**11-13 notes**

Data Modeling

Goals:

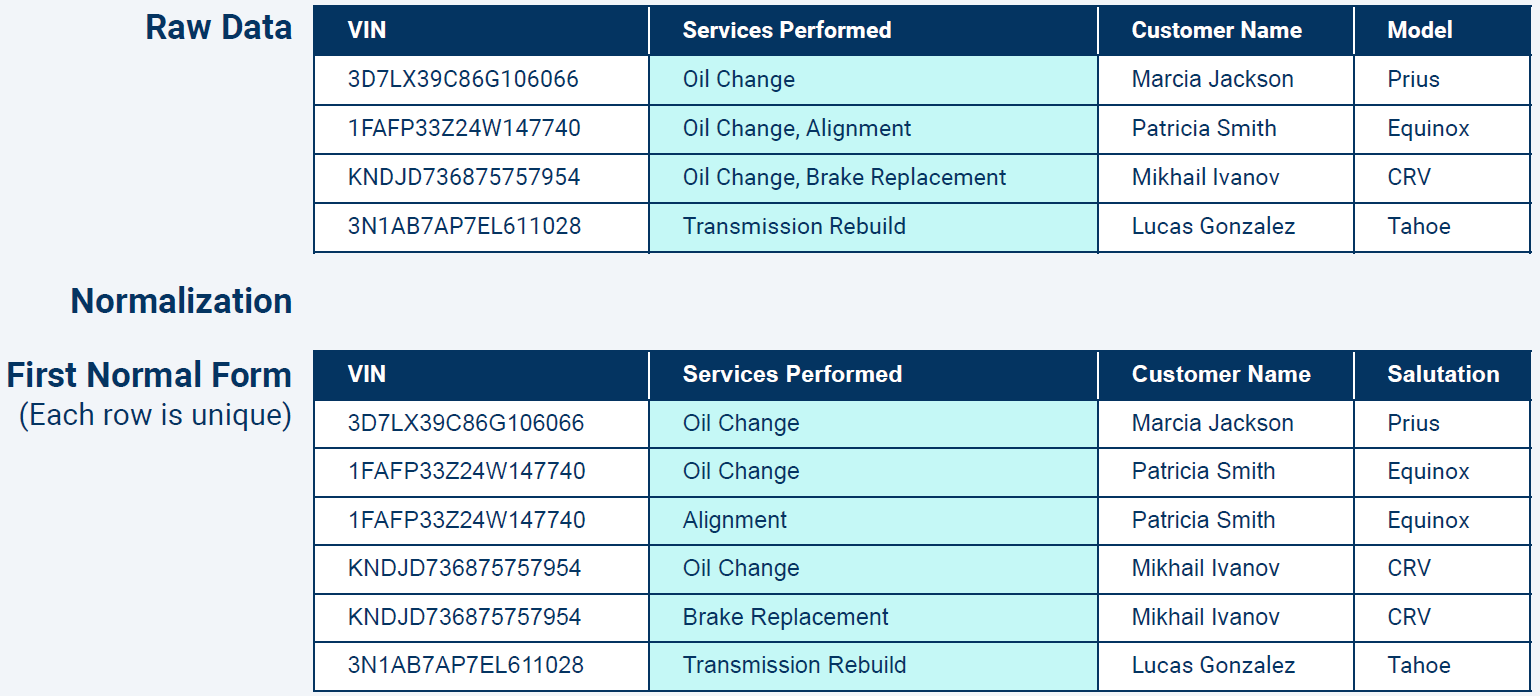
By the end of this lesson, you will be able to:

* Apply data modeling techniques to database design.
* Normalize data.
* Identify data relationships.
* Create visual representations of a database by using ERDs.

Data normalization:

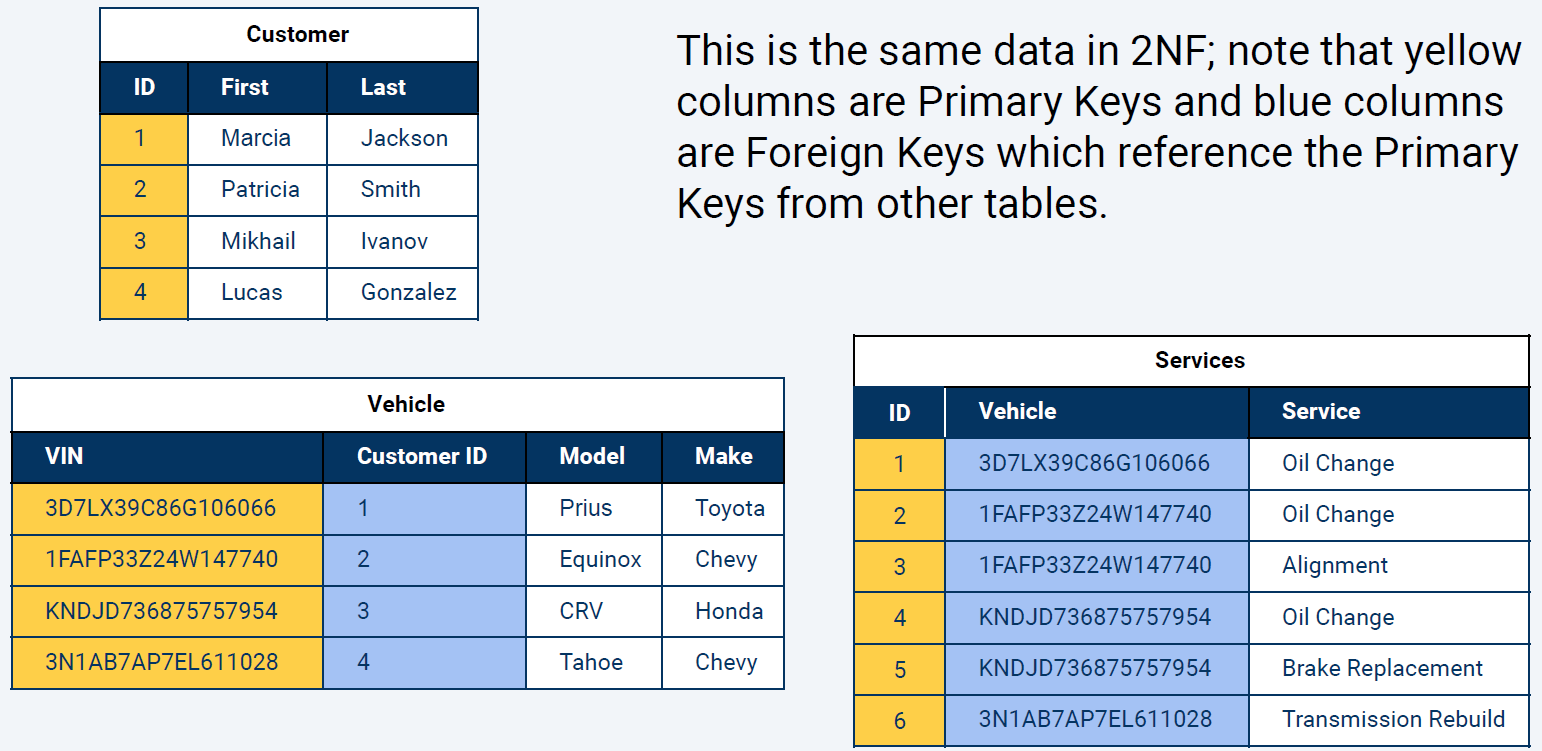
* How to make a database and what the pieces (tables) are.
* “Data normalization is the process of restructuring data to a set of defined “normal forms”, which eliminates data redundancy and inconsistencies.”

First normal form (1NF)- each field in the table row should have a single value.



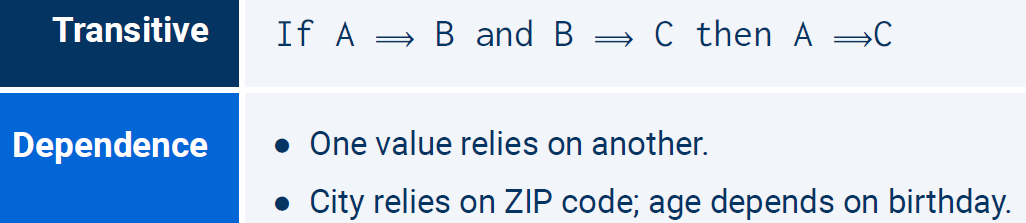
^ Raw data has multiple services in the Services Provided

Second normal form (2NF)- Add a primary key, and any repeat data should go in it’s own table.



^ Vehicle table uses the VIN as the primary key. Customer ID is a foreign key (since it’s from a dif table).

**Transitive dependency** is the reliance of a column’s value on another column through a third column.

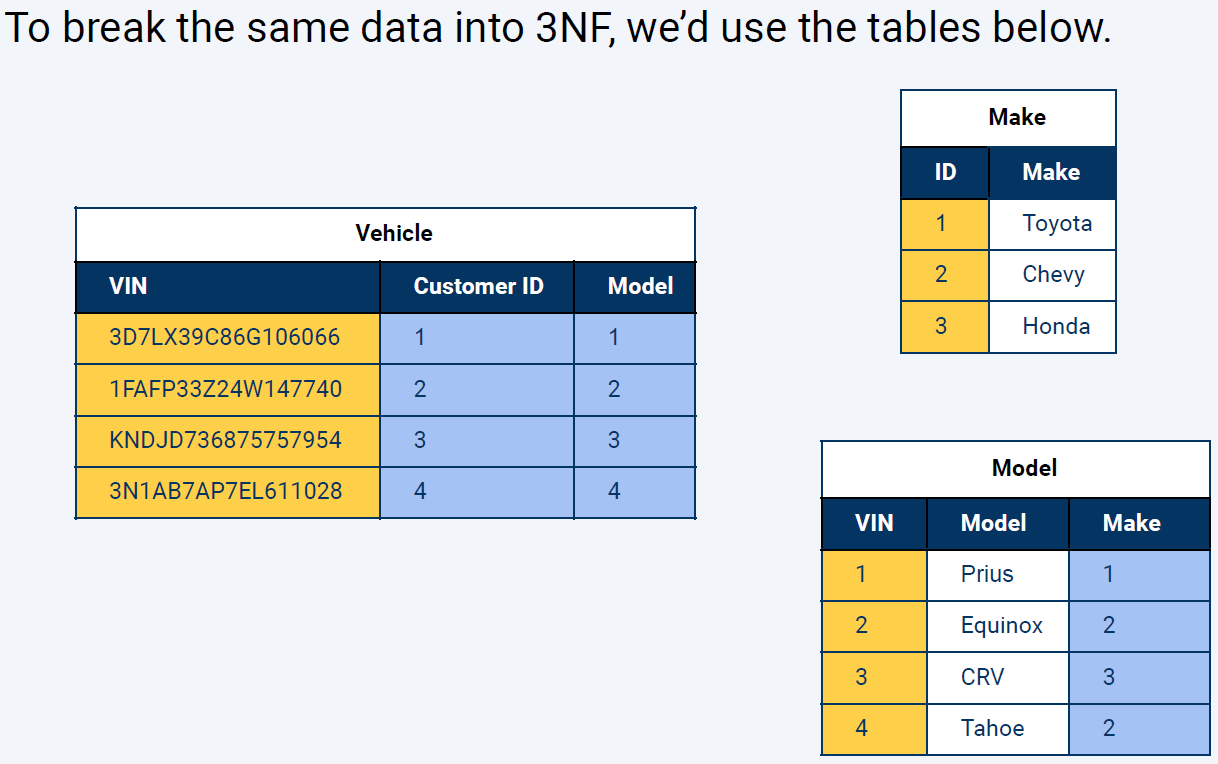
****

**For example:** Consider the VIN, Customer ID, Model, and Make columns in the Vehicle table.

* Customer ID depends on VIN.
* Model depends on Customer.
* Make thus depends on Model.

Third normal form (3NF)- simplifying the relationships

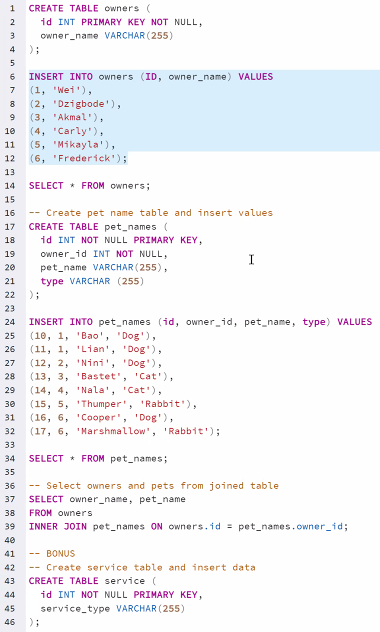
To transition to 3NF, data must be in 2NF. Additionally, no column can imply another column in the same table. For instance, “Make” is implied by “Model” in the table below. That is, a model “Prius” will always have a make of “Toyota”.

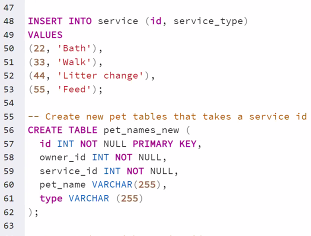


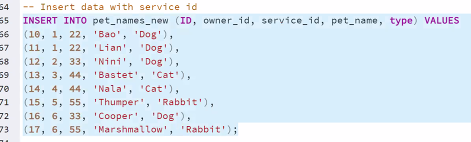
^ Vehicle will refer to Model table, Model table will refer to Make table

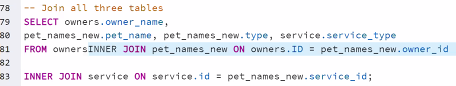
^ This is how you break the database down into the smallest referenceable units possible.

**pets\_db Solution:**







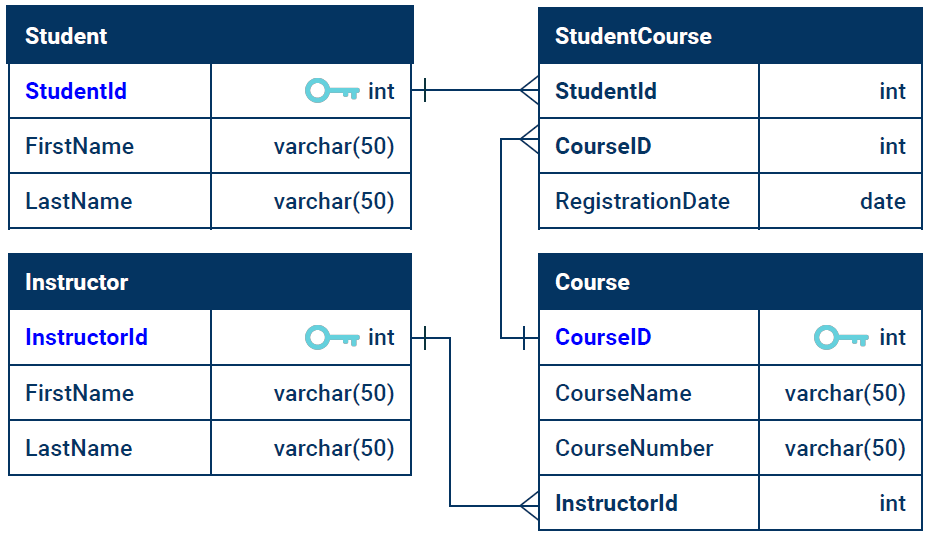


A Lookup table or Junction table is the table with the unique information. In this case, it’s the joined table with all 3 unique IDs

A Primary key can be used as a foreign key in another table

**Data relationships-**

One to one, one to many, many to many.



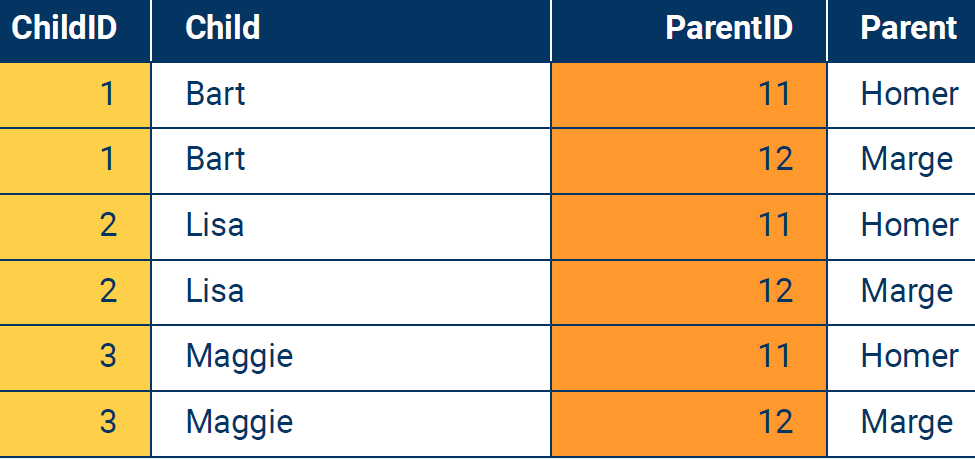
^ lines on left outside of table mean 1, little triangles on right mean many

Examples-

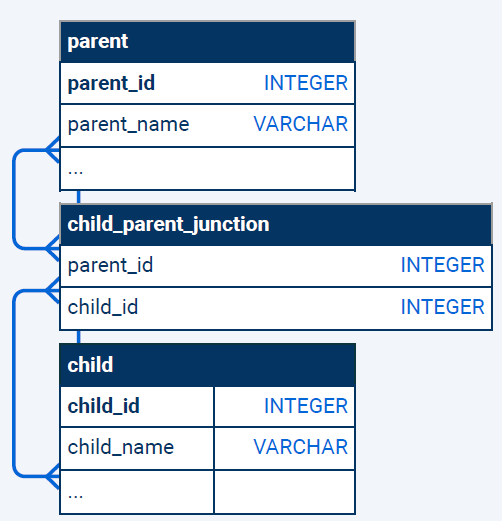
1 to 1- Social security number and a person

1 to many- Multiple people living at the same address

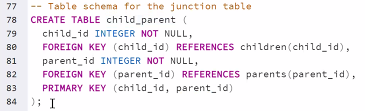
Many to many- A child has multiple parents, and a parent can have multiple children



^ table above is a junction table.

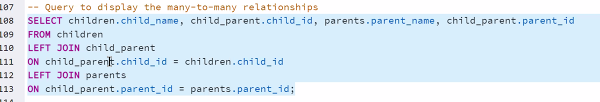


^ middle table above is a junction table. It’s a combination of Parent and Child tables



^ Line 80 is making a foreign key from children table’s child\_id column.

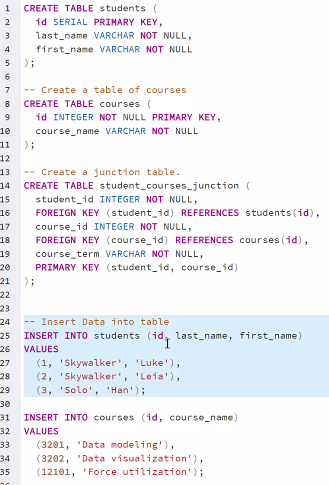
^ Line 83 is saying that the primary key is two columns in the table, and must be unique. No repeats.



Another way of writing the code above is:



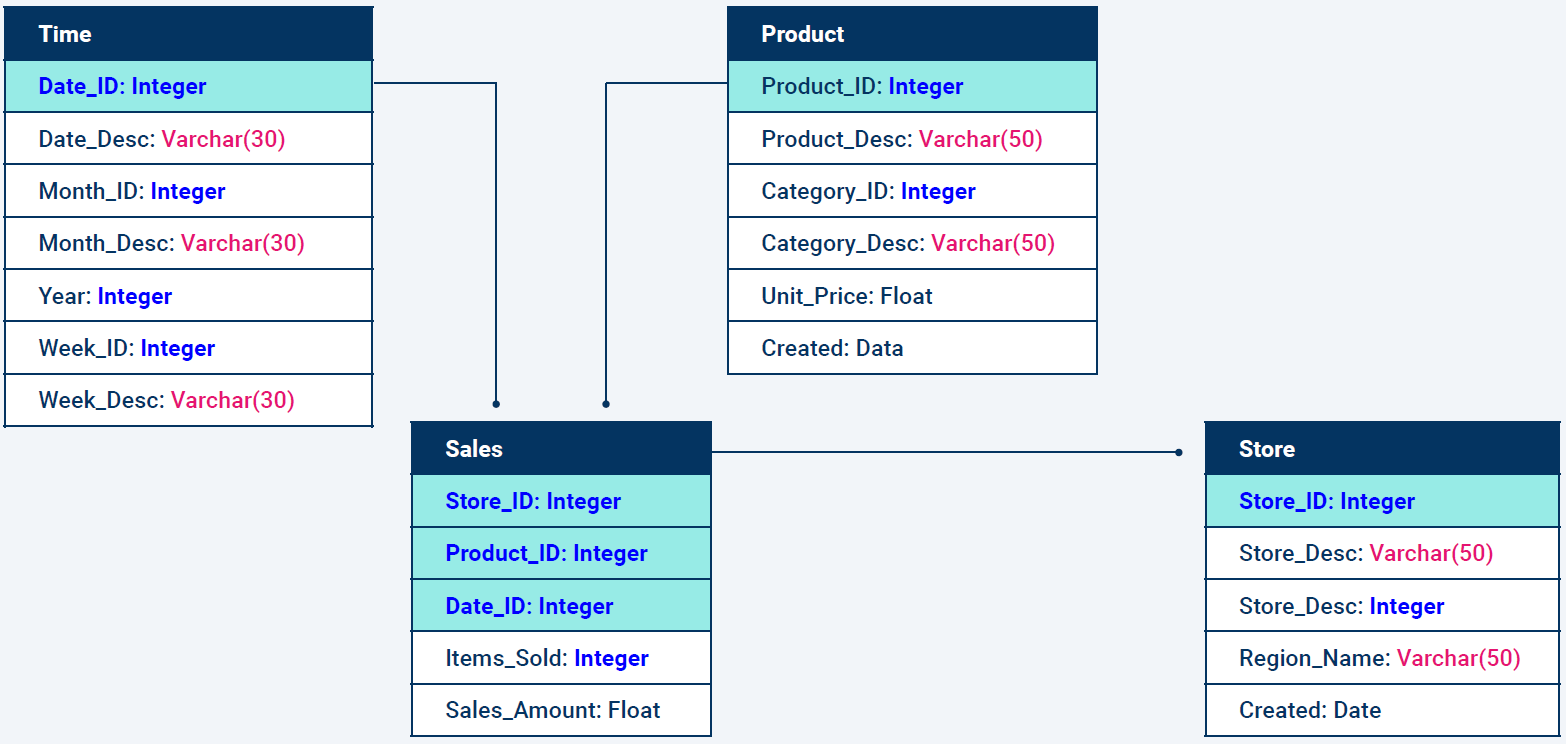
**Data Relationships Solution:**



^in example above, you’d populate the tables with data before you create a junction table

Serial and Identity both increase incrementally as you add more records to a database

ERDs- Entity Relationship Diagram



^ above is most complex. It’s a Physical Model Design, and has the input parameters.

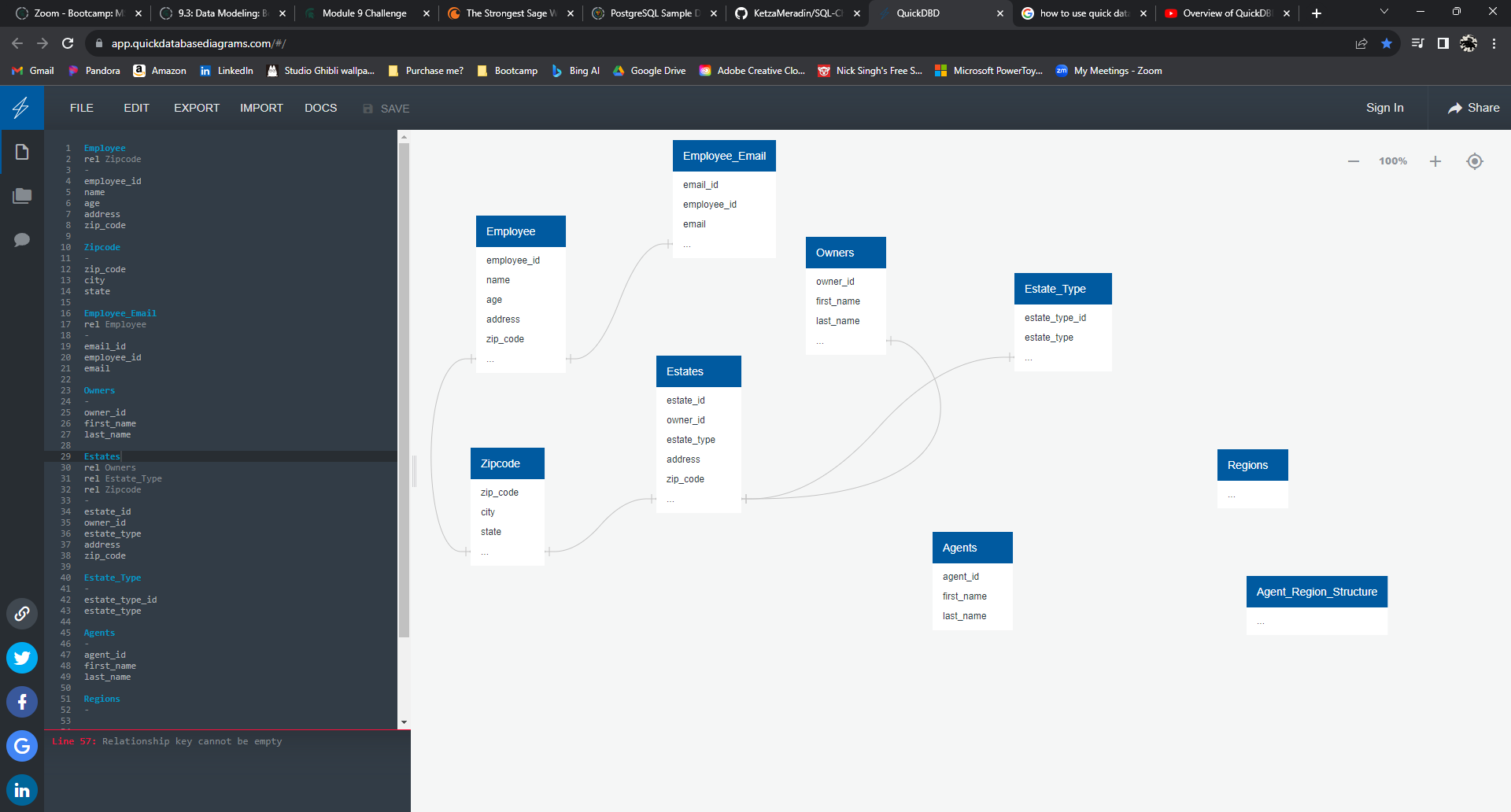
Logical Model Design would have just the column names

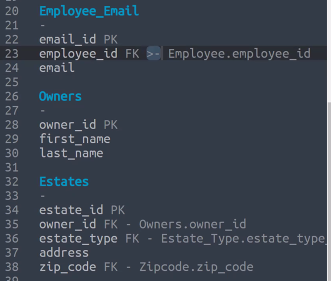
Conceptual Model Design would have just the table names and lines.

Making ERDs is a common task for Database Admins (DBAs).

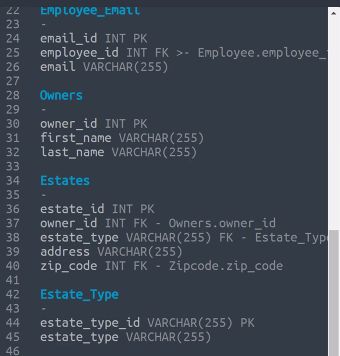
**QuickDatabaseDiagrams.com example (9pm in recording):**

Type in the left to get the diagram on right. You can drag and drop boxes on right as needed.





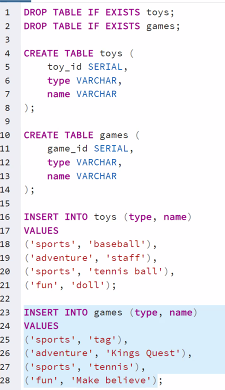
^ Line 23- the >- means many to one



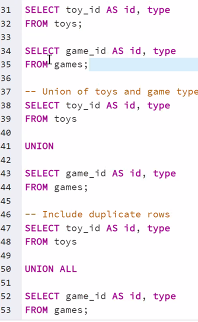
^ This code is the final one for the activity/demo.

Unions are another way to combine data from multiple tables without using joins.

Unions will not repeat columns where the values are the same.



^ Making the tables



^ Actually using Union.

