Jack T. Neely

Hello, my name is Jack Neely. I’m a database designer. Here, I am showcasing my database development skills by walking through the process of designing Entity Relationship Diagrams (ERDs), as well as developing a database in MySQL and testing with queries.

My first project is for a local business called “Nora’s Bagel Bin.”

First, we import data from the bagel order form, which was provided for this project.

A picture containing table

Description automatically generated

Next, we transform this data into First Normal Form (1NF).  
For this project, the transformation was provided.

A picture containing table

Description automatically generated

As you can tell, there are no repeating values in this table and a primary key has been defined.

Following this, we now transform the data to Second Normal Form (2NF).

A screenshot of a computer

Description automatically generated with medium confidence

As you can see, all non-prime attributes are separated by their functional dependencies.   
In other words, primary keys related to other attributes have been separated into their own tables.

You may notice the practicality of this standard from a business owner’s perspective.  
It can sometimes be easier to examine data that has undergone categorization.

Note the cardinality representations (1:1 for one-to-one, and M:1 for many-to-one).

Each Bagel Order is unique, and the primary key Bagel Order ID is used as a Foreign Key in Bagel Order Line Item. Bagels are also unique, and there can be many bagels in each bagel order line item, as described by Bagel Quantity.

Sequentially, we transform to Third Normal Form (3NF).  
This allows us to improve consistency by removing transitive dependencies for non-prime attributes.  
Diagram

Description automatically generated

Customer details are separated from customer orders to preserve functional dependence and provide lossless data management. Each Customer Order contains one Foreign Key, Customer ID, which is referenced by the Primary Key in Customer Details. We notice here that the cardinality established is Many to One, as a given customer can place many orders.

Finally, we transform the dataset into a working model schema.  
This involved specifying datatypes and formatting value names and attribute names.

A screenshot of a computer

Description automatically generated with low confidence

That’s it. We have the dataset normalized into 3NF and transformed into a usable set of tables.  
We can perform queries on this, insert into it, create indexes, perform joins, etc.

REQUIREMENT B

The second project is to create a database for the business “Jaunty Coffee Co.”

We can start by creating the database, tables, values, and attributes:

Text

Description automatically generated

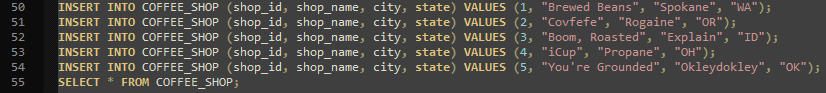
For this project, I’m using MySQL Workbench.  
  
We can execute this selection of the code, and it returns the following:

Graphical user interface, text, application, Word

Description automatically generated

The execution passes.

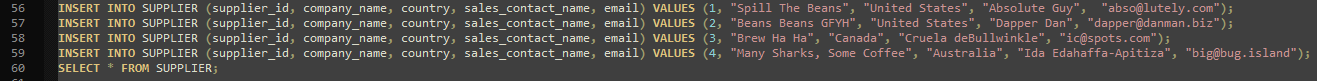
Next, we will populate each table with dummy data using INSERT INTO statements:



And execute in MySQL Workbench to view results:  
Graphical user interface, application

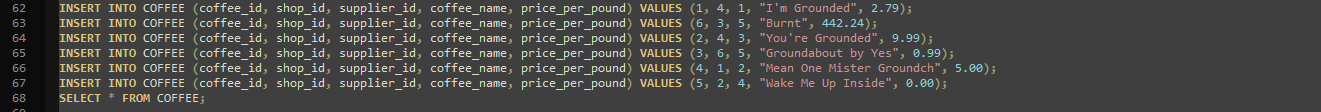
Description automatically generated with medium confidence

Continued:



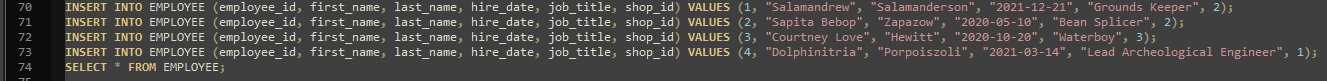
Graphical user interface, text, application, email

Description automatically generated



Graphical user interface, text, application, email

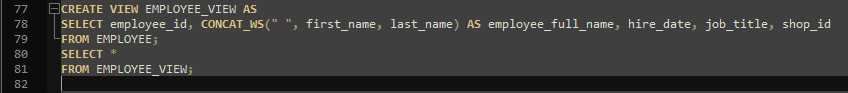
Description automatically generated



Graphical user interface, text, application, email

Description automatically generated

Now that we have populated our tables and queried them successfully, we can create a view.  
For this example, let’s create a view EMPLOYEE\_VIEW from the EMPLOYEE table. We will grab every attribute and perform a concatenation on first\_name and last\_name to create the attribute employee\_full\_name and select to view our results.



Graphical user interface, text, application, email

Description automatically generated

Now let’s create an index on the COFFEE table and test:  

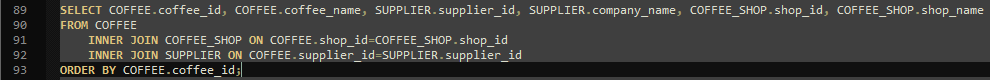

Graphical user interface, application

Description automatically generated

We can perform other queries. Here, we will perform a Select From Where (SFW) query on the COFFEE table with a comparison operand to list all rows where the price\_per\_pound of coffee is equal to or less than $5.00.  


Graphical user interface, text, application, email

Description automatically generated

Lastly for this project, we can join tables. Let’s join COFFEE with both COFFEE\_SHOP and SUPPLIER. We can do this with two separate INNER JOIN statements, then ORDER BY coffee\_id.  


Graphical user interface, text, application, email

Description automatically generated

That’s it for our second project, and that completes our task. We created an Entity Relationship Diagram (ERD) for “Nora’s Bagel Bin” and transformed the original data from unnormalized to First Normalized Form (1NF) to Second Normalized Form (2NF) to Third Normalized Form (3NF) and finally to useable data properly formatted for database use. After that, we create a database for Jaunty Coffee Co. using MySQL. We executed our code in MySQL Workbench to test and output our data.

1. We created an entire relational database,
2. we populated several tables using our own data,
3. we created a view that simplified the name attributes for the EMPLOYEE table,
4. we created an index on the COFFEE table,
5. we performed a Select From Where query on the COFFEE table for price\_per\_pound, and
6. we joined the tables COFFEE\_SHOP and SUPPLIER on the COFFEE table by related attributes, ordered by coffee\_id, and viewed our final result.

We accomplished our goals. The PRIMARY KEY (here is that we have inserted into ourselves an index of database management knowledge.) REFERENCES work\_skill(Now we can inner join select data management organizations that can select from where we live our expertise.)  
  
  
Thank you.

­ Text

Description automatically generated