Graph data modeling

NoSQL: Lecture 1, part 2

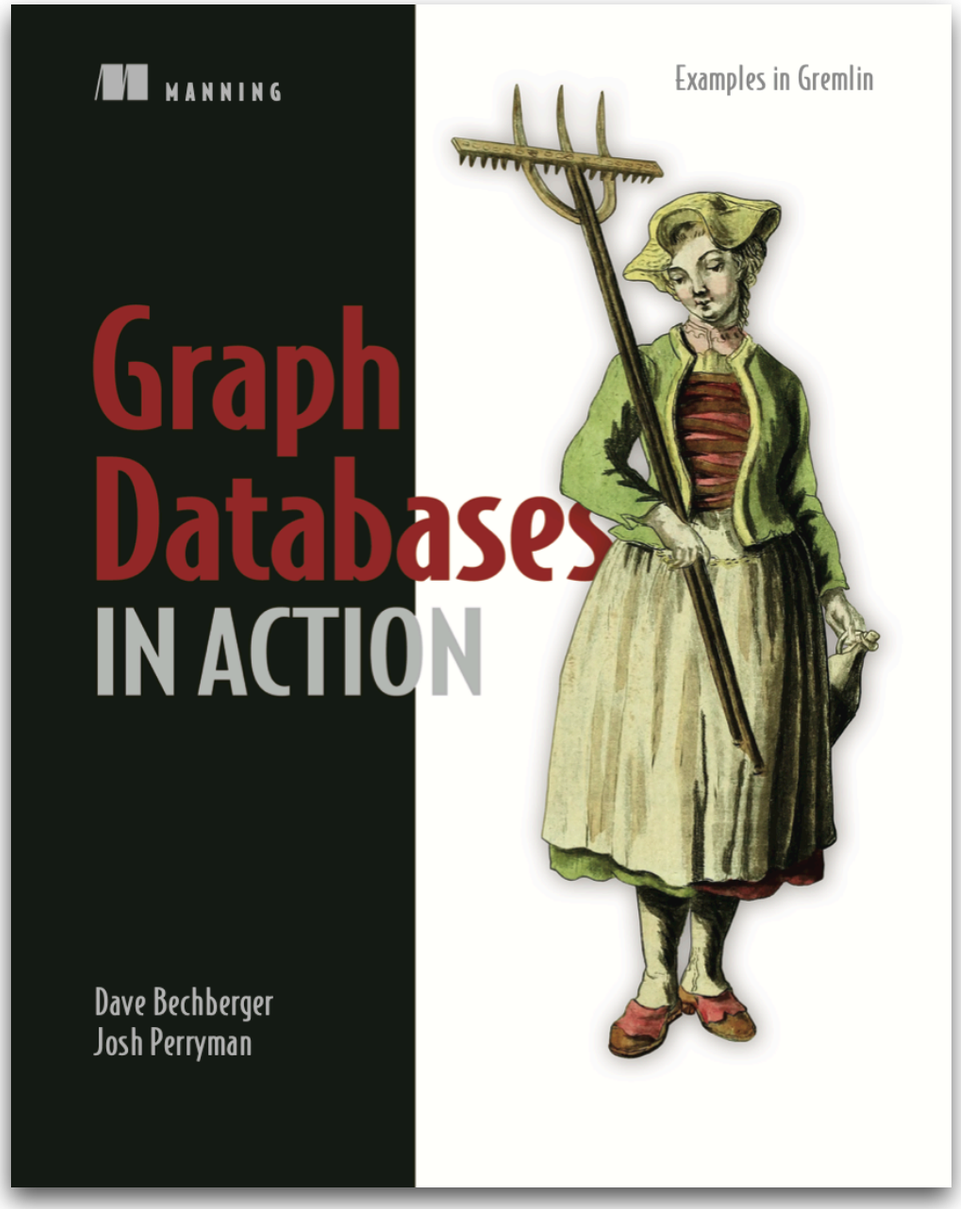
Piotr Fulmański

FACULTY OF MATHEMATICS
AND COMPUTER SCIENCE
University of Lodz

Graph databases In Action

by Dave Bechberger and Josh Perryman

Manning Publications, 2020



The data model

The data model

- The data model is a significant part of the mental picture in software production cycle. A well thought out data model, with a helpful level of abstraction and consistent naming conventions, is intuitive to work with, maybe even a joy to use.

The data model

- The data model is a significant part of the mental picture in software production cycle. A well thought out data model, with a helpful level of abstraction and consistent naming conventions, is intuitive to work with, maybe even a joy to use.

- When building anything, no matter whether it is a software or a real existing thing you can touch, it’s critically important to start out with a good mental picture of the end result.

The data model

- The data model is a significant part of the mental picture in software production cycle. A well thought out data model, with a helpful level of abstraction and consistent naming conventions, is intuitive to work with, maybe even a joy to use.

- When building anything, no matter whether it is a software or a real existing thing you can touch, it’s critically important to start out with a good mental picture of the end result.

- This picture includes the scope the solution should address and the requirements to complete the solution. The more details that this mental picture provides, the easier it’ll be to build that solution.

The data modeling process

The data modeling process

- The process of data modeling is about translating certain real-world problems, understandings, and questions into software, usually focused on creating a technical implementation involving a database.

The data modeling process

- The process of data modeling is about translating certain real-world problems, understandings, and questions into software, usually focused on creating a technical implementation involving a database.
- In case of relational database applications this includes:

The data modeling process

- The process of data modeling is about translating certain real-world problems, understandings, and questions into software, usually focused on creating a technical implementation involving a database.
- In case of relational database applications this includes:
    - identifying and understanding the problem,

The data modeling process

- The process of data modeling is about translating certain real-world problems, understandings, and questions into software, usually focused on creating a technical implementation involving a database.
- In case of relational database applications this includes:
    - identifying and understanding the problem,
    - determining the entities and relationships in that problem,

The data modeling process

- The process of data modeling is about translating certain real-world problems, understandings, and questions into software, usually focused on creating a technical implementation involving a database.
- In case of relational database applications this includes:
  - identifying and understanding the problem,
  - determining the entities and relationships in that problem,
  - and then creating a representation of that problem in the database.

The data modeling process

- The process of data modeling is about translating certain real-world problems, understandings, and questions into software, usually focused on creating a technical implementation involving a database.
- In case of relational database applications this includes:
    - identifying and understanding the problem,
    - determining the entities and relationships in that problem,
    - and then creating a representation of that problem in the database.
- In case of graph databases… hmmmm...
  (see *Access pattern* in next part: *Data modelling terms*)

Data modeling terms

- ***Entity*** **–** Describe the “things” or type of things in the domain, such as users, devices, geographic locations etc. As we move from problem definition and conceptual modeling, entities will often become **vertices** in the logical model and technical implementation. Commonly represented by nouns.

- ***Relationship*** **–** Describe how entities interact with one another. It could something like “moves” as in “a vehicle moves to a location” or “friends,” in the Facebook sense of this word as a verb, as in “a person friends another person.” As we move from problem definition and conceptual modeling, relationships will often become **edges** in the logical model and technical implementation. Often represented by verbs (or verbal phrases).

- ***Attribute*** **–** Describes a **property** of the entity or relationship, so it always exists in their context.

- ***Access pattern*** **–** Describes either questions or methods of interaction in the domain. Examples could be things such as “Where is this vehicle going?” or “Who are this person’s friends?” As we move from problem definition and conceptual modeling, access patterns will often become **queries** in the logical model and technical implementation.

Use the right language

Vertex or node?

Use the right language

Vertex or node?

- As we saw, there are some obvious correlations between *data modeling terms* (entity, relationship) and the *graph elements* (vertex, edge).

Use the right language

Vertex or node?

- As we saw, there are some obvious correlations between *data modeling terms* (entity, relationship) and the *graph elements* (vertex, edge).
- Why use separate terms when they all mean the same things?

Use the right language

Vertex or node?

- As we saw, there are some obvious correlations between *data modeling terms* (entity, relationship) and the *graph elements* (vertex, edge).
- Why use separate terms when they all mean the same things?
- Because they are not the same things.

Use the right language

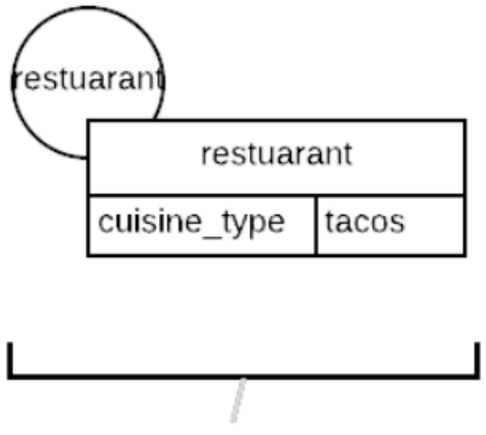
Vertex or node?

- As we saw, there are some obvious correlations between *data modeling terms* (entity, relationship) and the *graph elements* (vertex, edge).
- Why use separate terms when they all mean the same things?
- Because they are not the same things.
- It is perfectly normal to have an *entity* in the conceptual model implemented in the logical model either *as a property* on a vertex or *as another vertex*:

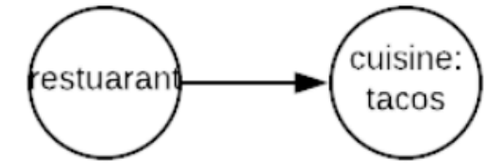
Use the right language

Vertex or node?

- As we saw, there are some obvious correlations between *data modeling terms* (entity, relationship) and the *graph elements* (vertex, edge).
- Why use separate terms when they all mean the same things?
- Because they are not the same things.
- It is perfectly normal to have an *entity* in the conceptual model implemented in the logical model either *as a property* on a vertex or *as another vertex*:



Graph implemention of cuisine as a property



Graph implemention of cuisine as another vertex

Use the right language

Vertex or node?

Note:

data modeling is about translating a real-world problem into a technical domain.

The terms vertex and edge are *technical terms* used when working with a specific type of data engine: a graph database. If we were using a relational database, the technical terms would be *table* and *column*.

But data modeling starts with engagement with the business, with users and their perspectives. The business and end users do not think in terms of vertices and edges, nor do they use these terms in their normal day to day tasks, and they shouldn't. The process uses different terms, like entity and relationship, to remind us that the conceptual model is a tool for communicating requirements between the end users and developers.

Steps of data modeling process

Steps of data modeling process

1. **Understand the problem** Explore the project goals to ensure that the domain and scope of the problem is clear. Specify the common terms and the core access patterns of the users.
    1. **Who:** Business, Developers
    2. **Tools:** Domain, Scope, Business Entities, Functionality
    3. **Result:** Textual description of problem

Steps of data modeling process

1. **Understand the problem** Explore the project goals to ensure that the domain and scope of the problem is clear. Specify the common terms and the core access patterns of the users.
    1. **Who:** Business, Developers
    2. **Tools:** Domain, Scope, Business Entities, Functionality
    3. **Result:** Textual description of problem
2. **Create a whiteboard or conceptual model** Turn text into a pictures. drawing a diagram which makes sense to the business users and will be useful to the technical developers. Create high-level picture of the problem domain from the business perspective.
    1. **Who:** Business, Developers
    2. **Tools:** Entities, Relationships, Access Patterns
    3. **Result:** Drawing with entities and their relationships

Steps of data modeling process

Steps of data modeling process

3. **Create a logical data model** Combine the domain defined in the first step with the conceptual model from the second step to create the physical description of the graph data model. This includes defining the vertices, defining the edges, as well as specifying the properties on those vertices and edges. This completes the data model for our use case.

    1. **Who:** Developers
    2. **Tools:** Vertices, edges, properties
    3. **Result:** Diagram of graph elements

Steps of data modeling process

3. **Create a logical data model** Combine the domain defined in the first step with the conceptual model from the second step to create the physical description of the graph data model. This includes defining the vertices, defining the edges, as well as specifying the properties on those vertices and edges. This completes the data model for our use case.
    1. **Who:** Developers
    2. **Tools:** Vertices, edges, properties
    3. **Result:** Diagram of graph elements
4. **Test the model** Validate that our developed model satisfies the defined problem, verify its coherence.
    1. **Who:** Developers
    2. **Tools:** Results from all preceding steps
    3. **Result:** Coherence of the outputs of prior steps

Step 1: Understand the problem

DOMAIN AND SCOPE QUESTIONS

Define the boundaries of the problem.

Every problem can expand in infinite directions, so the more precisely we define the scope, the more likely we are to succeed. If we make the domain too broad, then then we risk not understanding the boundaries of the problem and may never complete the application.

Questions:

- What will application do for users?
- What types of information does the application need to record to perform these tasks?
- Who are the users of our application?

Step 1: Understand the problem

BUSINESS ENTITLES QUESTIONS

Identify the fundamental building blocks of our application and how they are related to one another.

Questions:

- What sort of items or things does the application utilize?
- How do these items interact with one another?
- What are the critical pieces of data you need to know about each entity?

Step 1: Understand the problem

FUNCTIONALITY QUESTIONS

Identify how a user is going to interact with the system and/or what problems they want the system to solve for them.

Questions:

- How are people going to use the system?
- What questions does application need to answer for the user?

Step 2: Create a whiteboard model

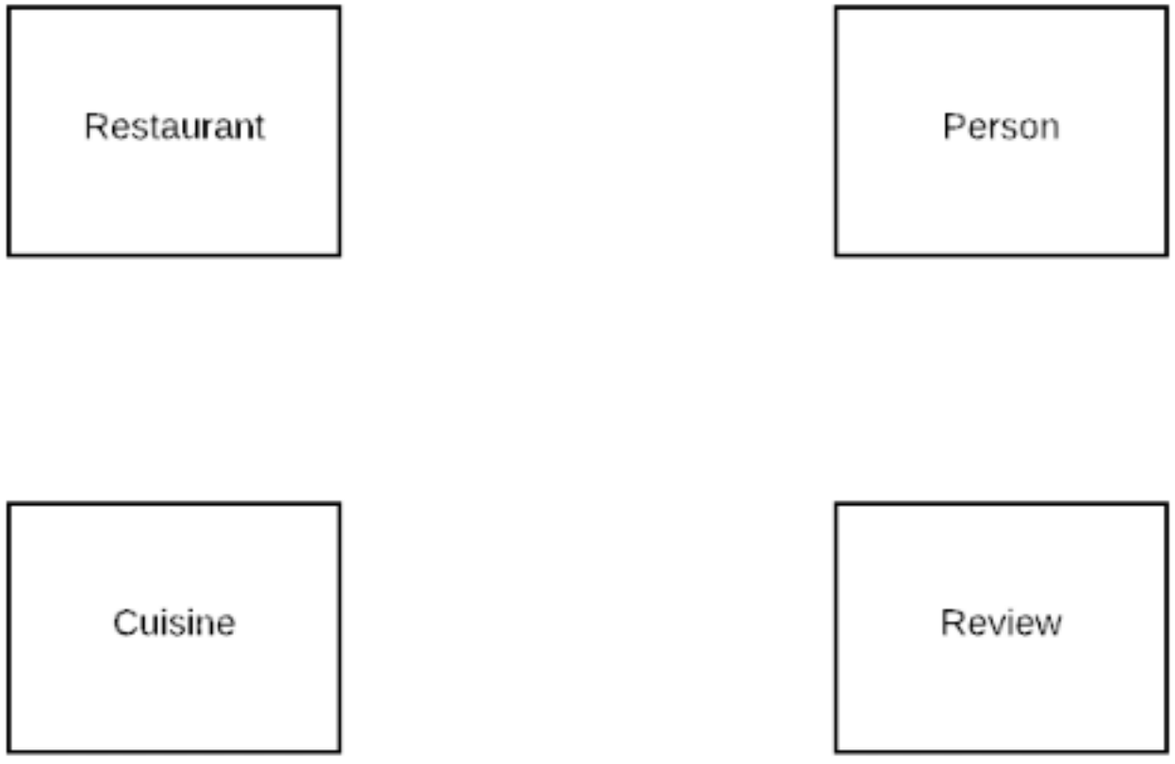
IDENTIFYING AND GROUPING ENTITIES

Answer for the question: ***what***

Step 2: Create a whiteboard model

IDENTIFYING AND GROUPING ENTITIES

Answer for the question: ***what***



Step 2: Create a whiteboard model

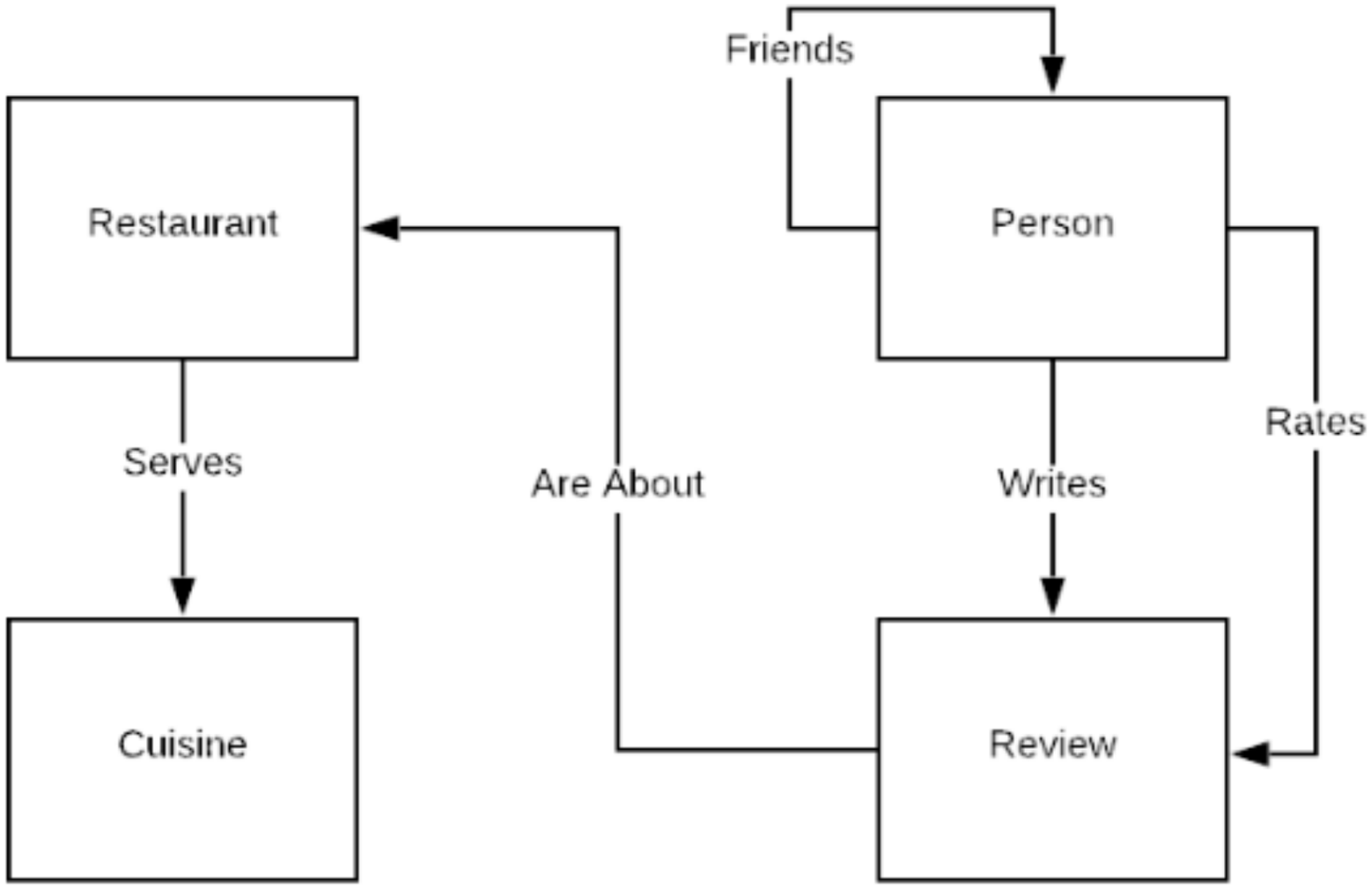
IDENTIFYING RELATIONSHIPS BETWEEN ENTITIES

Answer for the question: ***how***

Step 2: Create a whiteboard model

IDENTIFYING RELATIONSHIPS BETWEEN ENTITIES

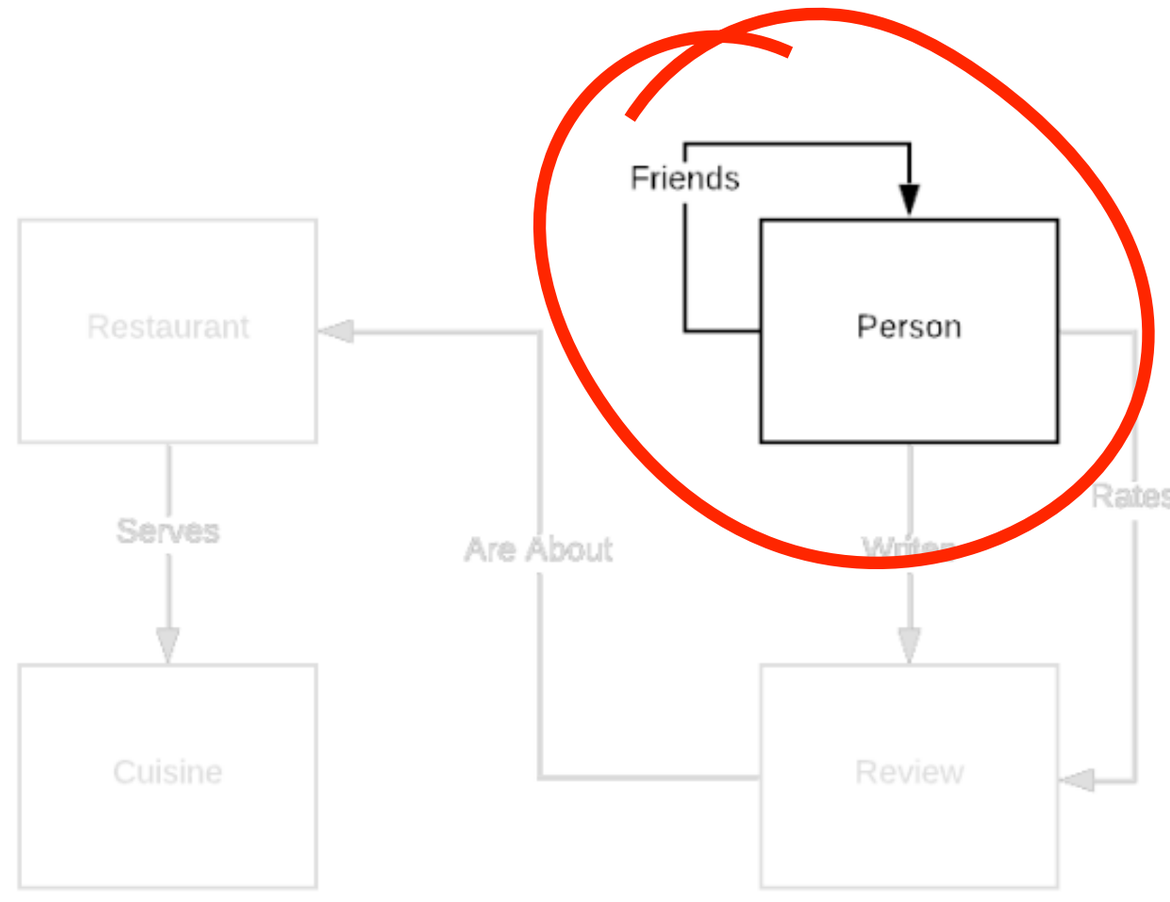
Answer for the question: ***how***



Step 3: Create a logical data model

TRANSLATE ENTITIES TO VERTICES

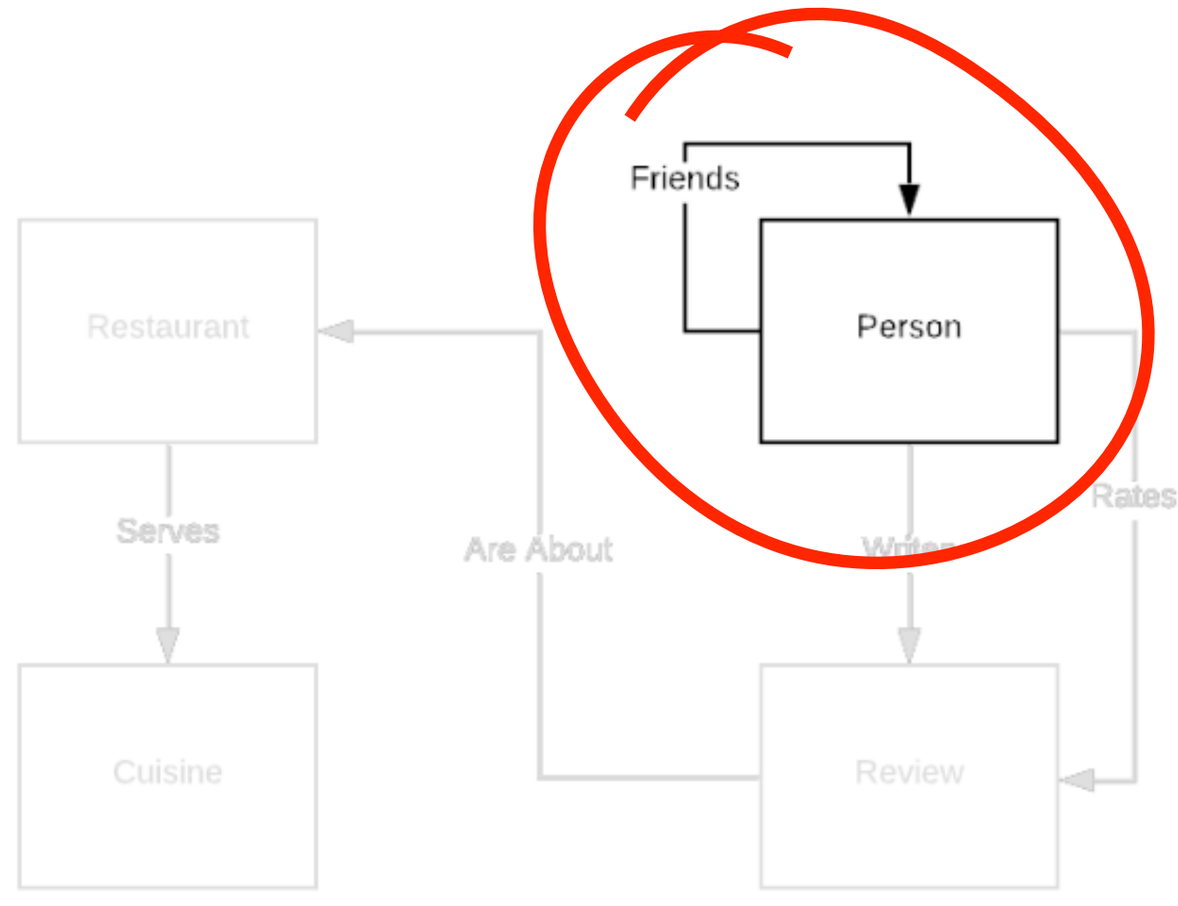
- Finding the conceptual entities
- Naming the vertex labels



Step 3: Create a logical data model

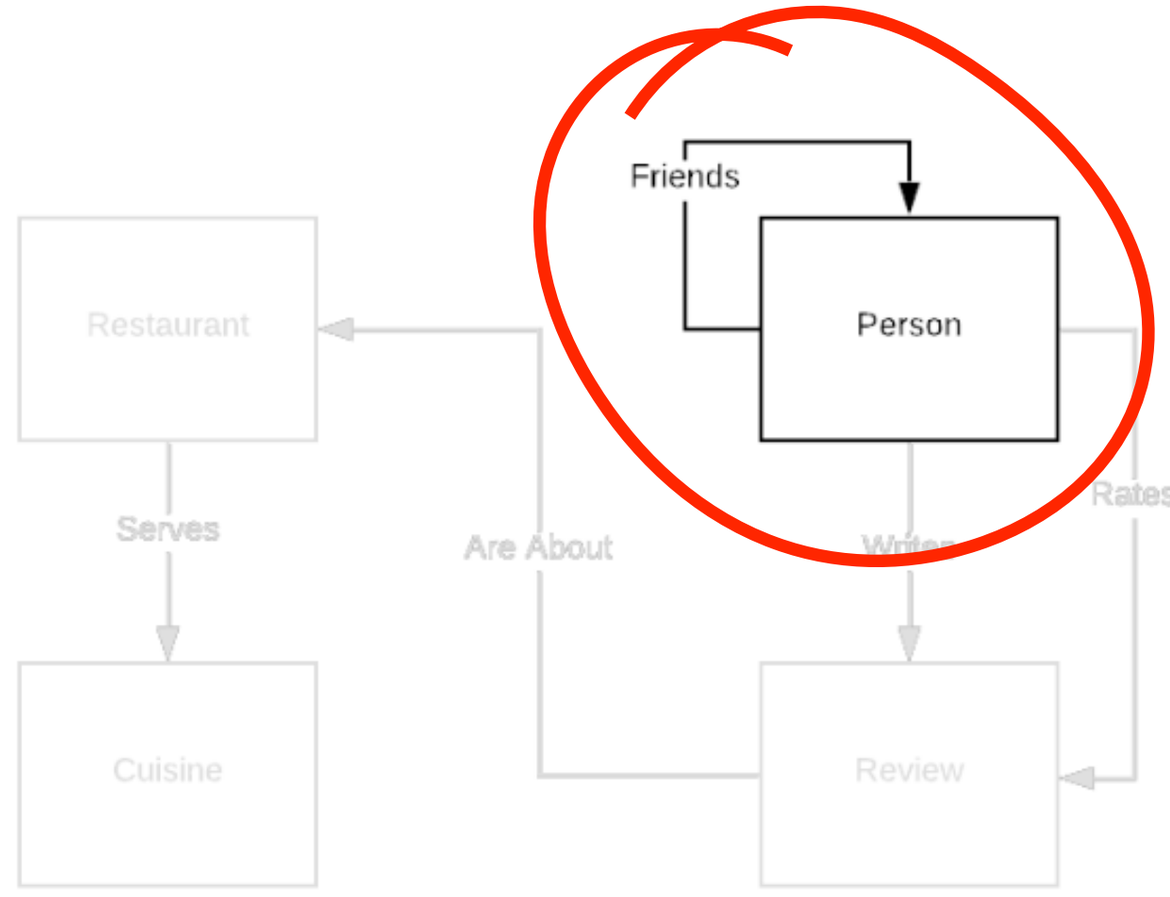
TRANSLATE ENTITIES TO VERTICES

- Finding the conceptual entities
- Naming the vertex labels



Step 3: Create a logical data model

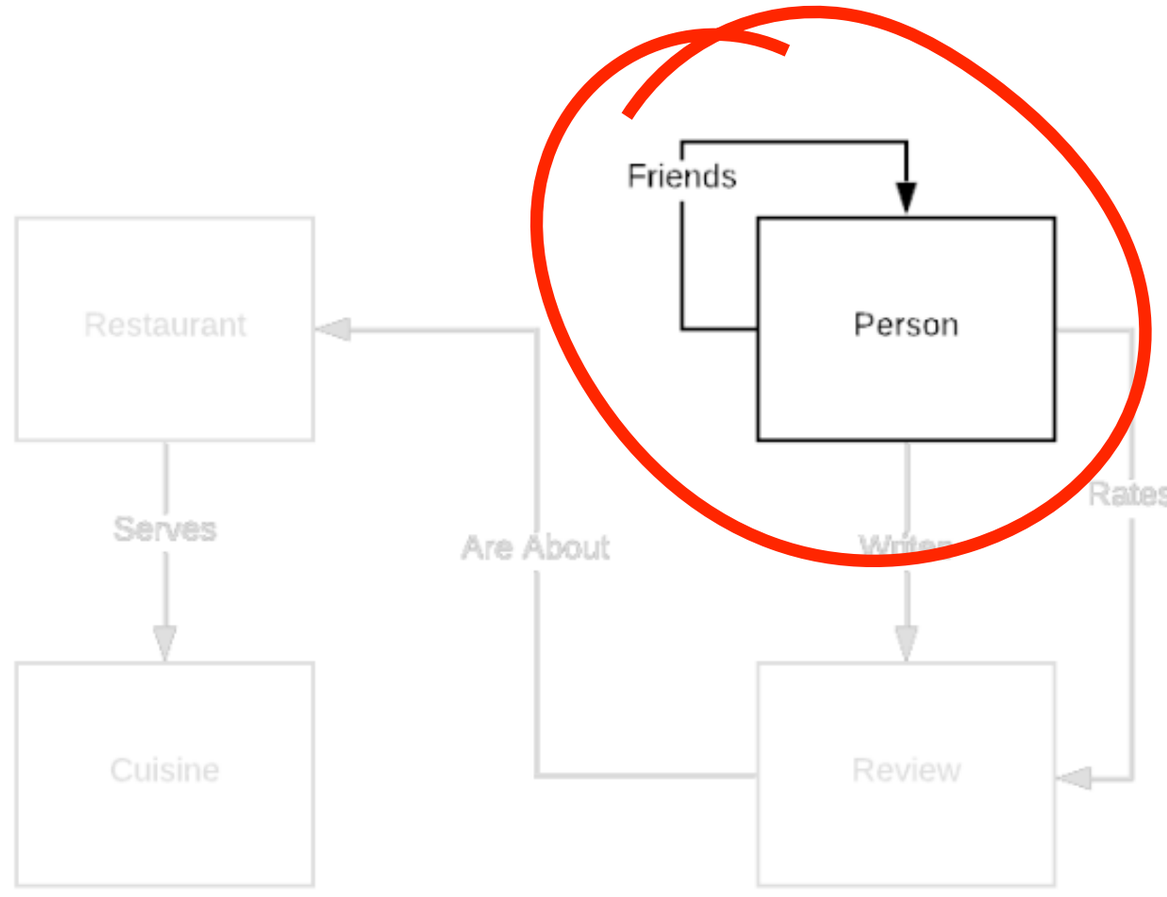
TRANSLATE RELATIONSHIPS TO EDGES



Step 3: Create a logical data model

TRANSLATE RELATIONSHIPS TO EDGES

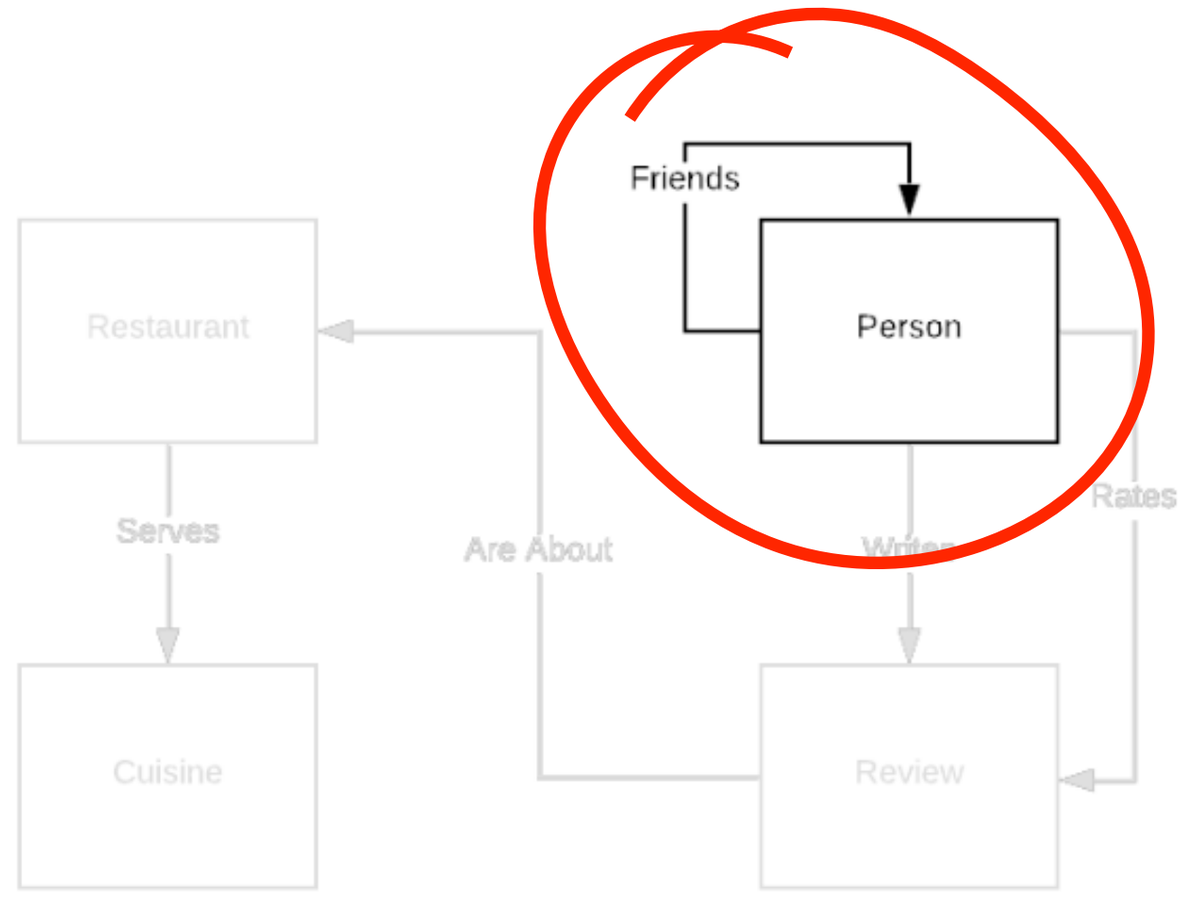
- Finding the relationships



Step 3: Create a logical data model

TRANSLATE RELATIONSHIPS TO EDGES

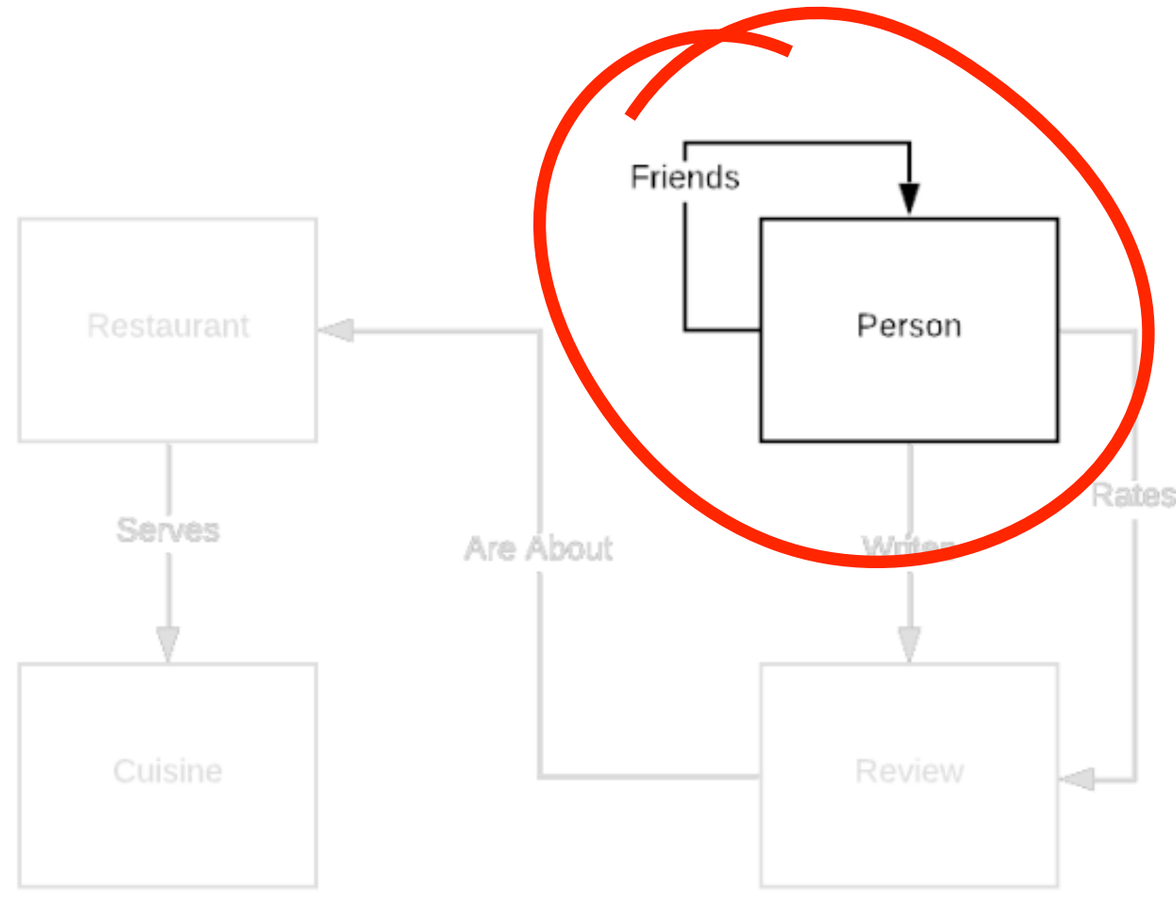
- Finding the relationships
- Naming the edge labels

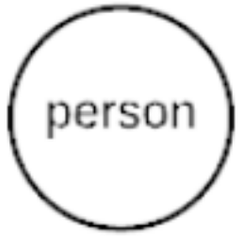


Step 3: Create a logical data model

TRANSLATE RELATIONSHIPS TO EDGES

- Finding the relationships
- Naming the edge labels

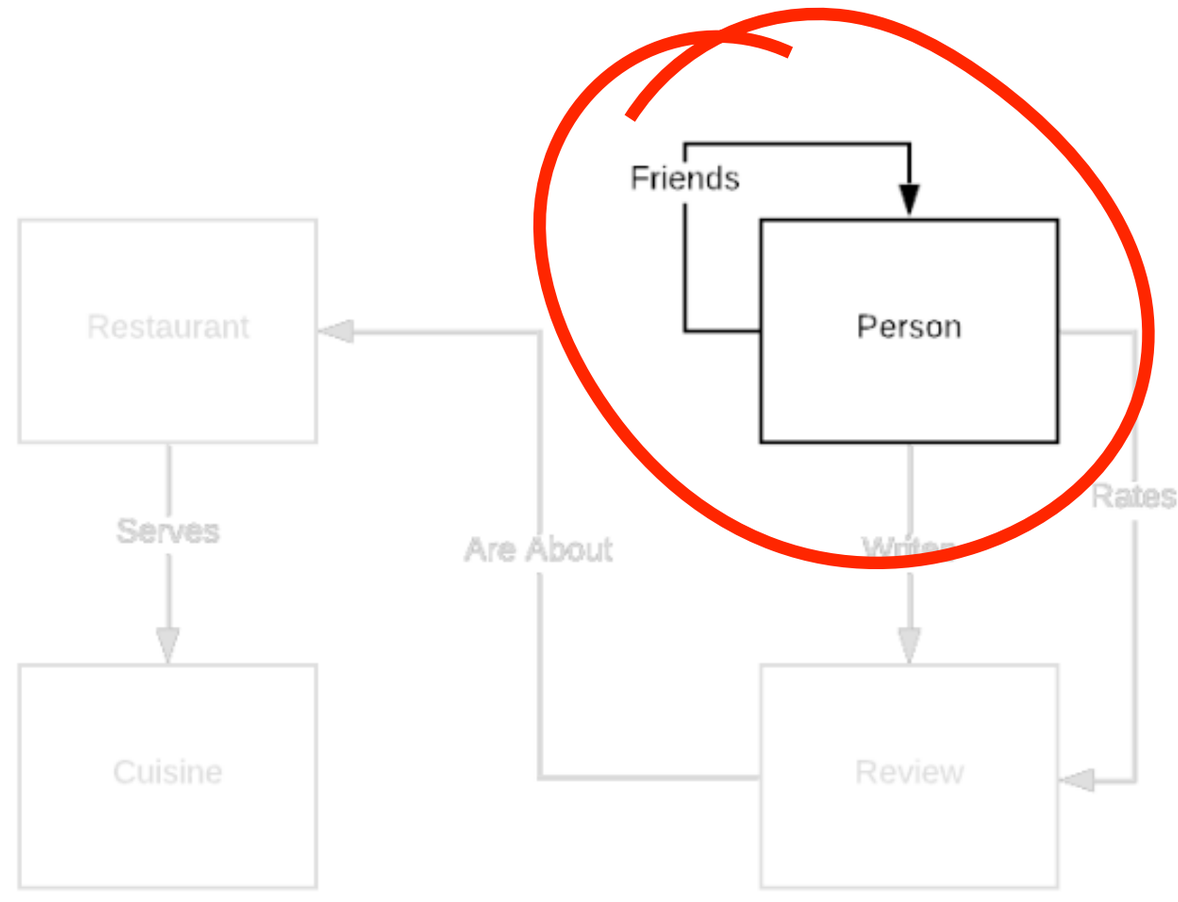




Step 3: Create a logical data model

TRANSLATE RELATIONSHIPS TO EDGES

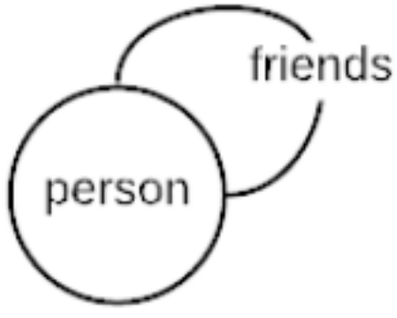
- Finding the relationships
- Naming the edge labels

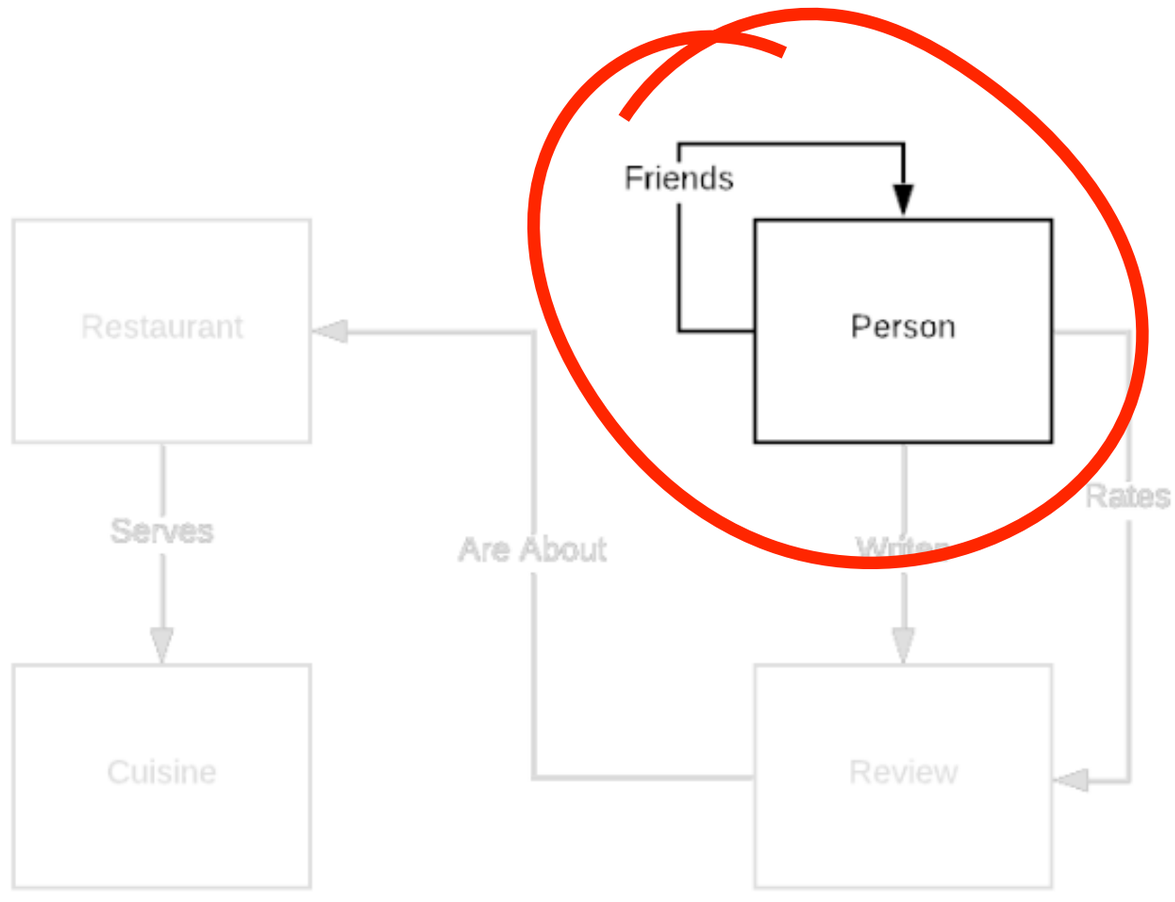


Step 3: Create a logical data model

TRANSLATE RELATIONSHIPS TO EDGES

- Finding the relationships
- Naming the edge labels

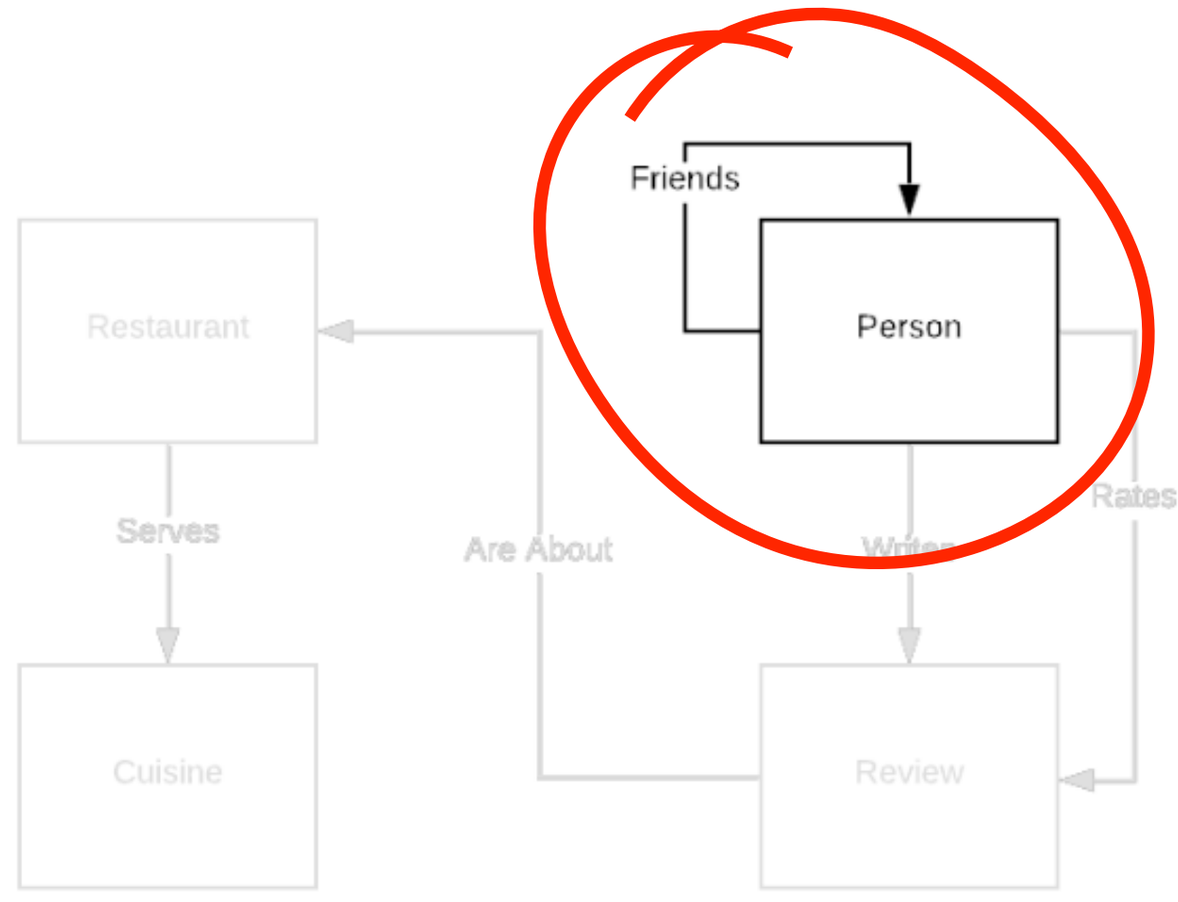




Step 3: Create a logical data model

TRANSLATE RELATIONSHIPS TO EDGES

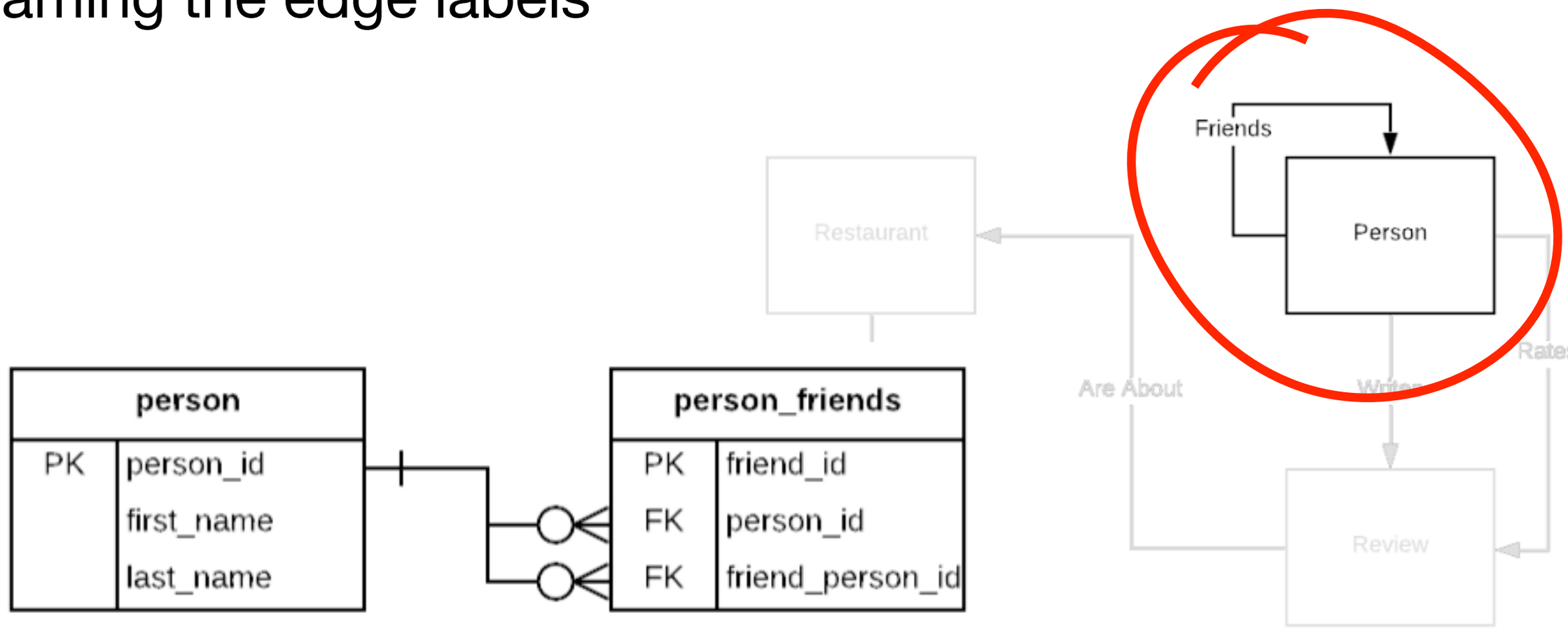
- Finding the relationships
- Naming the edge labels



Step 3: Create a logical data model

TRANSLATE RELATIONSHIPS TO EDGES

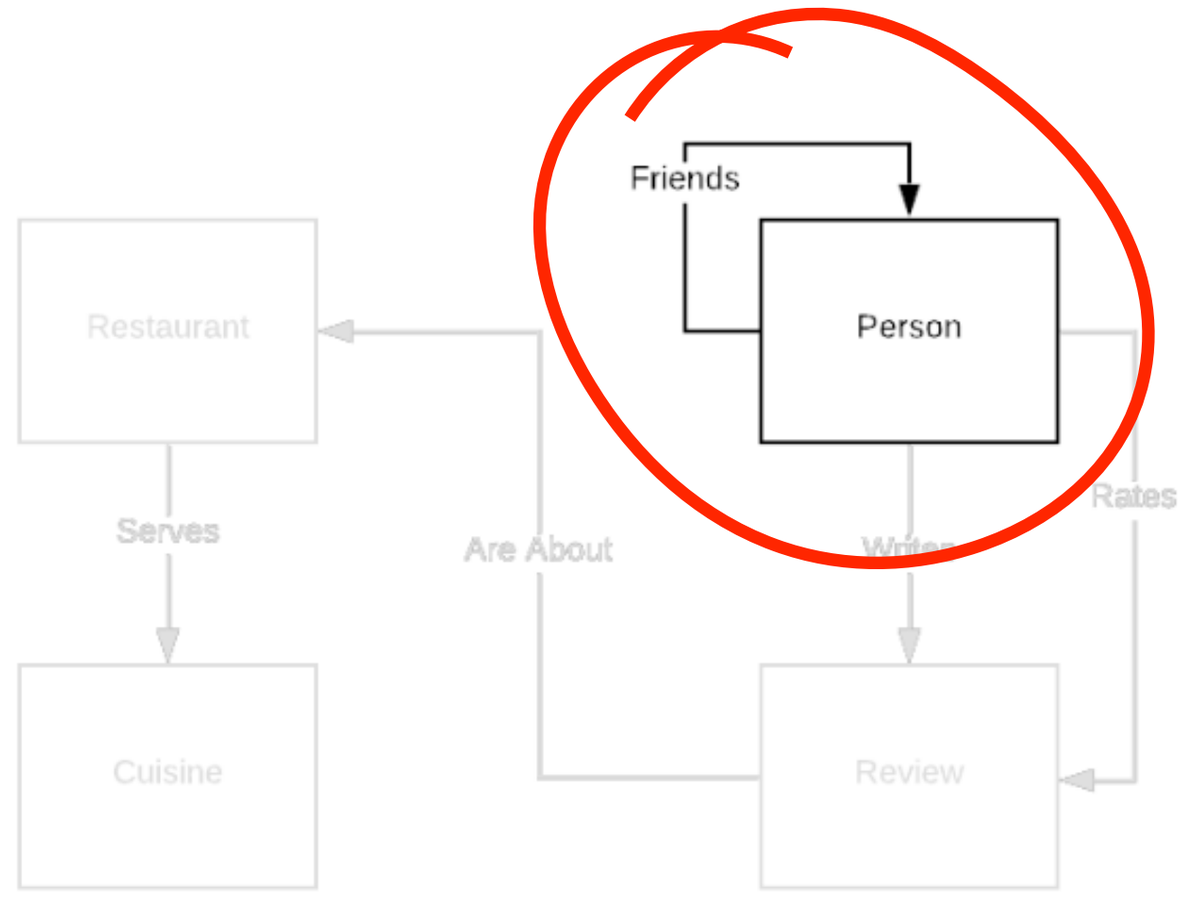
- Finding the relationships
- Naming the edge labels



Step 3: Create a logical data model

TRANSLATE RELATIONSHIPS TO EDGES

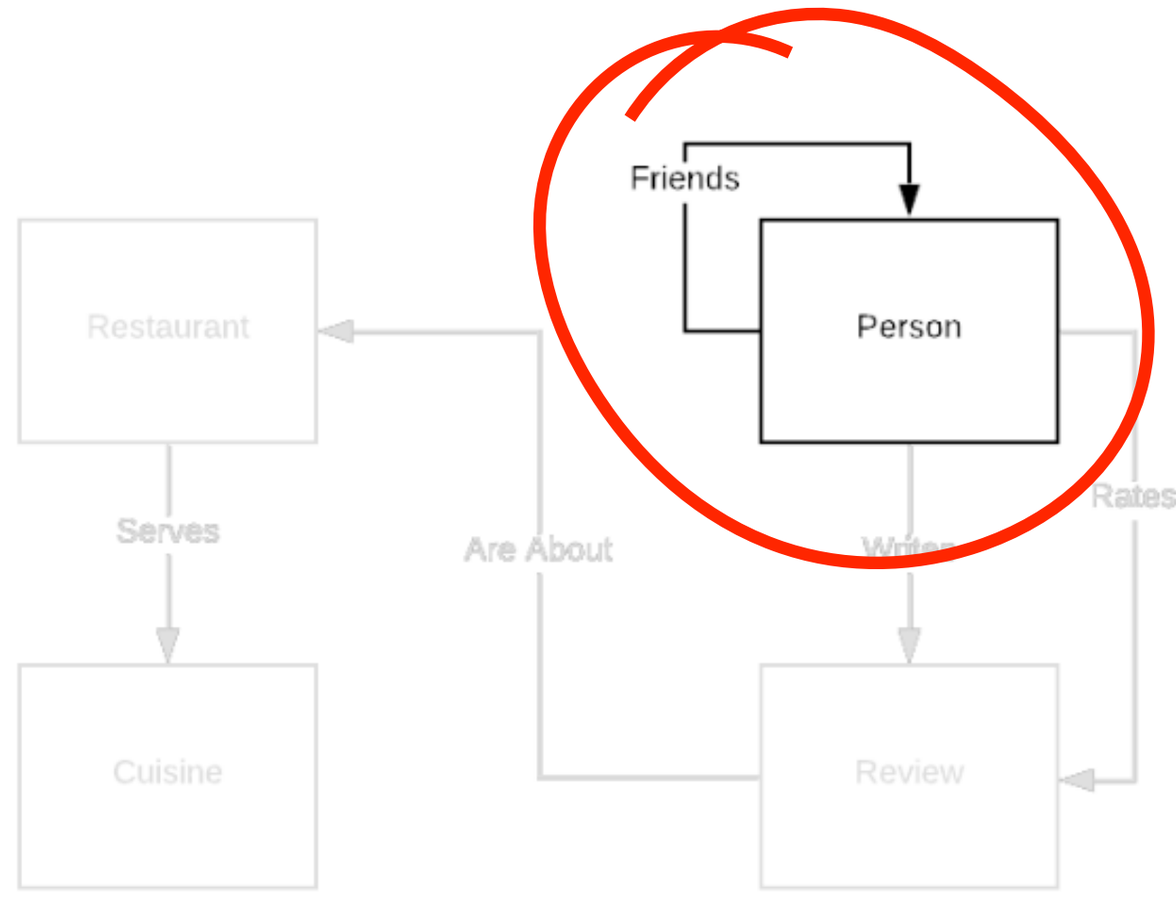
- Finding the relationships
- Naming the edge labels



Step 3: Create a logical data model

TRANSLATE RELATIONSHIPS TO EDGES

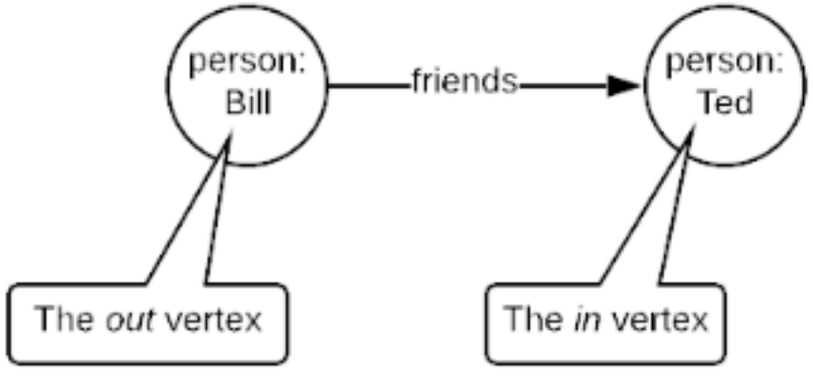
- Finding the relationships
- Naming the edge labels
- Give the edge a direction

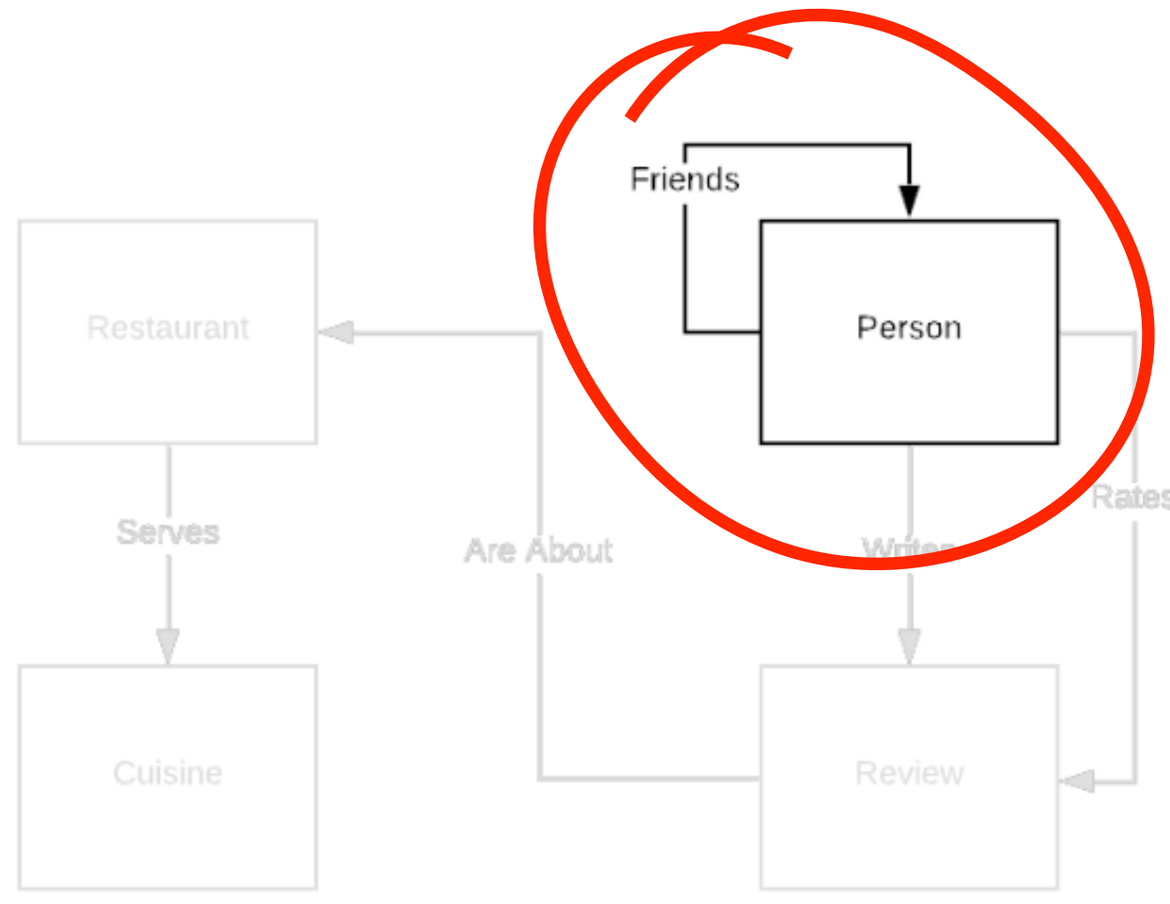


Step 3: Create a logical data model

TRANSLATE RELATIONSHIPS TO EDGES

- Finding the relationships
- Naming the edge labels
- Give the edge a direction

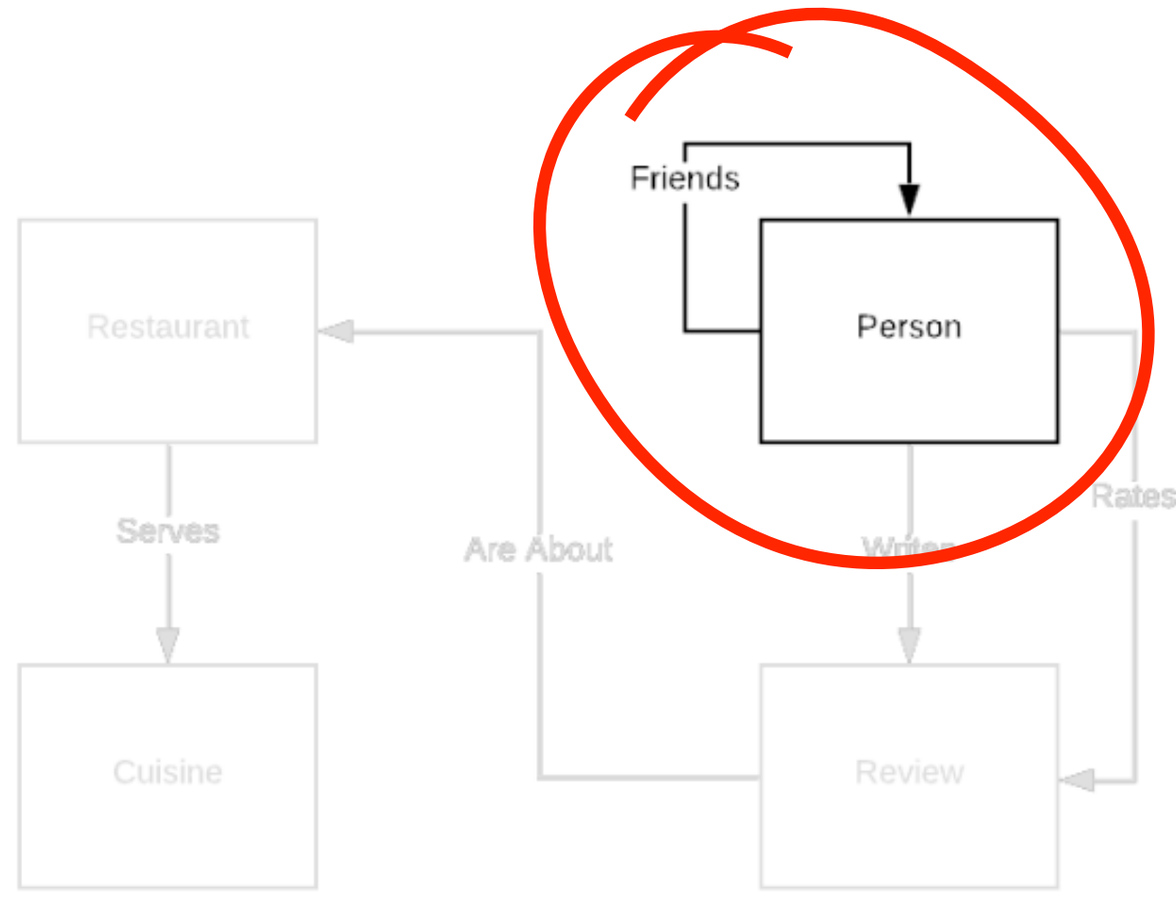




Step 3: Create a logical data model

TRANSLATE RELATIONSHIPS TO EDGES

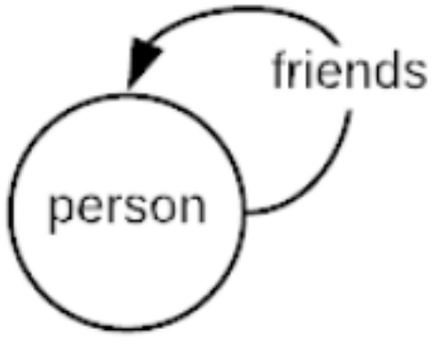
- Finding the relationships
- Naming the edge labels
- Give the edge a direction

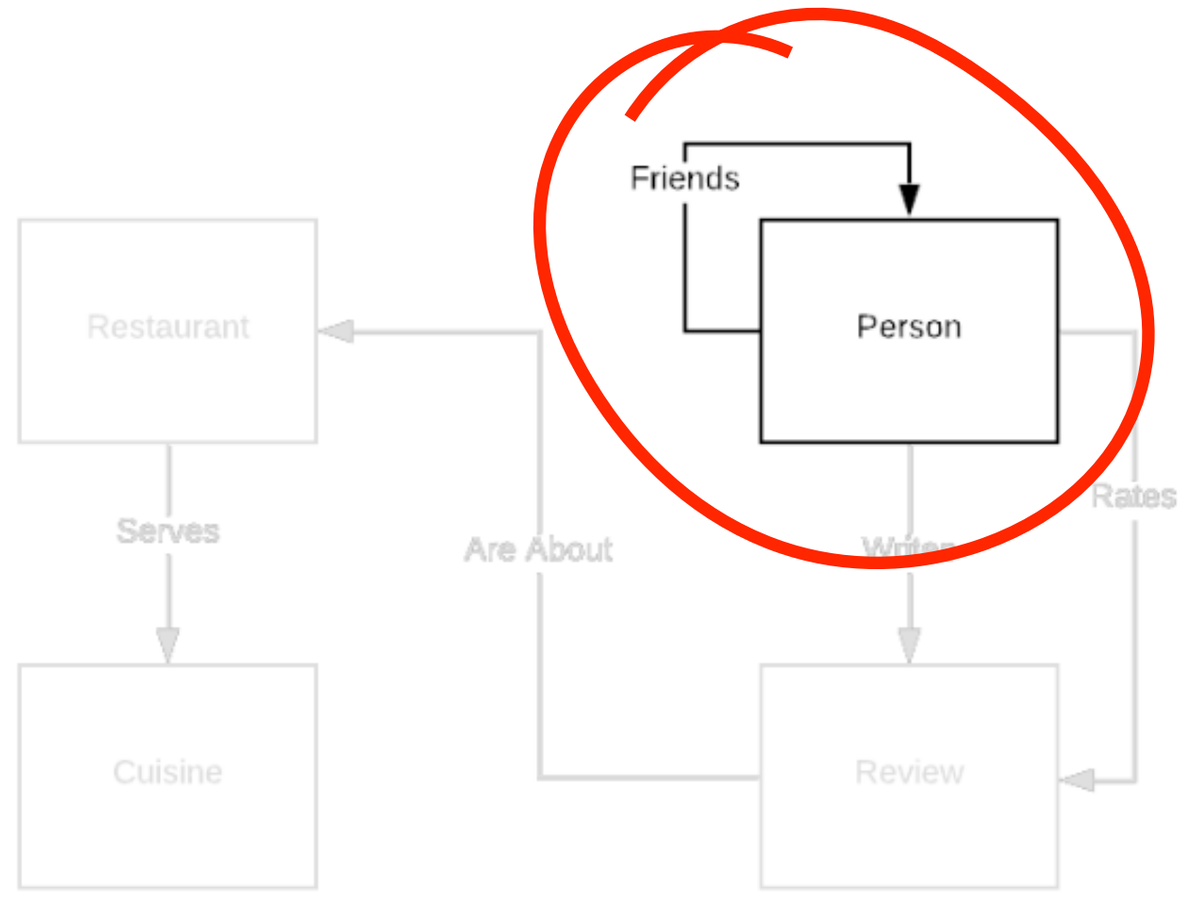


Step 3: Create a logical data model

TRANSLATE RELATIONSHIPS TO EDGES

- Finding the relationships
- Naming the edge labels
- Give the edge a direction

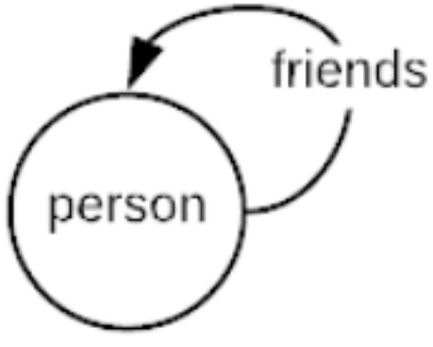


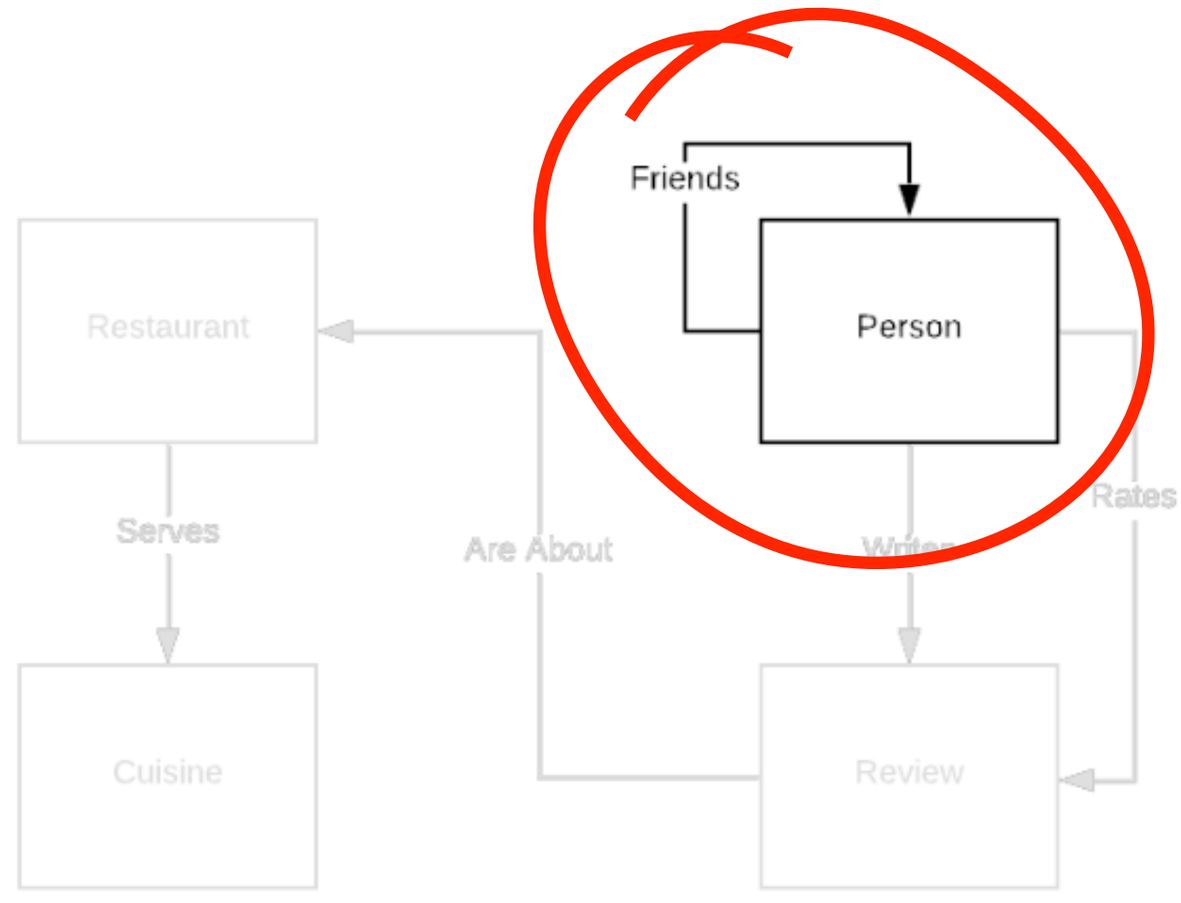


Step 3: Create a logical data model

TRANSLATE RELATIONSHIPS TO EDGES

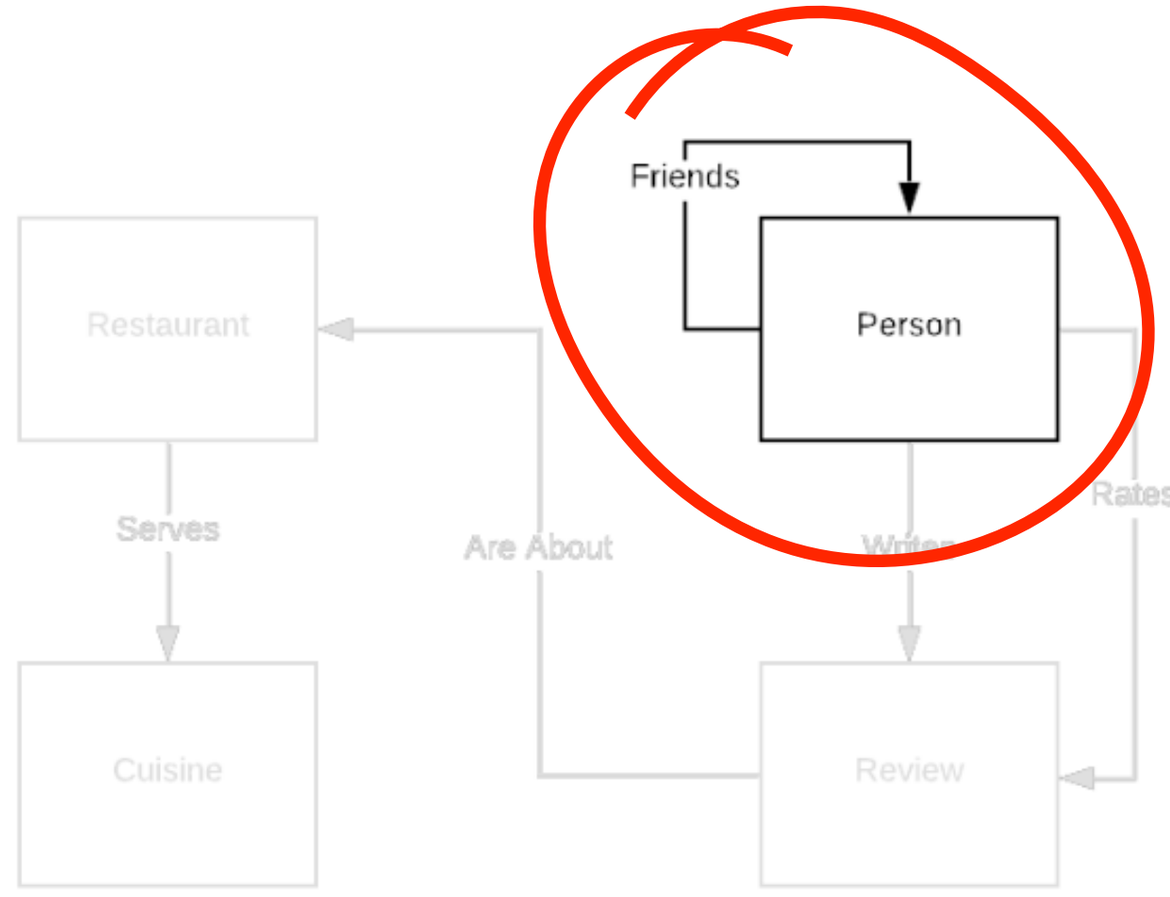
- Finding the relationships
- Naming the edge labels
- Give the edge a direction
- Determine edge uniqueness





Step 3: Create a logical data model

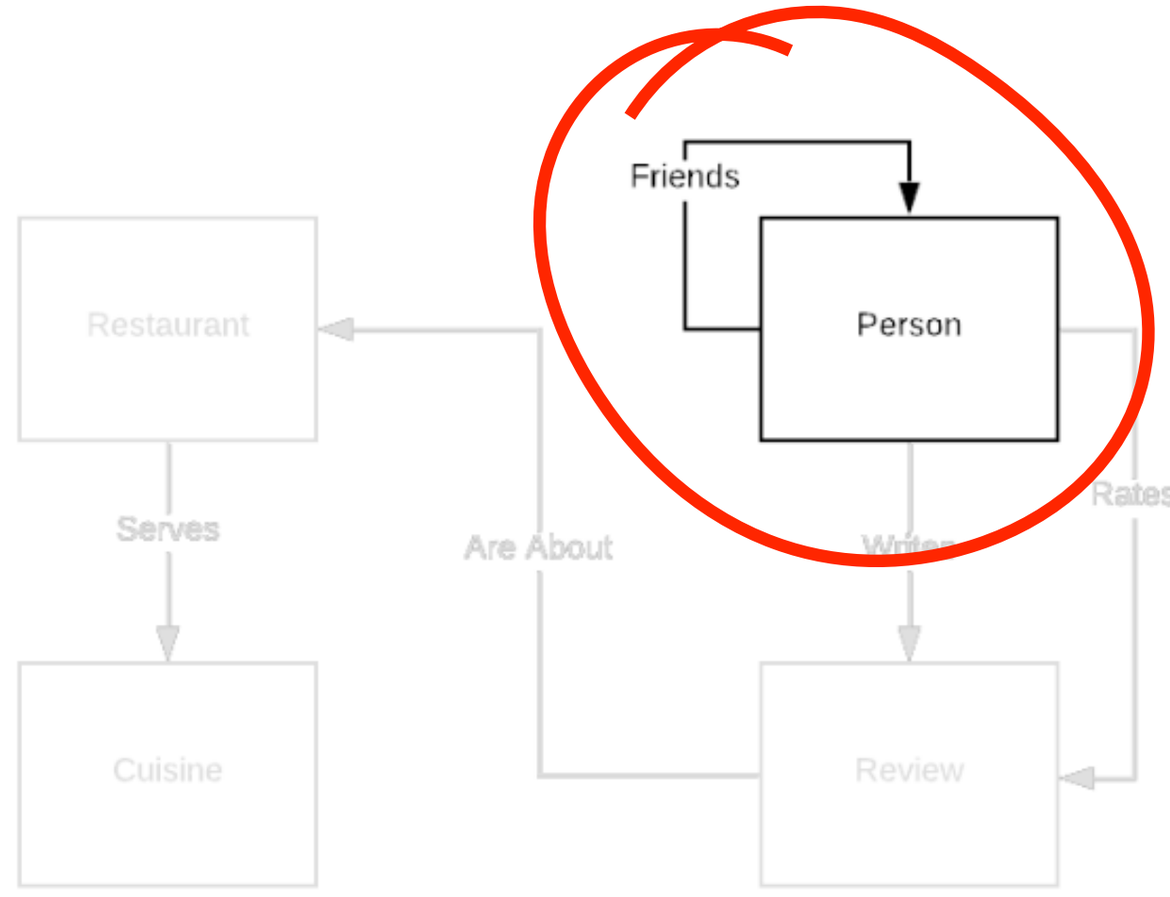
FIND AND ASSIGN PROPERTIES TO VERTICES AND EDGES



Step 3: Create a logical data model

FIND AND ASSIGN PROPERTIES TO VERTICES AND EDGES

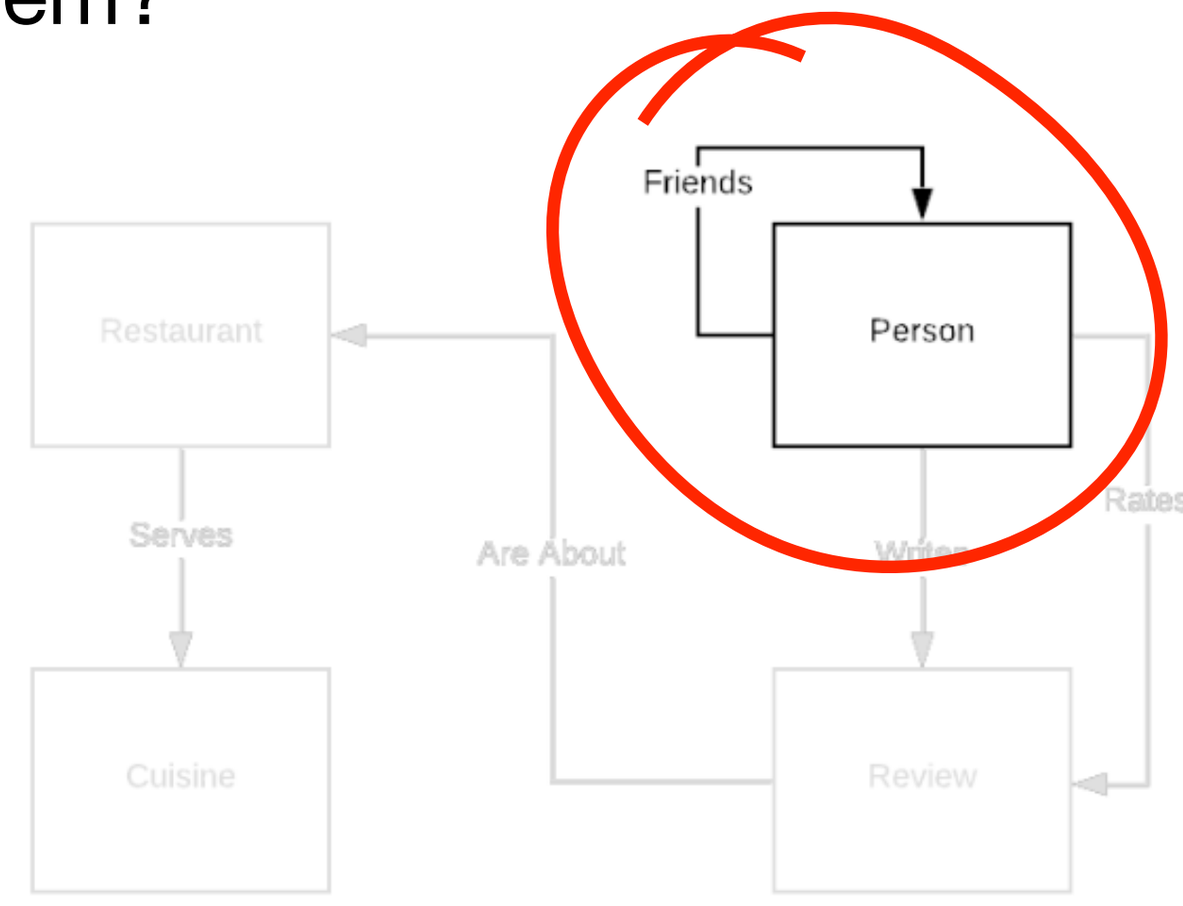
- What properties are required?



Step 3: Create a logical data model

FIND AND ASSIGN PROPERTIES TO VERTICES AND EDGES

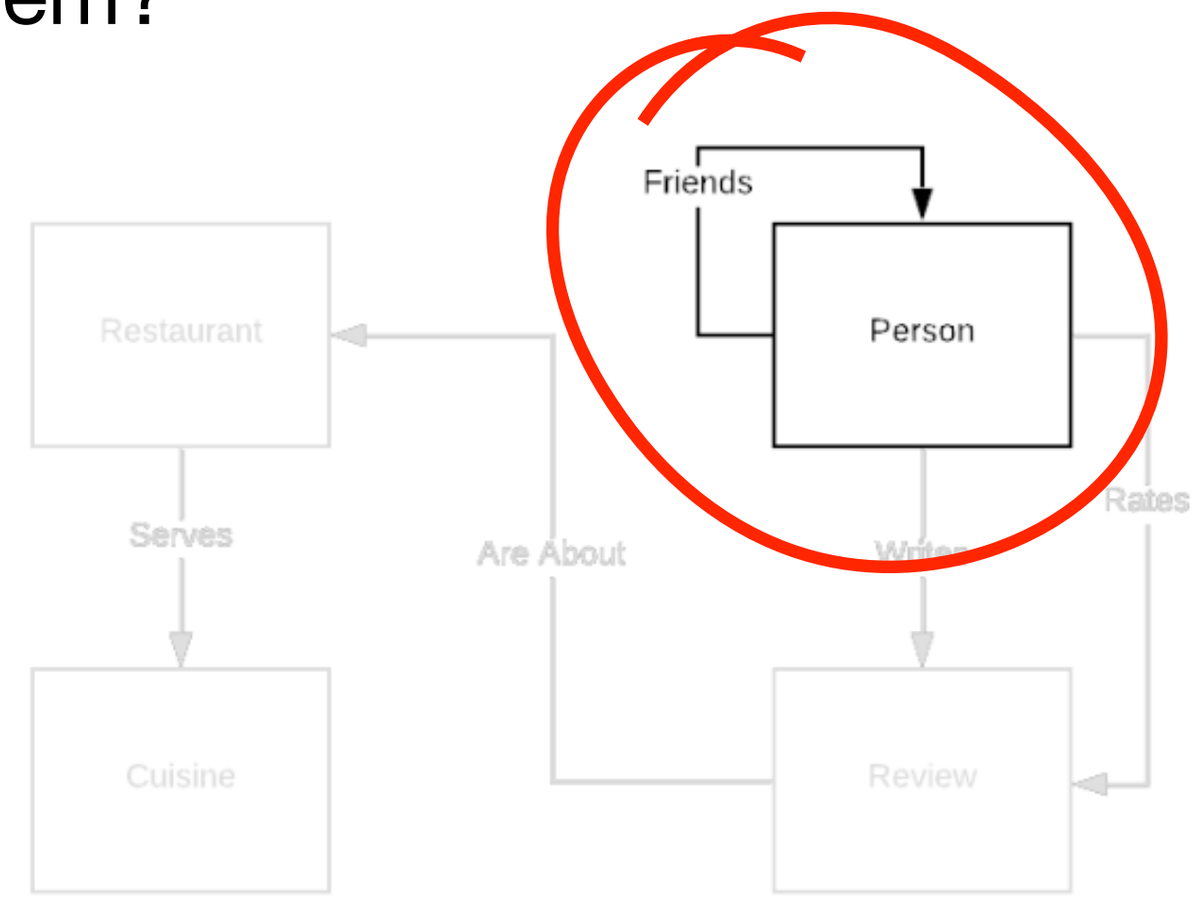
- What properties are required?
- How we are going to name them?



Step 3: Create a logical data model

FIND AND ASSIGN PROPERTIES TO VERTICES AND EDGES

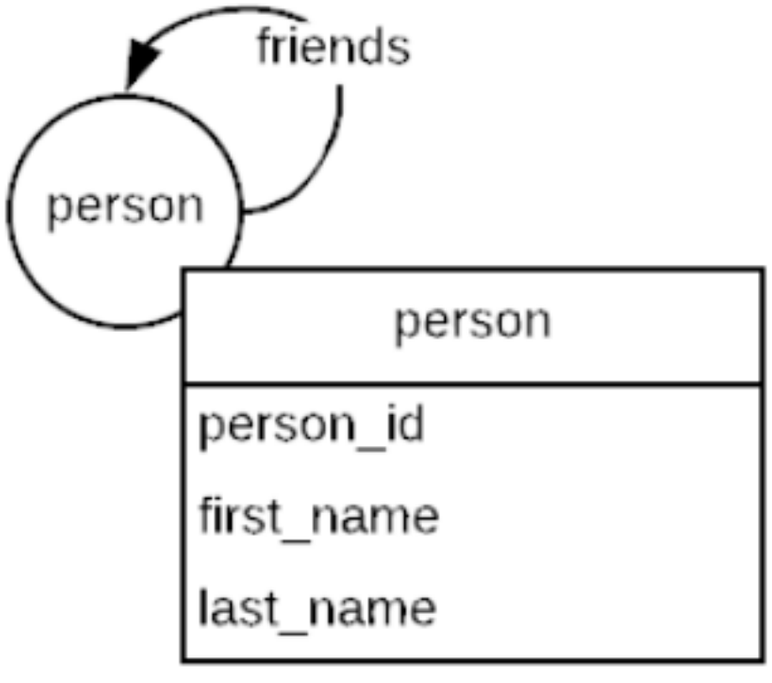
- What properties are required?
- How we are going to name them?
- What is their data type?

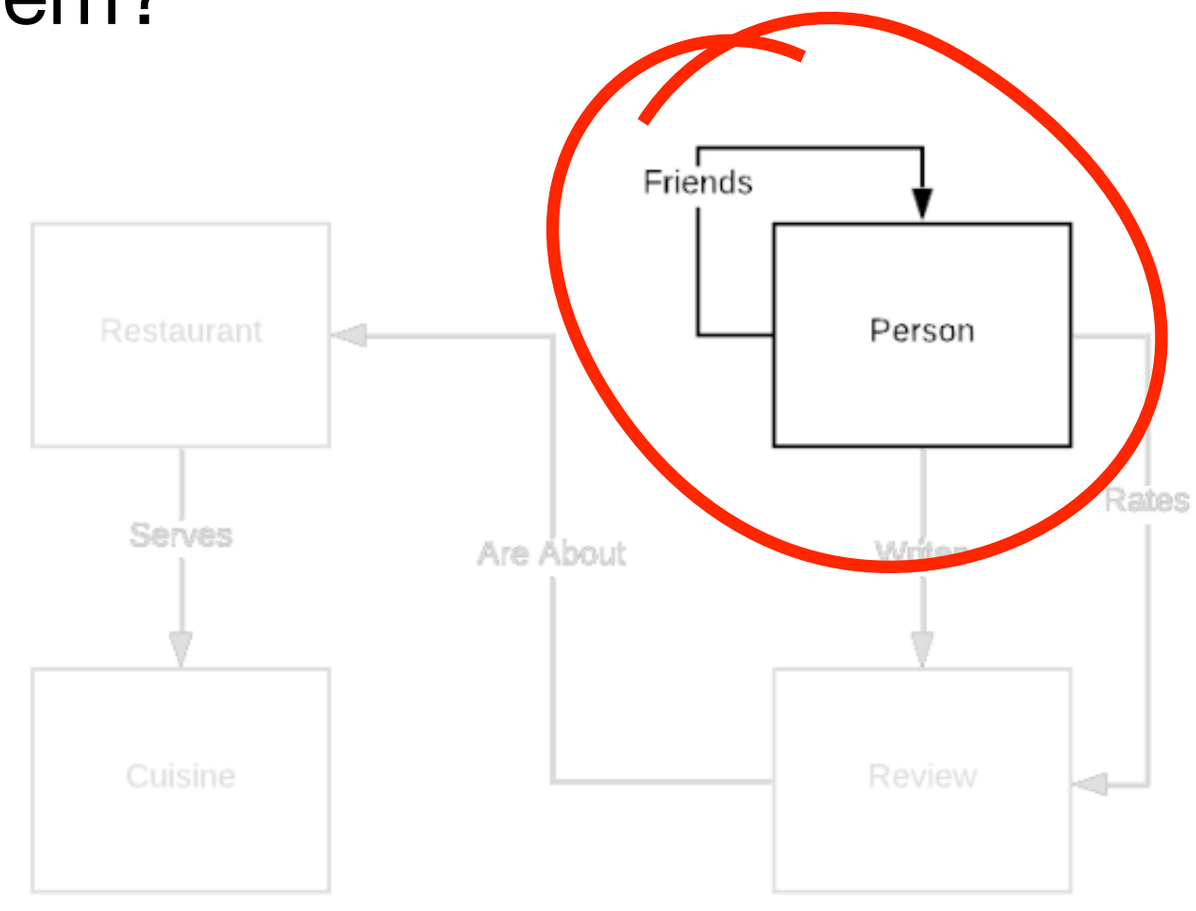


Step 3: Create a logical data model

FIND AND ASSIGN PROPERTIES TO VERTICES AND EDGES

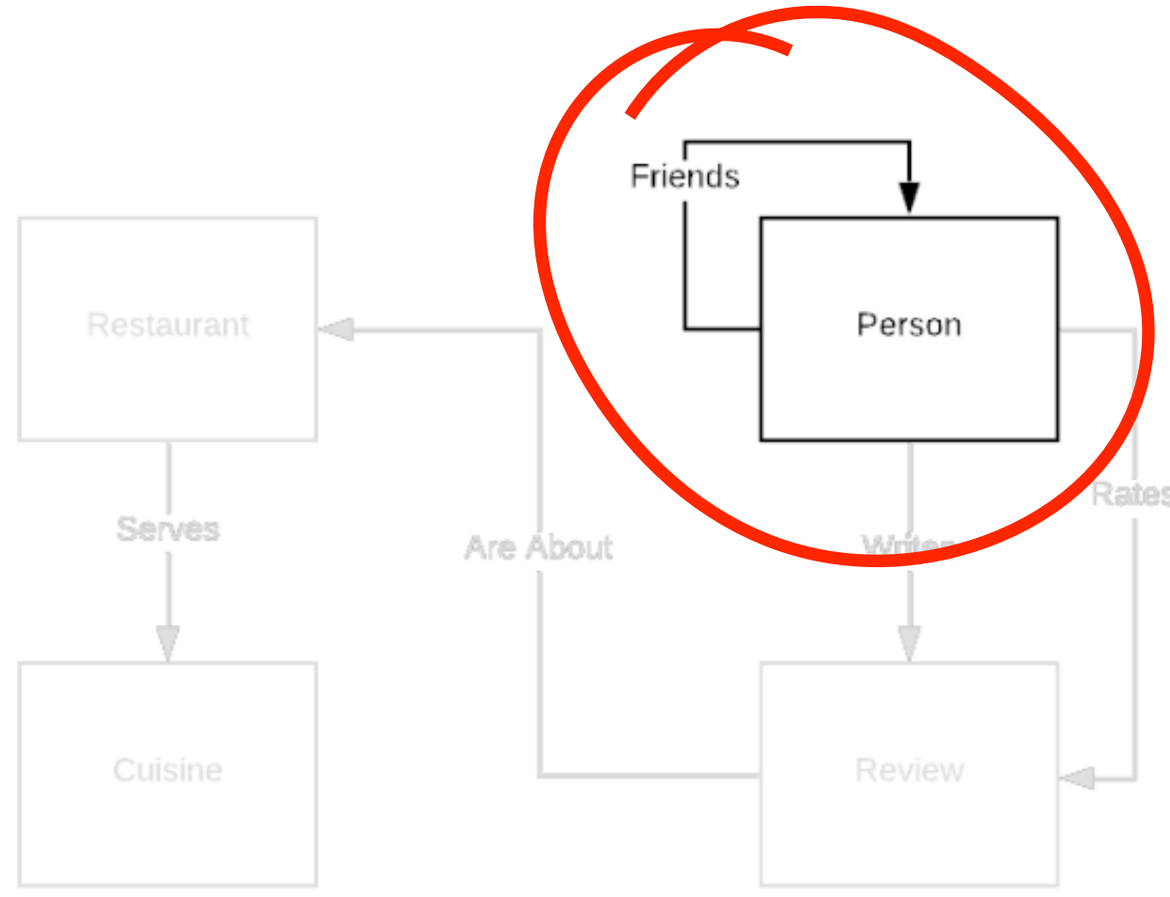
- What properties are required?
- How we are going to name them?
- What is their data type?





Step 4: Test the model

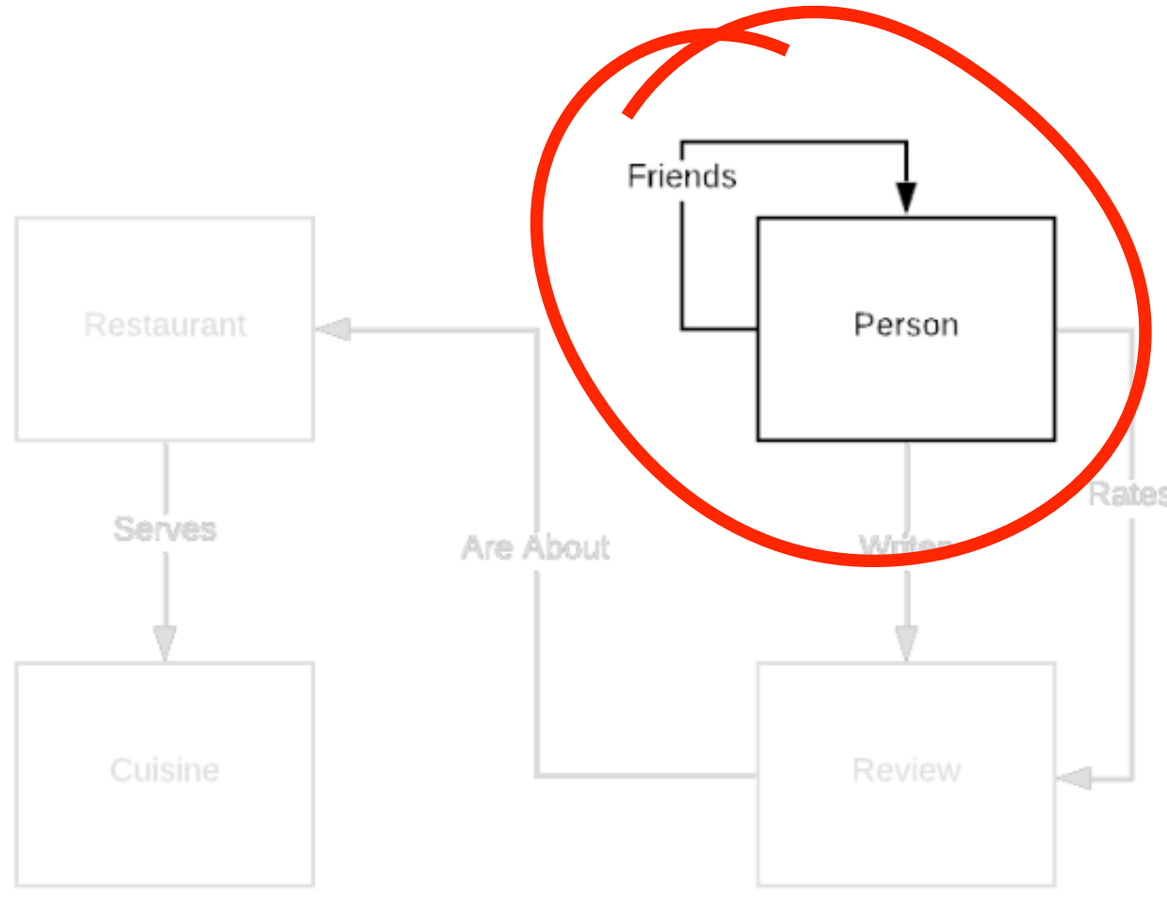
CAN WE ANSWER ALL OUR QUESTIONS?



Step 4: Test the model

CAN WE ANSWER ALL OUR QUESTIONS?

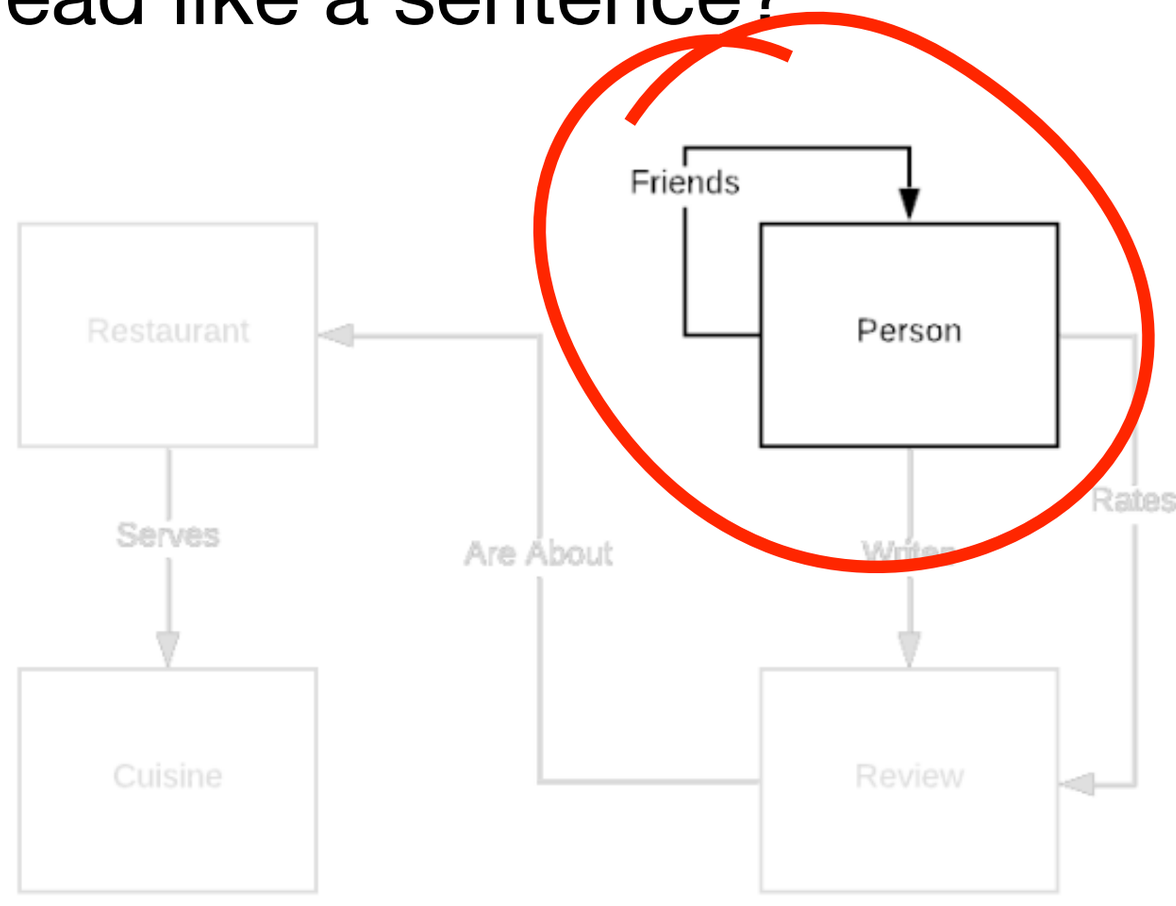
- Can we answer all our questions?



Step 4: Test the model

CAN WE ANSWER ALL OUR QUESTIONS?

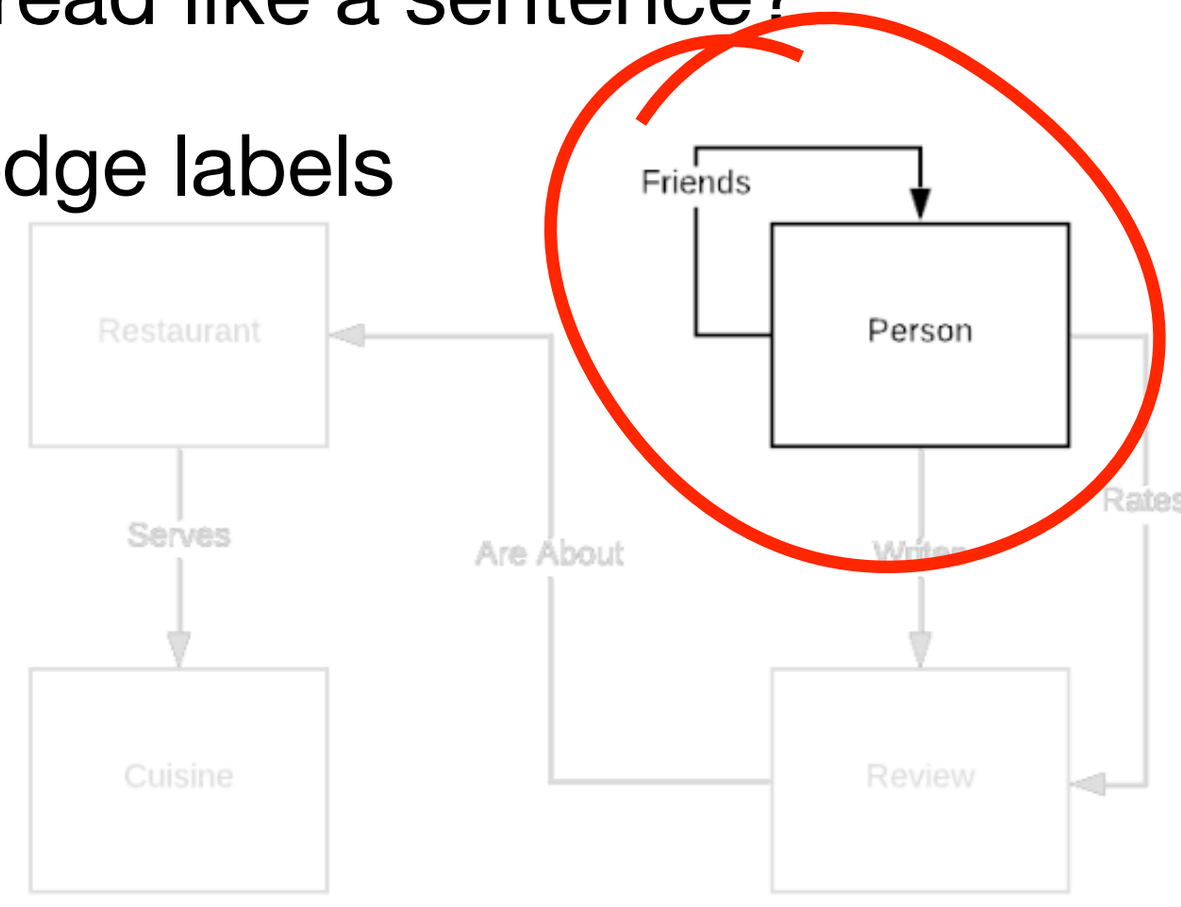
- Can we answer all our questions?
- Do the vertices and relations read like a sentence?



Step 4: Test the model

CAN WE ANSWER ALL OUR QUESTIONS?

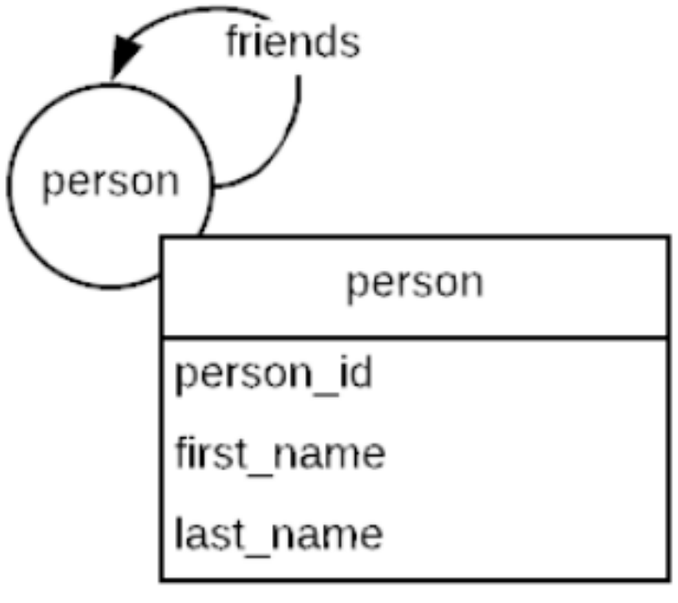
- Can we answer all our questions?
- Do the vertices and relations read like a sentence?
- Do I have different vertex or edge labels with the same properties?

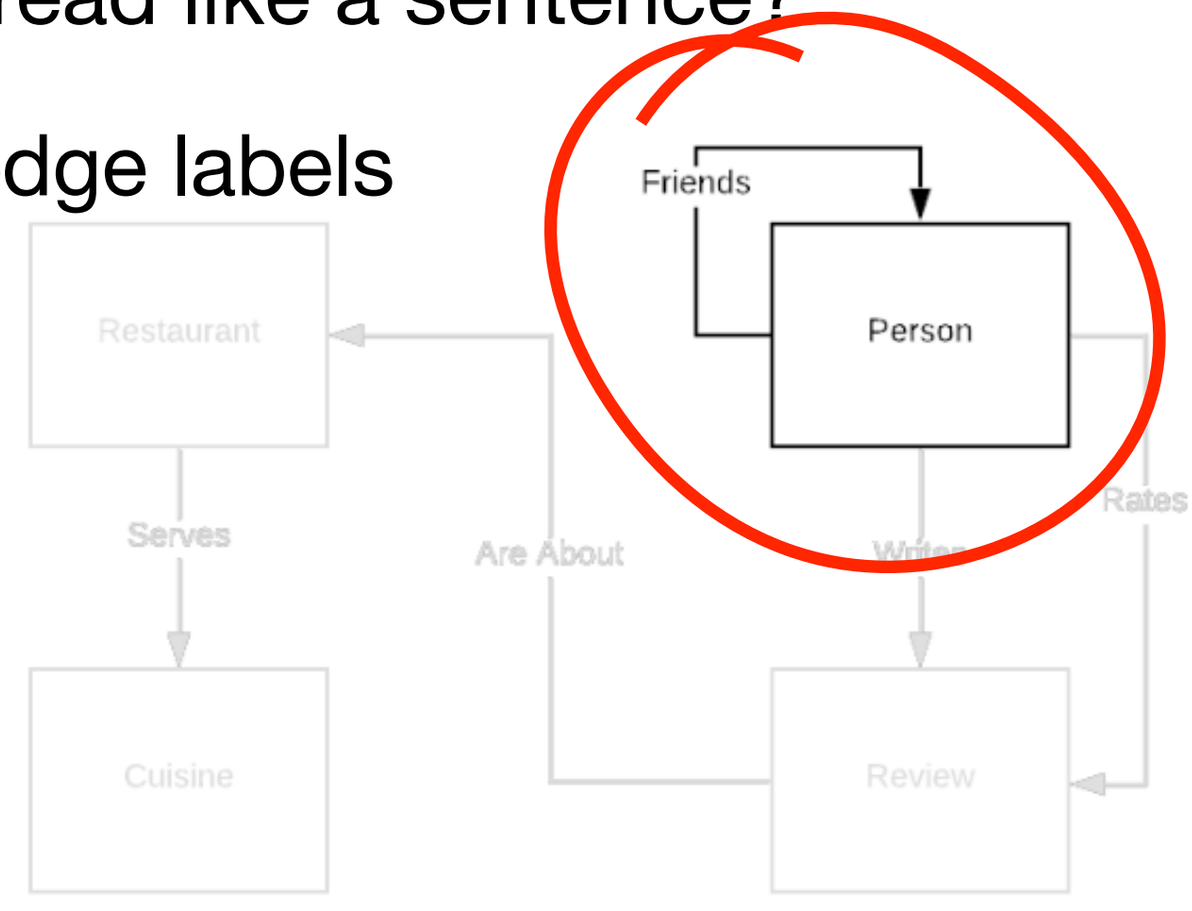


Step 4: Test the model

CAN WE ANSWER ALL OUR QUESTIONS?

- Can we answer all our questions?
- Do the vertices and relations read like a sentence?
- Do I have different vertex or edge labels with the same properties?





Bibliography

- [Bec] Dave Bechberger, Josh Perryman, *Graph Databases in Action*, Manning Publications, 2020