

Haley Candia Perez

September 20, 2023

DAD-220

Lab 4-2

Haley's starting point as per email (I hope I did this right! Sorry in advance if it's incorrect)

```
Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| QuantigrationRMA |
| candiaperez |
| classicmodels |
| last_name_here |
| mysql |
| performance_schema |
+-----+
7 rows in set (0.00 sec)

mysql> use candiaperez;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> █
```

Start a terminal session and run this command: `mysql < mysqlsampledatabase.sql`

Type `mysql` in the command line and begin working with SQL the way you've been in previous labs. Write commands to use the `classicmodels` database and show its tables to verify that you're in the right place.

```
codio@zoomprotect-chinaholiday:~/workspace$ mysql < mysqlsampledatabase.sql
codio@zoomprotect-chinaholiday:~/workspace$ mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 37
Server version: 5.5.62-0ubuntu0.14.04.1 (Ubuntu)

Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use classicmodels;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+-----+
| Tables_in_classicmodels |
+-----+
| customers               |
| employees               |
| offices                 |
| orderdetails            |
| orders                  |
| payments                |
| productlines            |
| products                |
+-----+
8 rows in set (0.00 sec)

mysql>
```

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

- Command for San Francisco: `select firstName, lastName, jobTitle, offices.city from employees inner join offices on employees.officeCode = offices.officeCode where state = 'CA'.`
- Write and run a command to return records from New York on your own.

- Validate the completion of this step with a screenshot of these two tables.

```
mysql> select firstName, lastName, jobTitle, offices.city from employees inner join offices on employees.officeCode =
offices.officeCode where state = 'CA'
-> ;
```

firstName	lastName	jobTitle	city
Diane	Murphy	President	San Francisco
Mary	Patterson	VP Sales	San Francisco
Jeff	Firrelli	VP Marketing	San Francisco
Anthony	Bow	Sales Manager (NA)	San Francisco
Leslie	Jennings	Sales Rep	San Francisco
Leslie	Thompson	Sales Rep	San Francisco

```
6 rows in set (0.00 sec)
```

```
mysql> select firstName, lastName, jobTitle, offices.city from employees inner join offices on employees.officeCode =
offices.officeCode where state = 'NY';
```

firstName	lastName	jobTitle	city
Foon Yue	Tseng	Sales Rep	NYC
George	Vanauf	Sales Rep	NYC

```
2 rows in set (0.00 sec)
```

```
mysql> █
```

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

- Retrieve the order details by running SELECT queries with WHERE clauses against the orders table.
- Validate the completion of this step with a screenshot.

```
mysql> select orders.orderNumber, productCode, quantityOrdered, priceEach, orderLineNumber
-> from orders inner join orderdetails on orders.orderNumber = orderdetails.orderNumber
-> where orders.orderNumber = 10330 or orders.orderNumber = 10338 or orders.orderNumber = 10194;
```

orderNumber	productCode	quantityOrdered	priceEach	orderLineNumber
10194	S10_1949	42	203.59	11
10194	S10_4962	26	134.44	4
10194	S12_1666	38	124.37	8
10194	S18_1097	21	103.84	10
10194	S18_2432	45	51.05	2
10194	S18_4600	32	113.82	5
10194	S18_4668	41	47.79	9
10194	S24_2300	49	112.46	1
10194	S32_1268	37	77.05	3
10194	S32_3522	39	61.41	7
10194	S700_2824	26	80.92	6
10330	S18_3482	37	136.70	3
10330	S18_3782	29	59.06	2
10330	S18_4721	50	133.92	4
10330	S24_2360	42	56.10	1
10338	S18_1662	41	137.19	1
10338	S18_3029	28	80.86	3
10338	S18_3856	45	93.17	2

```
18 rows in set (0.00 sec)

mysql>
```

- Then, reference the Module Four Lab ERD to assist in identifying relationships. A version with alternative text is available: [Module Four Lab ERD With Alternative Text](#).
- Now, identify what type of cardinality this represents in the entity relationship model.

A. The type of cardinality would be one to many, because there can be one order with many different order details.

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

- Run a DESCRIBE statement to identify fields in the payments table first.
- Select the records from the payments table for customer number 103 before deleting them.

- Validate that the above instructions have worked with a screenshot.

```
mysql> describe payments;
+-----+-----+-----+-----+-----+-----+
| Field          | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| customerNumber | int(11)        | NO   | PRI | NULL    |       |
| checkNumber    | varchar(50)    | NO   | PRI | NULL    |       |
| paymentDate    | date           | NO   |     | NULL    |       |
| amount         | decimal(10,2) | NO   |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)

mysql> select * from payments where customerNumber = 103;
+-----+-----+-----+-----+
| customerNumber | checkNumber | paymentDate | amount |
+-----+-----+-----+-----+
| 103            | HQ336336   | 2004-10-19  | 6066.78 |
| 103            | JM555205   | 2003-06-05  | 14571.44 |
| 103            | OM314933   | 2004-12-18  | 1676.14 |
+-----+-----+-----+-----+
3 rows in set (0.00 sec)

mysql> 
```

- Delete the records from the payments table for customer number 103.
- Run a SELECT statement against the table to show that customer number 103 is no longer there.
- Validate the completion of this step with a screenshot.

```
mysql> DELETE FROM payments WHERE customerNumber = 103;
Query OK, 3 rows affected (0.02 sec)

mysql> select * from payments where customerNumber = 103;
Empty set (0.00 sec)

mysql> 
```

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

- Remember: SELECT, FROM, Inner Join, and WHERE.
- Use Barry's employeeNumber, 1504, and perform a join between the customer salesRepEmployeeNumber to retrieve these records.
- Validate the completion of this step with a screenshot.
- Identify whether these entities demonstrate one-to-one or one-to-many relationships.

```
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server
on for the right syntax to use near 'asd' at line 3
mysql> select employees.employeeNumber as 'SalesRep', customers.customerName, customerNumber
-> from employees inner join customers on employees.employeeNumber = customers.salesRepEmployeeNumber
-> where employees.employeeNumber = 1504;
```

SalesRep	customerName	customerNumber
1504	Baane Mini Imports	121
1504	Blauer See Auto, Co.	128
1504	Volvo Model Replicas, Co	144
1504	Herkku Gifts	167
1504	Clover Collections, Co.	189
1504	Toms Spezialitäten, Ltd	259
1504	Norway Gifts By Mail, Co.	299
1504	Bavarian Collectables Imports, Co.	415
1504	Scandinavian Gift Ideas	448

```
9 rows in set (0.00 sec)

mysql>
```

A. The relationships are one-many because the Sales Representative Barry Jones is only one person but he manages many different customers.

5. Retrieve records for customers who reside in Massachusetts and identify their sales rep and the relationship of entities.

- Remember: SELECT, FROM, Inner Join, and WHERE.

- Use employee.firstName and employee.lastName in your command.
- Identify whether these entities demonstrate one-to-one or many-to-many relationships.

```
mysql> select customers.customerName, customerNumber, Concat(employee.firstName, " ", employee.lastName) as 'SalesRep'
      -> from customers inner join employees on customers.salesRepEmployeeNumber = employees.employeeNumber
      -> where customers.state = 'MA';
```

customerName	customerNumber	SalesRep
Cambridge Collectables Co.	173	Julie Firrelli
Online Mini Collectables	204	Julie Firrelli
Mini Creations Ltd.	320	Julie Firrelli
Collectables For Less Inc.	379	Julie Firrelli
Diecast Collectables	495	Julie Firrelli
Auto-Moto Classics Inc.	198	Steve Patterson
Marta's Replicas Co.	286	Steve Patterson
Gifts4AllAges.com	362	Steve Patterson
FunGiftIdeas.com	462	Steve Patterson

```
9 rows in set (0.00 sec)

mysql>
```

A. This query retrieved two sales reps who have customers in Massachusetts. This would be a many to one case because even though there are multiple customers and sales representatives here, Julie is only one person, and so is Steve. They both have many customers that they take care of individually. If they had only one customer, then in that case it would be one to one. But this is not the case here.

6. 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. Think of this as your signature.
- 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;

- You should now be seeing all of the fields that you'll need to fill in to complete this step.
- Reference your Module Two lab or resources on how to populate these fields if you need to.
- Fields you'll need to populate: customerNumber, customerName, contactLastName, contactFirstName, phone, addressLine1, addressLine2, city, state, postalCode, country, salesRepEmployeeNumber, and creditLimit.
- Run a SELECT statement on the customers table, capture it in a screenshot, and put it in your template.

```
mysql> INSERT INTO customers
-> values (999, 'Number One', 'Candia', 'Haley', '1234567890', '29 Maple Street', NULL, 'Hartford', 'CT', '06727', 'USA', 1370, 45678.12);
Query OK, 1 row affected (0.02 sec)

mysql> select * from customers where customerNumber = 999;
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| customerNumber | customerName | contactLastName | contactFirstName | phone      | addressLine1 | addressLine2 | city      | state | postalCode | country |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 999 | Number One | Candia | Haley | 1234567890 | 29 Maple Street | NULL | Hartford | CT | 06727 | US |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql>
```

7. Reflection: Use the lab environment or the screenshots you've worked with for this step.

Address the following in your reflection:

- Define how cardinality is applied to the databases you've been working with and why different numbers of records returned from the different offices.

A. Cardinality is applied to the databases when, for example, one to one or one to many queries are searched. When you need to specifically look for one order or if you need to look at how many customers a sales representative is in charge of,

those both reference the one to one and one to many kinds of cardinality.

Different numbers of records were returned because I requested different information with each command, where the command directed me towards the exact information I was looking for in that instance.

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

B. The different queries we used showed how different cardinality can be applied in various ways. An example would be in the section where we were looking up a sales representative's number and seeing what customers they had. This used one to many as there was one sales representative with multiple customers. Another example was with customer 103, which is a zero to many cardinality because if there were no records for them it would populate zero, otherwise one or more records would populate if there was more than one order they had placed.

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

C. There are many benefits of cardinality in databases. The first that comes to mind would be how you can search a group of customers in a specific area. This can be useful to see trends and how many customers there are in a specific area which could help with marketing and showing where a business could potentially grow. Another benefit would be being able to search a specific order

number. This makes the information quick to retrieve, and you can see the status of a specific customer's account on the spot.