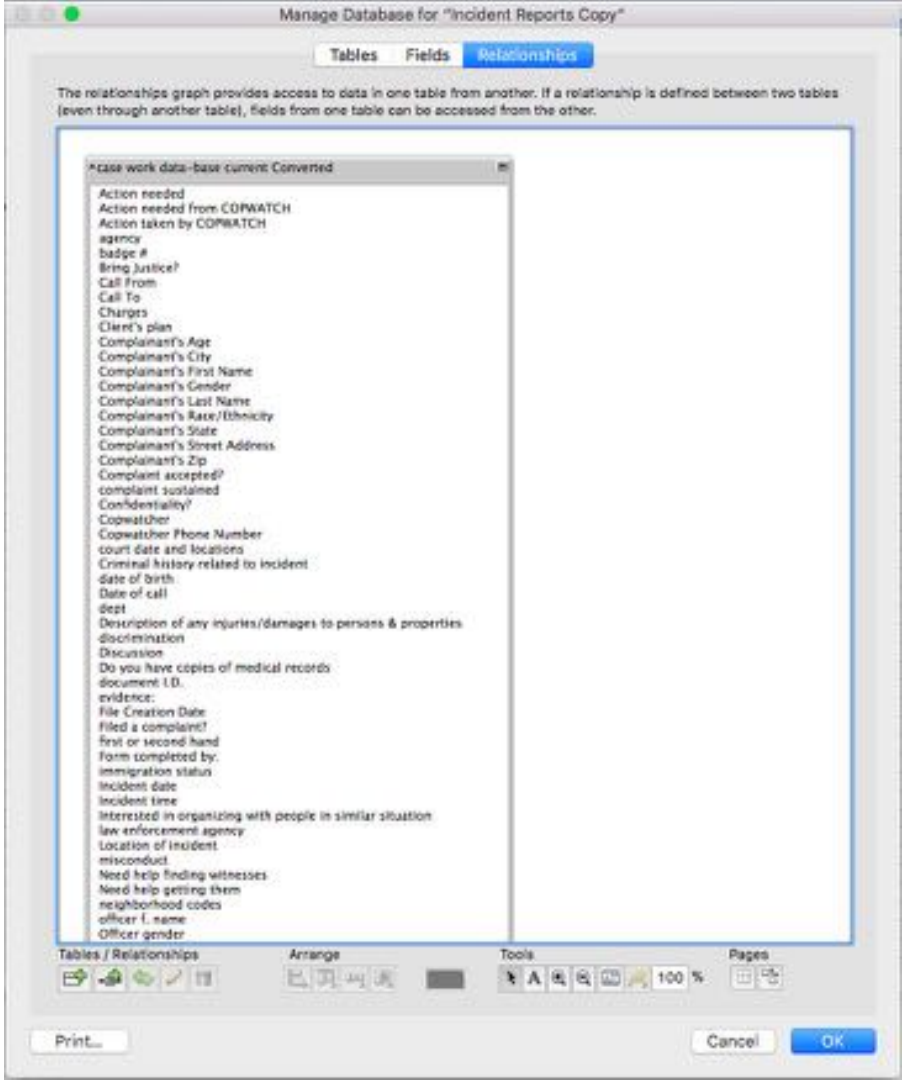
- Our questions / requirements list.
- Current Berkeley Copwatch database attributes (up next).

Tips for identifying attributes

- Any entity can have an infinite number of attributes. Choose ones that are important for identifying entity and answering questions.
- If there’s an important attribute that doesn’t belong to any of the entities we’ve identified, we will make a new entity. Don’t shoehorn attribute into entity it doesn’t describe.
- Consider what data you realistically have.

Current Berkeley Copwatch database

- Attributes list from DB
- Forms templates (1,2,3)
- Currently in one entity (incidents). Let's use these attributes but store under new entities.



Last data modeling step! Identifying relationships

Drawing the relationships that we want to examine between the entities.

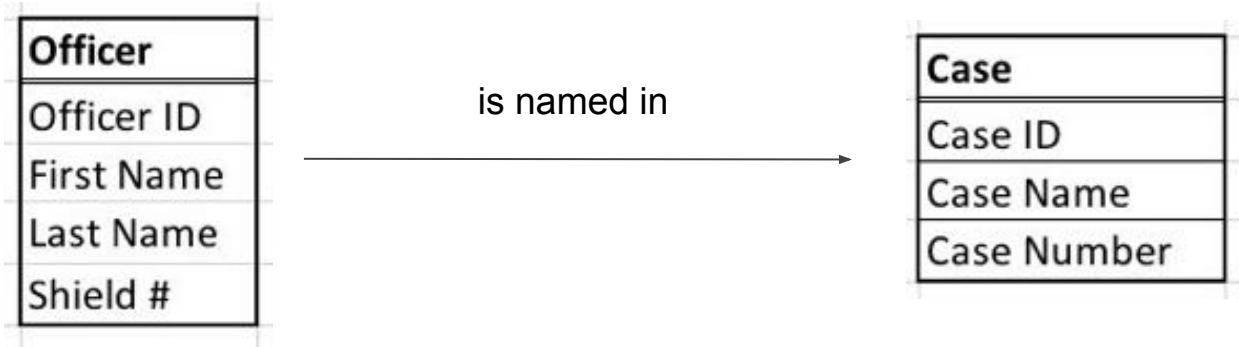
Many possible relationships, but create ones that:

- Reflect real-life relationships.
- Are as simple as possible, while allowing us to answer our questions.
- Tip: try stating the relationship in a “entity verb entity” sentence, e.g. “Videos show an Event”

Example: Identifying relationships

"Which officers have a history of lawsuits against them"?

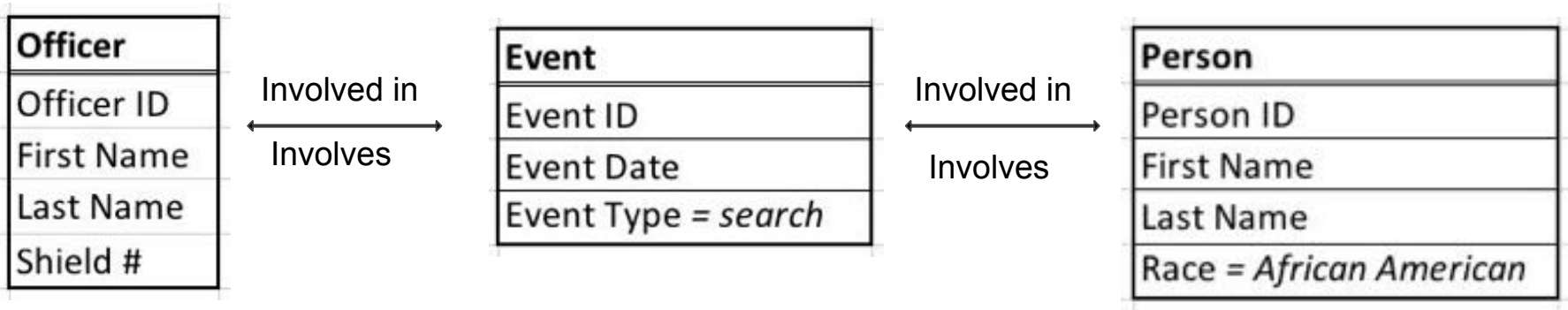
To answer this question, we should connect officer and lawsuit entities:



Example: Identifying relationships

"Do officers conduct searches based on people's race?"

This question can involve 3 related entities.



Exercise: Identifying relationships

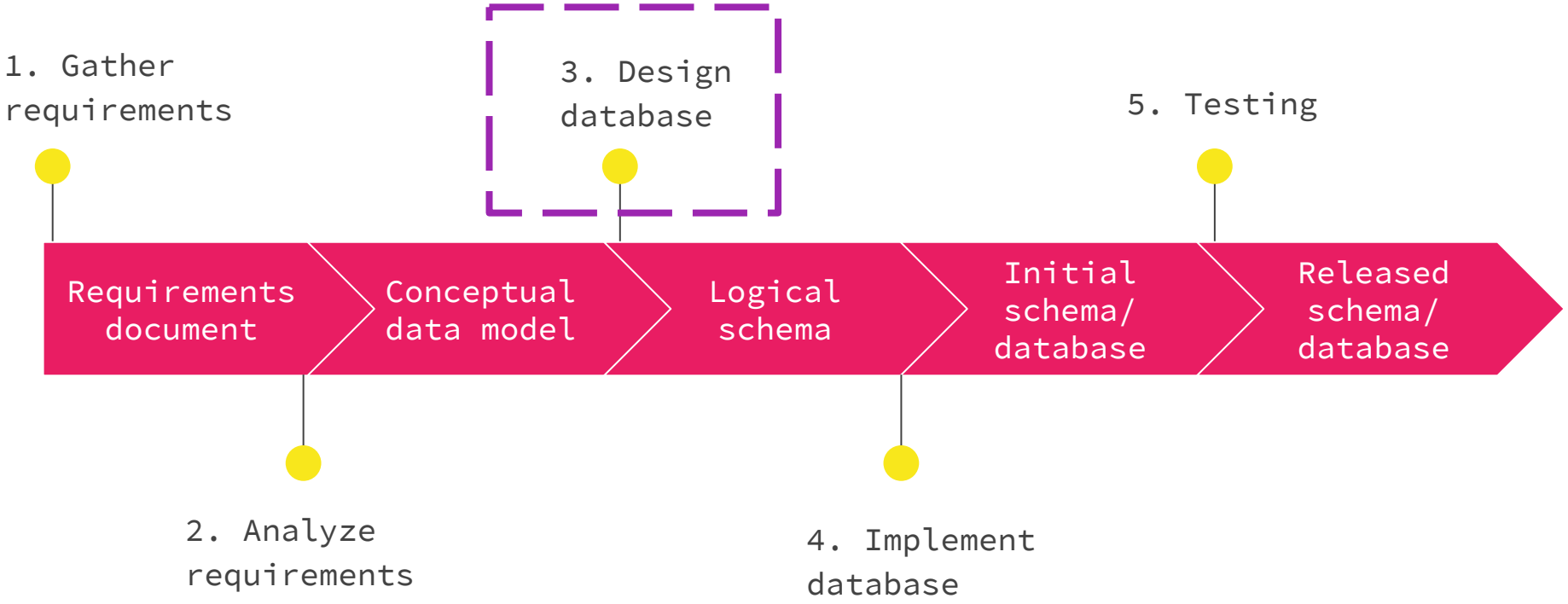
Using the string, let’s map out the relationships we want to explore between entities.

Relationships should:

- Help us answer our questions.
- Reflect real-life association.
- As simple as possible, avoiding redundancy and loops.

Congratulations!
We have a conceptual data model!

Database development process - where we are

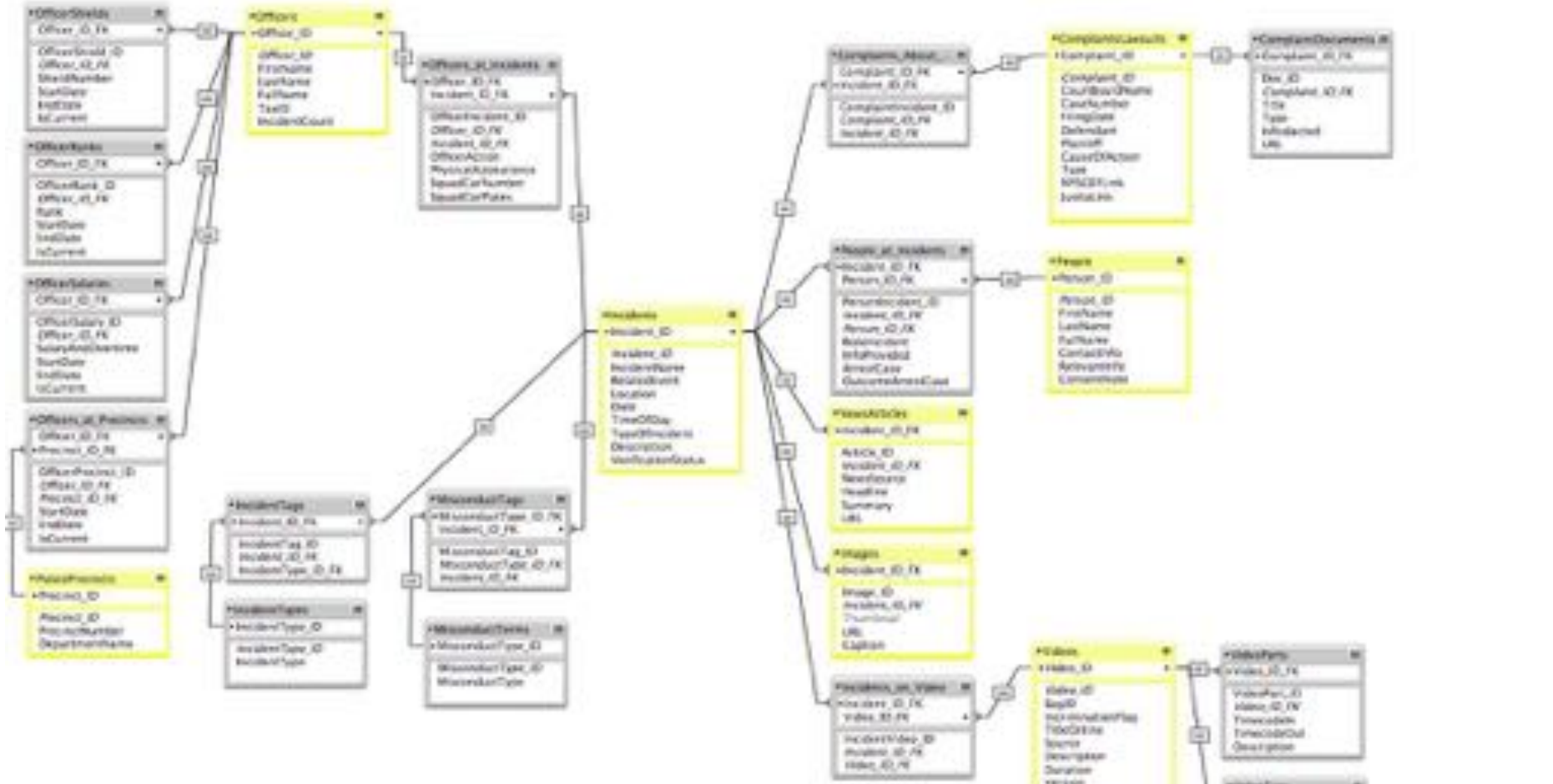


3. Design database: Elaborating the data model

Logical data model: Follows from conceptual model, but goes into more technical detail about the database structure (e.g. keys or ids, direction of relationship lines, additional tables to handle many-to-many relationships, etc).

Can use this to implement database in any database software.

Often visualized as an Entity Relationship Diagram.



Example: El Grito entity relationship diagram

Design database: setting data rules

Database design also involves setting rules for each data element:

- Definition of the field/attribute.
- Controlled vocabularies or defined value lists.
- Syntax rules for values.
- Whether a value is required or optional.

These rules can be documented in a **Data Dictionary.**

Example: Data dictionary

A	B	C	D	E	F
ELEMENT NAME	**FFN ID**	**Source ID**	**Title(s)**	**Description**	**Tags**
ELEMENT DEFINITION	Record reference number.	ID given to source media.	Cataloger-supplied title based on content.	Cataloger-supplied description based on viewing the content.	Cataloger-supplied keywords.
ELEMENT RULES	"F" followed by 5-digit sequential number.	Enter as written on source media.	Generally, enter text as given on the source (label), verbatim, unless correction or more elaboration is needed. Title on source is the text written on the spine of the tape, or the top text written on the DVD. Multiple titles should be separated with "/"	Generally, enter text as given on the source (label), verbatim, unless correction or more elaboration is needed (e.g. names, locations, affiliations, etc). Multiple descriptions should be separated with "/" in the same order as titles are entered.	Any words that describe the video that may help users search for content. Separate terms with commas.
ELEMENT REQUIREMENT	Required	Required.	Required	Optional	Optional
EXAMPLE ENTRY	F00001	S019	Black 14 Gathering / Rumah Panjang Jinjang Pongal	Anwar Ibrahim speech in Kelab Sultan Sulaiman.	Kampung Baru, press conference, Anwar Ibrahim, election, protest, Malaysia

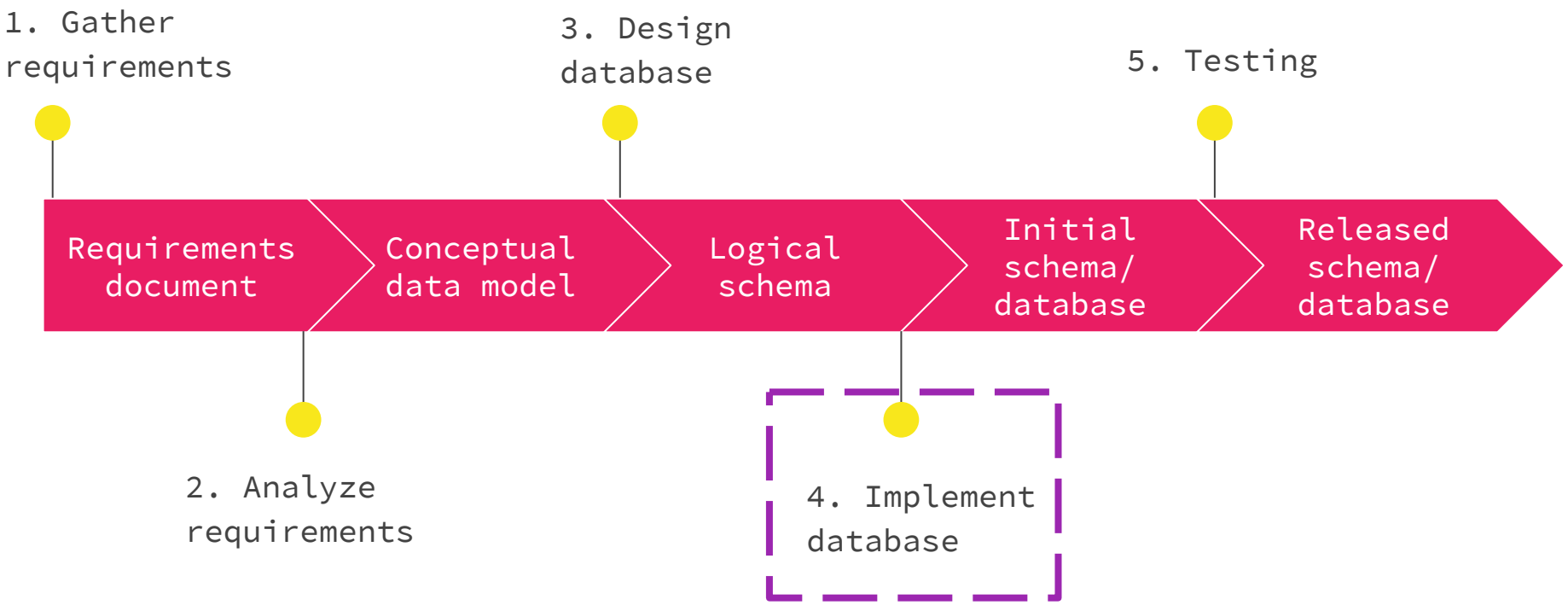
Exercise: Data Dictionary

Let's focus on the attributes that would be helpful to define as a group.

For controlled vocabularies, we can use current BC database value lists as a start.

- Any attributes need to be more clearly defined?
- Any attributes that need a controlled vocabulary? Do value terms need to be defined?
- Do we need to apply specific rules to any elements?

Database development process - next steps



Implementation: Next steps

- Implement the database for initial testing.
- Choose appropriate database management software / platform.
- Create views / presentation of the data (may have different views for different user types).