# DAD 220 Module Four Lab Template

## Overview

Begin by doing the following steps in the IDE (Codio):

1. Load the classicmodels data set.
2. Start a new terminal session and run this command: mysqlsampledatabase.sql
3. Write commands to use the classicmodels database and show its tables to verify that you are in the right place.

Then perform the steps below to complete the lab. Manually enter any commands you are asked to write. At the end of each step, replace the bracketed text in this template with your screenshot, response, or both, as indicated. Submit your completed template for grading and feedback. Screenshots should be sized to about one-quarter of a page. Written responses should be in complete sentences. Rename this document by adding your last name to the file name before you submit it.

## Identify Cardinality and Table Relationships

1. **Retrieve employee tuples and identify the number of employees** in San Francisco and New York.
   1. Command for San Francisco: select firstName, lastName, jobTitle, offices.city from employees inner join offices on employees.officeCode = offices.officeCode where state = 'CA'.
   2. Write and run a command to return records from New York on your own.
   3. Validate the completion of this step with a screenshot of these two tables.

A screenshot of a computer

Description automatically generated

1. **Retrieve order details** for orderNumber 10330, 10338, and 10194 and **identify** what **type of cardinality** this represents in the entity relationship model.
   1. Retrieve the order details by running SELECT queries with WHERE clauses against the Orders table.

A screenshot of a computer program

Description automatically generated

* 1. Now, identify what type of cardinality this represents in the entity relationship model.
     1. Reference the Module Four Lab ERD diagram linked in the guidelines and rubric to help identify relationships.

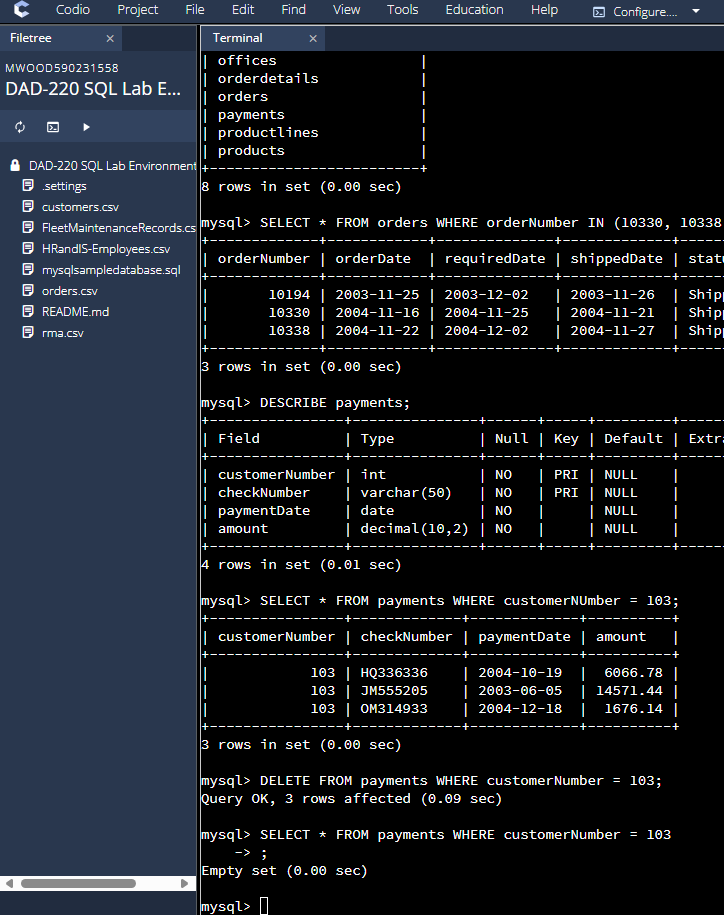
There is no maximum or minimum number of orders per customer, and each order can contain any amount of data. Minimum is one to zero.

1. **Delete records** from the payments table where the customer number equals 103.
   1. Run a DESCRIBE statement to identify fields in the Payments table first.
   2. Select the records from the Payments table for customer number 103 before deleting them.

A screenshot of a computer

Description automatically generated

* 1. Delete the records from the Payments table for customer number 103.
  2. Run a SELECT statement against the table to show that customer number 103 is no longer there.



1. **Retrieve customer records** for sales representative Barry Jones and **identify** if the **relationships** are one-to-one or one-to-many**.**
   1. Remember: SELECT, FROM, INNER JOIN, and WHERE.
   2. Use Barry’s employeeNumber, 1504, and perform a join between the customer salesRepEmployeeNumber to retrieve these records.
      1. Identify whether these entities demonstrate one-to-one or many-to-many relationships.

A screen shot of a computer screen

Description automatically generated

This is a one-to-many relationship due to the fact that each sales rep can have many customers but customers can only have one sales rep.

1. **Retrieve records** for customers who reside in Massachusetts and **identify** **their sales rep and the relationship of entities**. Identify if these entities demonstrate one-to-one or many-to-many relationships.
   1. Remember: SELECT, FROM, INNER JOIN, and WHERE.
   2. Use employee.firstName and employee.lastName in your command.
   3. Identify whether these entities demonstrate one-to-one or many-to-many relationships.

A screenshot of a computer

Description automatically generated

1. **Add one customer record** with your last name using an INSERT statement. You may use the name of a celebrity or fictional character if you don’t use your own name.
   1. You may use the name of a celebrity or fictional character if you don’t use your own name. Think of this as your signature.
   2. Complete these actions to get to the right place to enter this information: (1) Show databases, (2) use classicmodels, (3) show tables, (4) describe customers;
      1. You should now see all of the fields that you’ll need to fill in to complete this step.
      2. Reference your Module Two lab or resources on how to populate these fields if you need to.
      3. Populate the following fields:

customerNumber

customerName

contactLastName

contactFirstName

phone

addressLine1

addressLine2

city

state

postalCode

country

salesRepEmployeeNumber

creditLimit

* + 1. Run a SELECT statement on the Customers table and capture it in a screenshot.

A screen shot of a computer screen

Description automatically generated

1. **Reflection:** Use the lab environment or the screenshots you’ve worked with for this step. Address the prompts below in your reflection. Write a paragraph in response to each prompt:
   1. **Define how cardinality is applied** to the databases you’ve been working with. Explain why different numbers of records were returned from the different offices.

Cardinality describes the bounds (maxima and minima) of relationships and properties inside entities. Using this information, we can make connections between tables and their subtypes. Different offices returned different records depending on which states or cities they were supplied from.

* 1. **Compare and contrast** the different **queries** you ran and how cardinality applies to them.

In part 4, I utilized inner joins as an example of how cardinality applies to various queries. Thanks to his employeeNumber, the statement from part 4 was able to successfully retrieve customer and employee data for Barry Jones. We were able to retrieve his information from both the workers and customers tables because the foreign key in the employees table was employeeNumber and the primary value in the customers table was calledsalesRepEmployeeNumber.

* 1. **Describe two** of the crucial **benefits** **of cardinality** in this type of database.

One major advantage of cardinality that immediately springs to mind is the ease with which it facilitates the establishment of relationships between various supertypes and subtypes in cases when specific types of data are shared across tables. The ability to edit table data is another perk, since we will know exactly which properties are allowed in the tables.