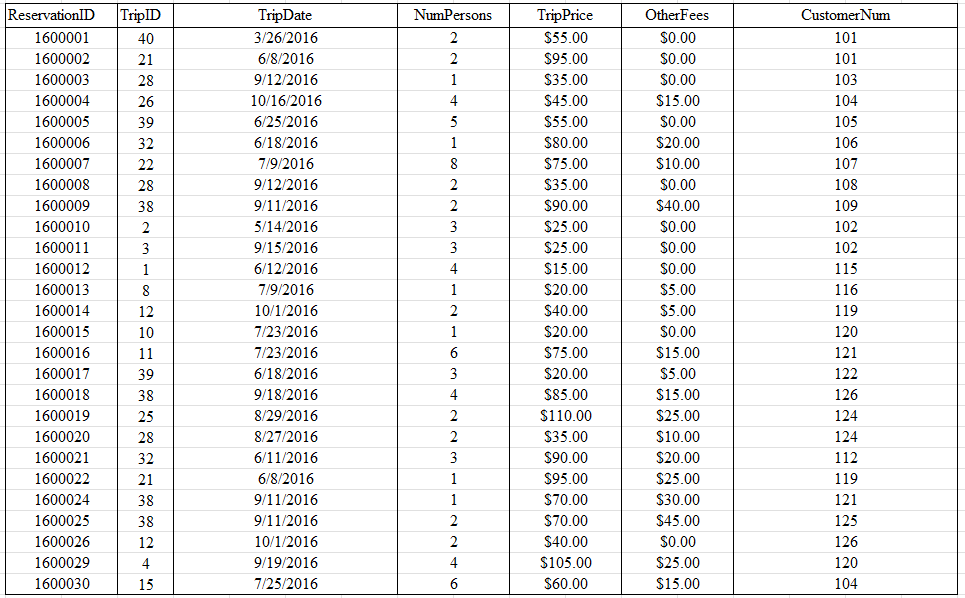
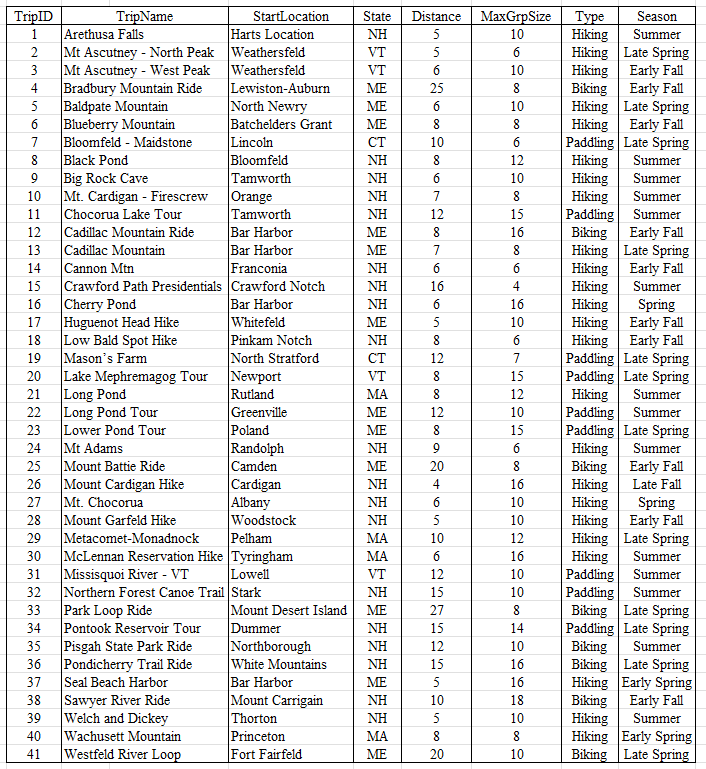
**Introduction**

The Colonial Adventure Tours Database is a vast and interconnected table designed to store and manage a wide range of information related to Colonial Adventure Tours' business operations. This database is critical for organizing and tracking various aspects of the business, such as customer data, reservations, trip details, and guide information. The use of a structured database system is critical for optimizing operations and providing a consistent experience for both the company and its customers.

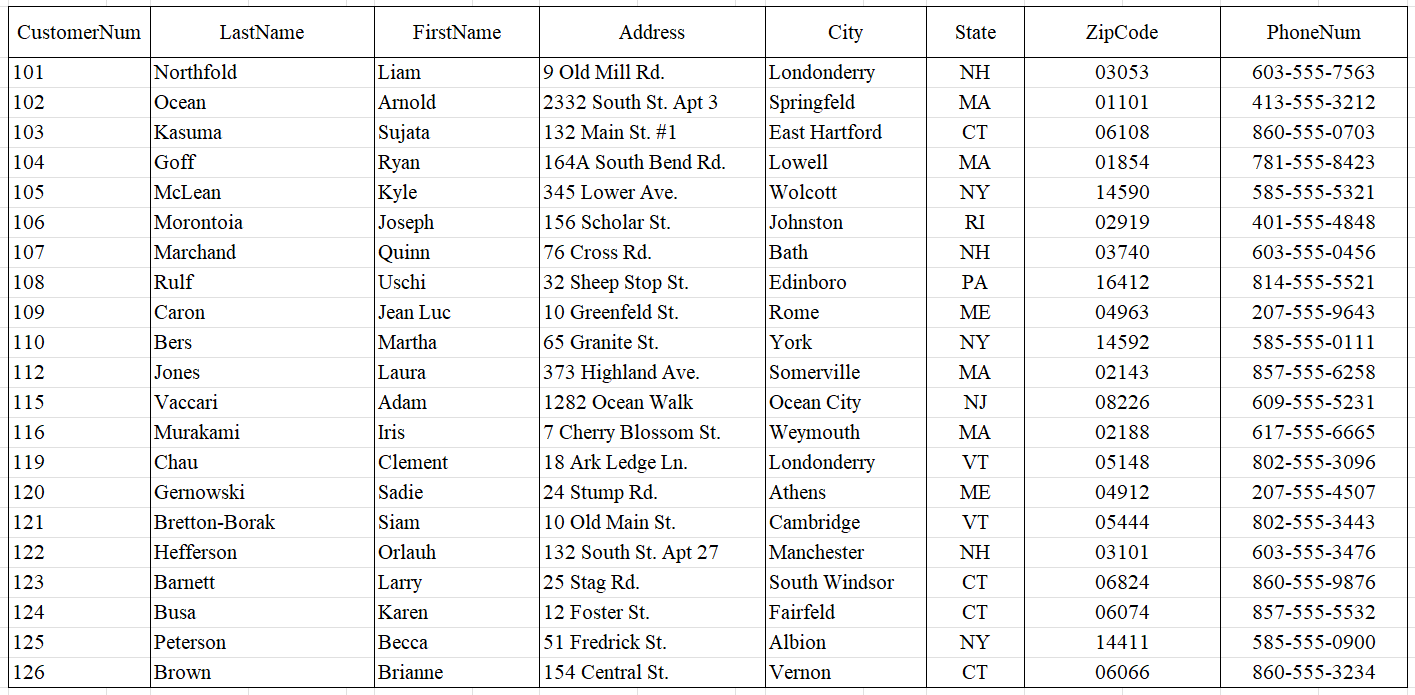
**Colonial Adventure Tours Database**

**Reservation Table  
**

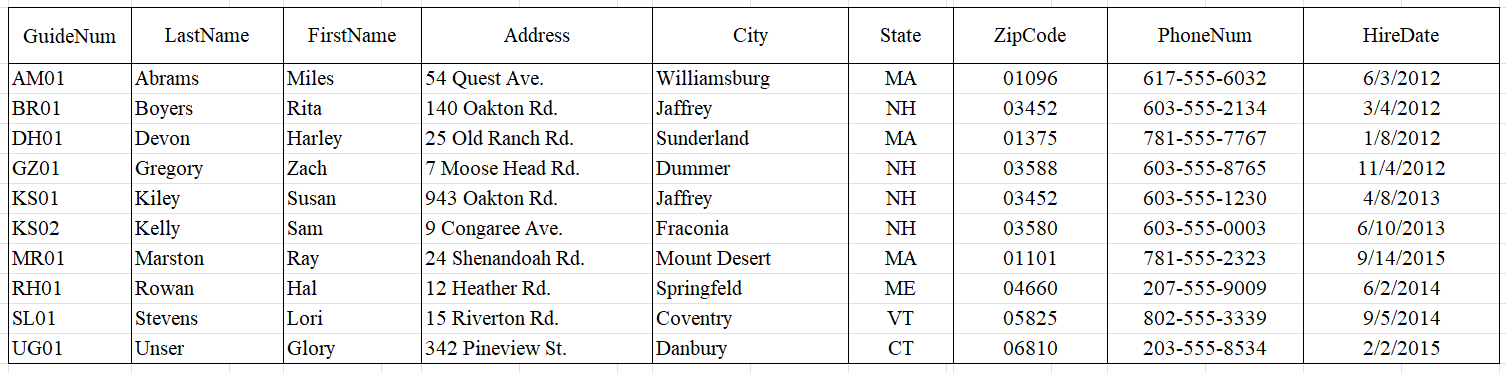
The reservation table has primary key ‘ReservationID’ which only contains unique values to identify other attributes such as TripID, TripDate, NumPersons, TripPrice, OtherFees, and CustomerNum. In this table, there are two (2) foreign keys which are ‘TripID’ to link the Trip table and ‘CustomerNum’ to link the Customer table. This table houses the information of every reservation placed.

**Trip Table**

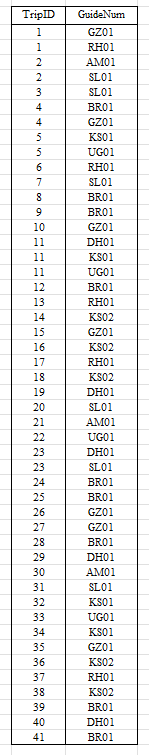
The trip table has the primary key ‘TripID’ which only contains unique values to reference other attributes in the table. It has the attributes ‘TripID’, ‘TripName’, ‘StartLocation’, ‘State’, ‘Distance’, ‘MaxGrpSize’, ‘Type’, and ‘Season’ that is functionally dependent to ‘TripID’. The trip table is where the trip information stored.

**Customer Table**

The customer table has the primary key ‘CustomerNum’ which its values is unique to identify the other attributes in the table. It has the attributes ‘CustomerNum’, ‘LastName’, ‘FirstName’, ‘Address’, ‘City’, ‘State’, ‘ZipCode’, and ‘PhoneNum’ which is functionally dependent to ‘CustomerNum’. This table stores the information of customers.

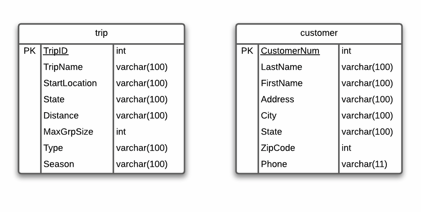
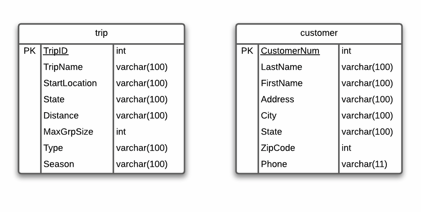
**Guide Table**

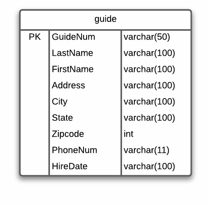
The guide table stores the information about the guides and has the primary key ‘GuideNum’ which its values to identify other attributes. It has the attributes ‘GuideNum’, ‘LastName’, ‘FirstName’, ‘Address’, ‘City’, ‘State’, ‘ZipCode’, ‘PhoneNum’, and ‘HireDate’.

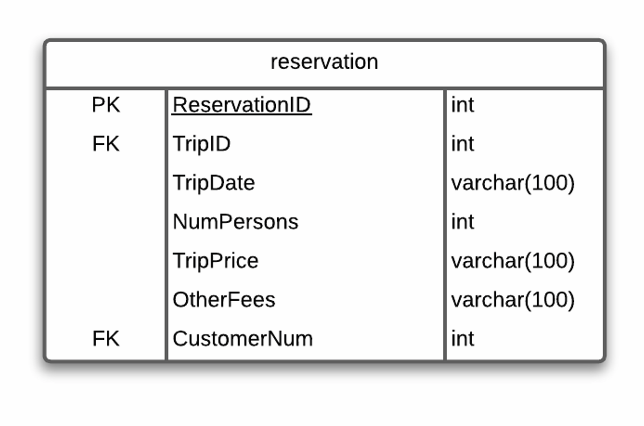
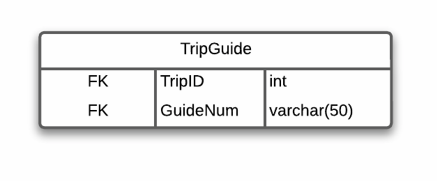
**TripGuide Table**

|  |  |
| --- | --- |
|  | This table is created to link the trip table and Guide table allowing a more organized information in the Colonial Adventure Tours Database. It has two (2) foreign keys, the ‘TripID’ form trip table and ‘GuideNum’ form guide table. |

**Entity Relationship Diagram**

An Entity-Relationship Diagram (ERD) is an illustration of the entities (objects or concepts), attributes (properties or characteristics), and relationships between those entities in a database or information system. ERDs are frequently used in database design and modeling to show how data is organized and linked. The tables below are from the database of Colonial Adventure Tours.





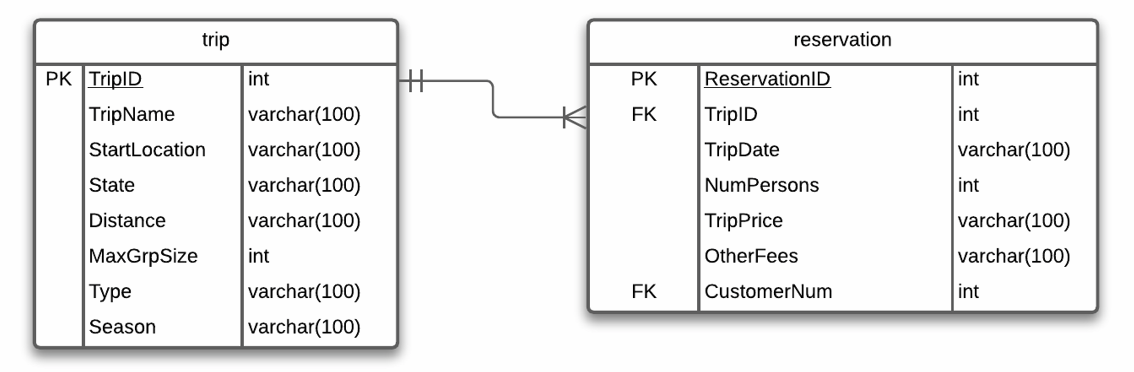
**Entity Relationship Diagram Notation (Crow’s Foot Notation)**

In Crow's Foot Notation, Entity-Relationship Diagram (ERD) cardinality represents the relationship between entities in a database model. Crow's Foot Notation is a popular graphical representation used in entity-relationship modeling to show the cardinality of entity relationships.

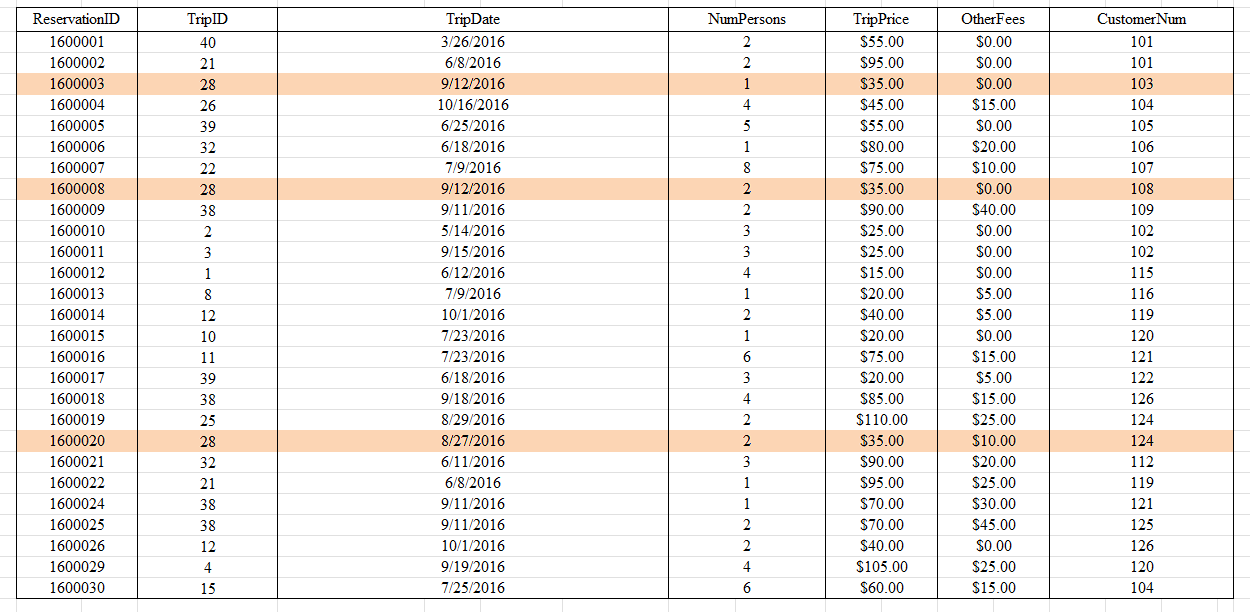
|  |  |
| --- | --- |
|  | This notation symbolizes Zero. |
|  | This notation symbolizes One. |
|  | This notation symbolizes Many. |
|  | This notation symbolizes Zero or Many. |
|  | This notation symbolizes One or Many. |
|  | This notation symbolizes One (and only One). |

With these notations, relationships between entities can be derived. Moreover, cardinality is the number of relationships or instances one entity has with another. Or simply put, it is the number of entities’ interactions have with each other. Cardinality is simply a ratio expressed in symbols, like one-to-one and the like.

Furthermore, the relationships between the tables in the Colonial Adventure Tours Database are defined by fields known as keys. The Reservation table is linked to the Trip and Customer tables via foreign keys (TripID and CustomerID, respectively), indicating that each reservation is associated with a specific trip and customer. The Trip table is then linked to the Guide table through the TripGuide table, establishing a many-to-one (and only one) relationship. This configuration enables the flexibility of assigning multiple guides to a single trip while ensuring that each guide is only responsible for leading one specific trip, resulting in a many-to-one relationship.

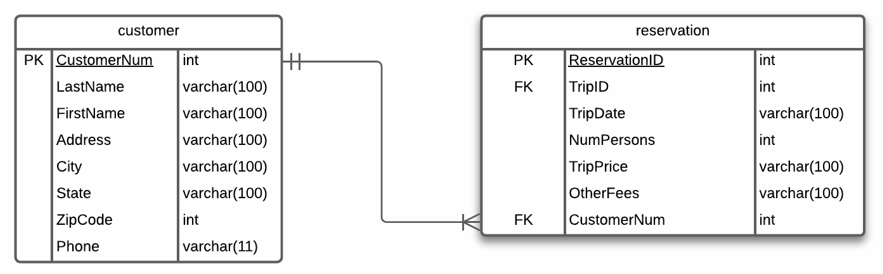
**Relationship between Trip and Reservation Table**

The entities above are trip and reservation. The **trip** entity is linked to **reservation** entity through a foreign key, ‘TripID’. Moreover, the relationship between these two (2) entities is **One (and only one)** to **One or Many.**  Each trip can have one or multiple reservations, and on the other hand, each reservation can only be associated with one trip.

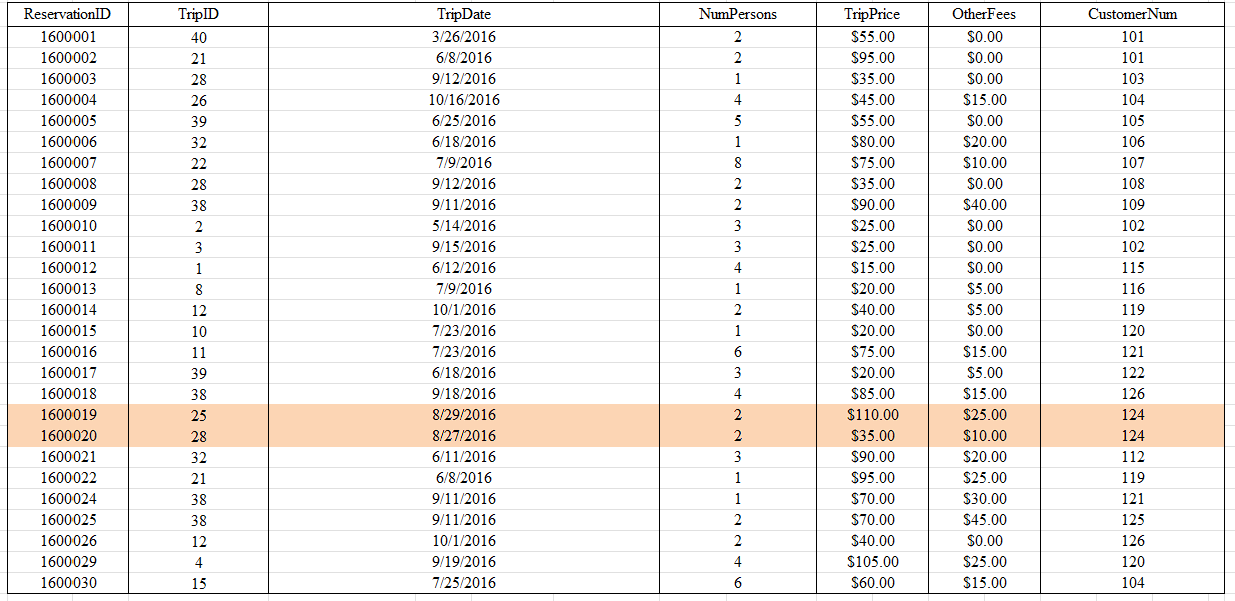
In Colonial Adventure Tours Database, a single trip has multiple reservations but each reservation was only associated with a single trip.

Referring from the table above, TripID 28 (Mount Garfeld Hike) has received multiple bookings. However, each trip is only associated with one and only one ReservationID.

**Relationship between Customer and Reservation**

****

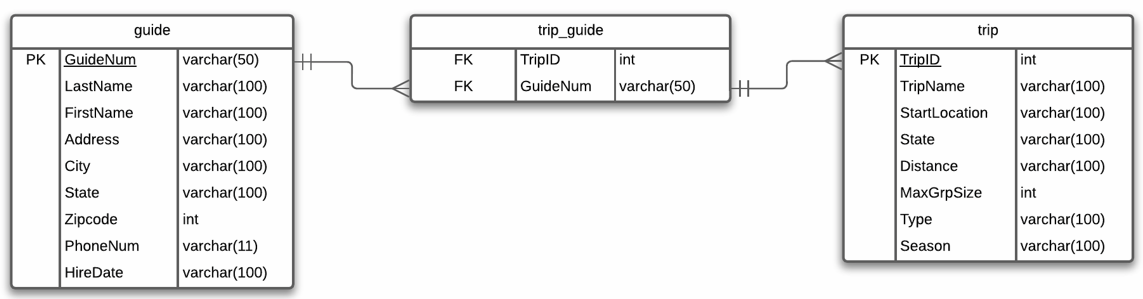
The entities above are customer and reservation. The **customer** entity is interconnected to the **reservation** entity through a foreign key, ‘CustomerNum’. The relationship between these entities is **One (and only one) to One or Many.** It is because a customer can book one or multiple reservations, and each reservation should only be associated with one and only one customer.



Using the same the same table (Reservation Table) from the Colonial Adventure Tour Database as an example. To emphasize the example, here is the compressed table where ReservationID, TripID and CustomerNum is only present.

|  |  |
| --- | --- |
|  | Basing from the table on the left, customer ‘124’ has booked or made a multiple trip reservation and those reservations is only to belong to customer ‘124’.  Moreover, a customer (customer ‘107’) can also book or have only one reservation. |

**Relationship between Guide and Trip**

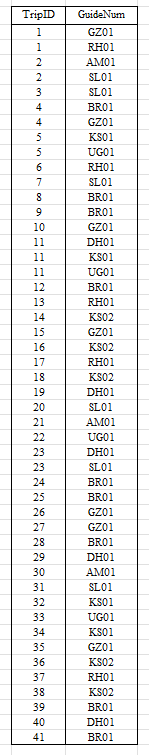
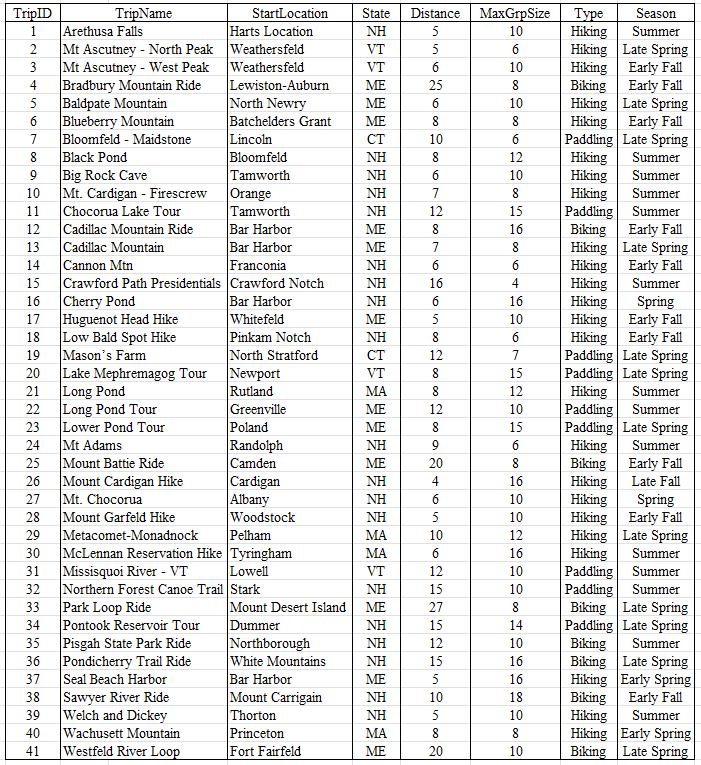
****

There are three (3) entities above which are guide, trip\_guide, and trip. However, just like with the previous discussion, the only focus in these entities is guide and trip. The **trip\_guide** entity will serve as the junction table or bridge containing two (2) foreign keys which are TripID and GuideNum to connect the **guide** and **trip** entity with each other. This established **One (and only One)** to **Many** relationships since a guide can handle multiple trips however each trip offered can only be guided with one (and only one) guide.

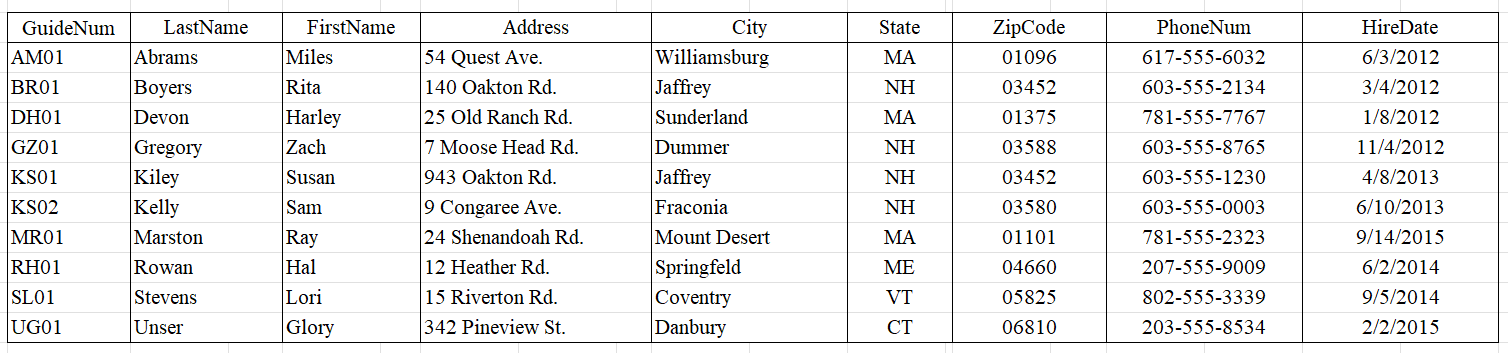
Below is the trip table, guide table, and the trip\_guide table from Colonial Adventure Tours Database for reference.

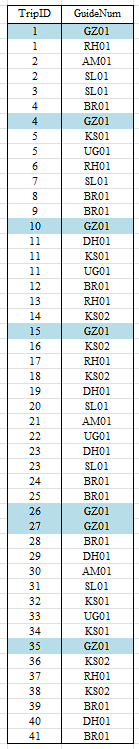
trip table

trip\_guide table

****

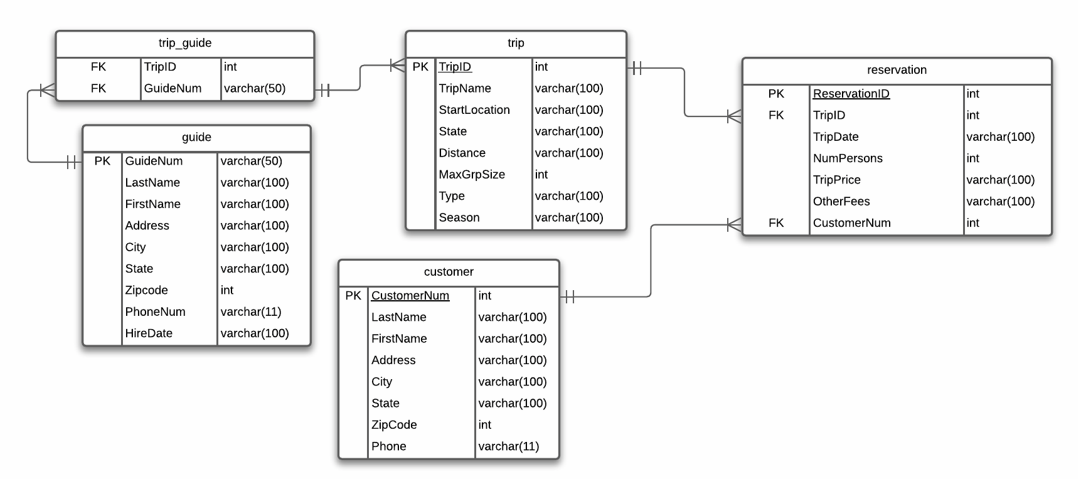
guide table



For better understandings, trip\_guide will be the focus since it contains the both primary key of guide and trip entity.

|  |  |
| --- | --- |
|  | Based from the table on the left, a Guide with a GuideNum ‘GZ01’ manage and handles multiple trips which are trip 1 (Arethusa Falls), trip 4 (Bradbury Mountain Ride), trip 10 (Mt. Cardigan – Firescrew), trip 15 (Crawford Path Presidentials Hike), trip 26 (Mount Cardigan Hike), trip 27 (Mt. Chocorua), and trip 35 (Pisgah State Park Ride).  However, each trip can only be guided by one guide at a time. |

**Colonial Adventure Tours Database Entity Relationship Diagram**

****

The Entity Relationship Diagram of the Colonial Adventure Tours database show the relationship between the entities and its attributes. The entity where most of the other entities link is the reservation entity with a primary key of ‘ReservationID’. The **trip** entity is linked to reservation entity through a foreign key, ‘TripID’. Same goes with the **customer** entity where it is also linked to reservation entity through a foreign key, ‘CustomerNum’. Furthermore, the **trip** and **guide** entity are linked with each other through a junction table or bridge using the foreign keys, ‘TripID’ and ‘GuideNum’. Moreover, there is no relationship between the entities’ **customer** and **trip** and **customer** and **guide.**