# DAD 220 Cardinality and Targeted Data Template

Replace the bracketed text in this template with your screenshots and responses. Then submit it to the Module Four Lab for submission, grading, and feedback. Screenshots should be sized to approximately 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.

1. **Retrieve employee tuples and identify the number of employees** in San Francisco and New York.

[A screenshot of a computer

Description automatically generated

There are two employees in New York City. There are six employees in San Francisco.

1. **Retrieve order details** for orderNumber 10330, 10338, and 10194 and **identify** what **type of cardinality** this represents in the entity relationship model.

This relationship is a dependency relationship because orders rely on orderdetails to know how much quantity was ordered for one product and the productCode, as well as priceEach for a single item. Without these orderdetails there are no orders to fulfill. This is a one-one relationship because officeCode becomes a foreign key in the table on either side of the relationship between offices and employees. The foreign key should be placed in the table with fewer rows to minimize the number of NULL values, in this case employees table. The foreign key (employees.officeCode) refers to the primary key (offices.officeCode) on the opposite side of the relationship.

A screenshot of a computer

Description automatically generated

the orders and orderdetails we ran orderNumber equal to 10330, 10338, and 10194 in the WHERE clause selecting the columns of the orderdetails table.

A screenshot of a computer

Description automatically generated

1. **Delete records** from the payments table where the customer number equals 103.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

1. **Retrieve customer records** for sales representative Barry Jones and **identify** if the **relationships** are one-to-one or one-to-many**.**

A screenshot of a computer

Description automatically generated

This is a one-many relationship because the employeeNumber and the salesRepEmployeeNumber because a foreign key in the table on the 'many' side of the relationship (salesRepEmployeeNumber), referring to the primary key on the 'one' side (employeeNumber). The foreign key name is the primary key name with an optional prefix. The prefix is derived from the relationship name and clarifies the meaning of the foreign key. . In retrieving customer records, we made employees.employeeNumber equal to customers.salesRepEmployeeNumber because they share a relationship name and foreign key. This represents a one-many relationship and can use the foreign key salesRepEmployeeNumber to specify in the where clause what employeeNumber you want, in this case it was 1504.

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.

A screenshot of a computer

Description automatically generated

In retrieving records for customers, we SELECT employees.firstName and employees.lastName and run the same INNER JOIN statement made employees.employeeNumber equal to customers.salesRepEmployeeNumber to specify in the where clause the customer had to be in MA.

This represents a many-many relationship because the employee.firstName has two firstNames residing in MA and employee.lastName has two lastNames residing in MA. Also, their salesRepEmployeeNumber are the same refereeing to their names.

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.

A screenshot of a computer

Description automatically generated

I chose John Wayne although I did mess up by entering the name backwards so its Wayne John.

1. **Reflection**
   1. **Define how cardinality is applied** to the databases you’ve been working with and why different numbers of records returned from the different offices.

Cardinality is applied to the databases we’ve been working with because it defines how many of an attribute maximum there are and attribute minimum there are. Also, refers to the relationships and attributes in a database like the ERD template. The reason different number of records returned from different cities is because we used the WHERE clause to specify which cities, we wanted returned in this case it was San Francisco and New York City.

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

The different queries I ran apply to cardinality because they are different relationships and attributes. In the orders and orderdetails we ran orderNumber equal to 10330, 10338, and 10194 in the WHERE clause selecting the columns of the orderdetails table. In retrieving customer records we made employees.employeeNumber equal to customers.salesRepEmployeeNumber because they share a relationship name and foreign key. This represents a one-many relationship and can use the foreign key salesRepEmployeeNumber to specify in the where clause what employeeNumber you want, in this case it was 1504. Finally In retrieving records for customers we SELECT employees.firstName and employees.lastName and run the same INNER JOIN statement made employees.employeeNumber equal to customers.salesRepEmployeeNumber to specify in the where clause the customer had to be in MA. The attributes employees.firstName and emplyees.lastName show the result of a many-many relationship because each attribute has more than one values on both sides of the relationship. The main difference in running these different queries was knowing the relationship of the tables and the attributes they share with primary keys and foreign keys. Also, knowing the attributes of one or many on one side or both sides of the relationship.

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

One crucial benefit of cardinality in this type of database are Determine relationship maxima. Relationship Maximum the greatest number of instances of one entity that can relate to a single instance of another entity. Next crucial benefit of cardinality is attribute Minimum is the least number of attribute instances possible for a single entity instance.