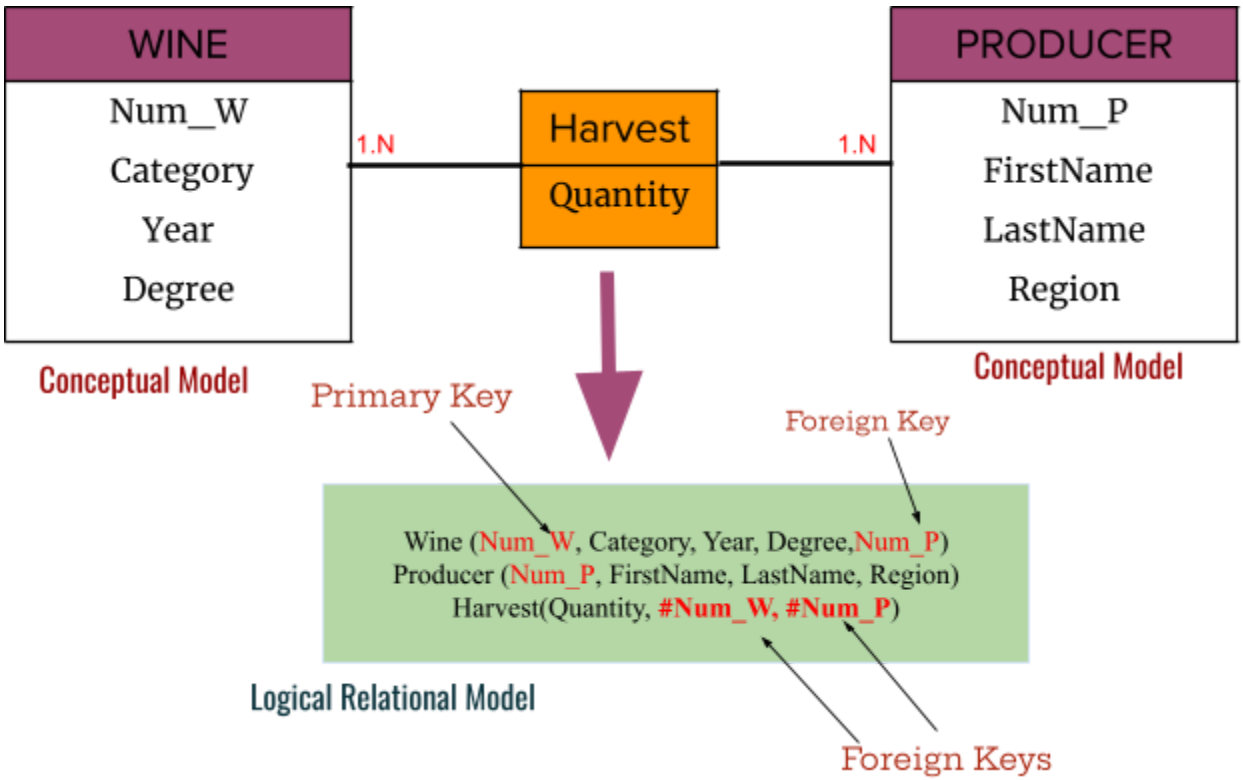
Relational Model, And It's Implementation Using SQL

x- -Chizey_oF_CederWood

Transitioning from a Conceptual Entity-Relationship (ER) Diagram to a Relational model, implement it using SQL



Entities

- **Wine (Num_W, Category, Year, Degree,Num_P)**
- **Producer (Num_P, FirstName, LastName, Region)**
- **Harvest(Quantity, #Num_W, #Num_P)**

Attributes

- For the Wine Entity: Num_W [Unique Identifier] Category, Year, Degree (Descriptional Details)
- For the Wine Producer: Num_P [Unique Identifier] FirstName, LastName, Region (Descriptional Details)
- For the Harvest Entity: **#Num_W, #Num_P** [Unique Identifier] Quantity (Descriptional Details)

Relationship

A new table, **Harvest**, is created to manage the One -to-many relationship between Wine and Producer. This table includes

- The relationship '**Harvest**' is represented in the relational model by including **Num_W and Num_P** as a **foreign key** since they are both from the **Wine** and **Producer** table respectively.
- **Num_W and Num_P** as foreign keys from the **Wine** and **Producer** tables respectively, which together act as the composite primary key for the **Harvest** table.
- Additional attributes from the **Harvest** relationship, which is Quantity, is included to provide context for each Query.
- Each **Wine** entry in the **Wine** table has a reference (foreign key) to a **Producer** from the **Producer** table. This establishes which producer is responsible for each **Wine**.
- A **Producer** can produce many different wines, but each **Wine** is produced by only one **Producer**.