COMS W4111-002, V02 (Spring 2022)
Introduction to Databases

Homework 2: Non-Programming

Due Sunday, February 27, 2022 at 11:59 PM

Introduction

Overview

This homework has 1 section:

1. A section for non-programming track.

Submission

You will **submit 2 files** for this assignment.

1. Submit a zip file titled `<your_uni>_hw2_nonprogramming.zip` to **HW2 Non-Programming - Zip** on Gradescope.
    - Replace `<your_uni>` with your uni. My submission would be `dff9_hw2_nonprogramming.zip` .
    - The zipped directory should include:
        - TODO: include files in the hw directory
        - `<your_uni>_hw2_nonprogramming.ipynb` (substitute with your uni as above)
        - Any image files you embed in your notebook.
2. Submit a PDF title `<your_uni>_hw2_nonprogramming.pdf` to **HW2 Non-Programming - PDF** on Gradescope.
    - This should be a PDF of your completed HW2 Non-Programming Python notebook.
        - **Tag pages for each problem**. Per course policy, any untagged submission will receive an automatic 0.
        - Double check your submission on Gradescope to ensure that the PDF conversion worked and that your pages are appropriately tagged.

Collaboration and Information

- Answering some of the questions may require independent research to find information. We encourage you to try troubleshooting problems independently before reaching out for help.
- You may use any information you get in TA or Prof. Ferguson's office hours, from lectures or from recitations. This includes slides related to the recommended textbook.
- You may use information that you find on the web.
- You are NOT allowed to collaborate with other students outside of office hours.

Non-Programming

Setup

- Modify the cells below to setup your environment.
- The change should just be setting the DB user ID and password, replacing my user ID and password with yours for MySQL.

In [1]:
```
database_user_id = "root"
database_pwd = "Edy990127"
```

In [2]:
```
database_url = "mysql+pymysql://" + \
    database_user_id + ":" + database_pwd + "@localhost"
database_url
```

Out[2]:
```
'mysql+pymysql://root:Edy990127@localhost'
```

In [3]:
```
%reload_ext sql
```

In [4]:
```
%sql $database_url
```

Out[4]:
```
'Connected: root@None'
```

In [5]:
```
from sqlalchemy import create_engine
```

In [6]:
```
sqla_engine = create_engine(database_url)
```

In [7]:
```
#
# We are going to create a schema and some tables for the HW.
#
%sql create schema if not exists S22_W4111_HW2_B
%sql select 1;
```

```
 * mysql+pymysql://root:***@localhost
1 rows affected.
 * mysql+pymysql://root:***@localhost
1 rows affected.
```

Out[7]:

1
1

Install Datasets

Classic Models

- We will use the [Classic Models Tutorial (https://www.mysqltutorial.org/mysql-sample-database.aspx)](https://www.mysqltutorial.org/mysql-sample-database.aspx) database for HW 2 Non-Programming, other homework assignments, and exams.
- Lecture 5 briefly explained why this data model is interesting for education purposes. The problems on homework assignments and exams will further explore why it's interesting.
- The zip file for HW 2 Non-Programming contains an SQL script for creating a database `classicmodels` and loading the data. The script is `classicmodels.sql` .
- Use DataGrip to run the script. You performed this task for HW 0 with different SQL scripts. The basic approach is:
  - Right click on `@localhost`
  - Choose `Run SQL Script` .
  - Navigate to and select `classicmodels.sql` .
- The following cells test for correct installation.
- These cells are also examples of DDL statements and querying the "catalog."

In [8]:
```
%sql show tables from classicmodels
```

```
 * mysql+pymysql://root:***@localhost
8 rows affected.
```

Out[8]:

Tables_in_classicmodels
customers
employees
offices
orderdetails
orders
payments
productlines
products

In [9]:
```
%%sql

select
    table_schema, table_name, column_name, IS_NULLABLE, DATA_TYPE from informatic
where
    table_schema='classicmodels'
order by
    table_schema, table_name, ORDINAL_POSITION;
```

```
 * mysql+pymysql://root:***@localhost
59 rows affected.
```

Out[9]:

TABLE_SCHEMA	TABLE_NAME	COLUMN_NAME	IS_NULLABLE	DATA_TYPE
classicmodels	customers	customerNumber	NO	int
classicmodels	customers	customerName	NO	varchar
classicmodels	customers	contactLastName	NO	varchar
classicmodels	customers	contactFirstName	NO	varchar
classicmodels	customers	phone	NO	varchar
classicmodels	customers	addressLine1	NO	varchar
classicmodels	customers	addressLine2	YES	varchar
classicmodels	customers	city	NO	varchar
classicmodels	customers	state	YES	varchar
classicmodels	customers	postalCode	YES	varchar
classicmodels	customers	country	NO	varchar
classicmodels	customers	salesRepEmployeeNumber	YES	int
classicmodels	customers	creditLimit	YES	decimal
classicmodels	employees	employeeNumber	NO	int
classicmodels	employees	lastName	NO	varchar
classicmodels	employees	firstName	NO	varchar
classicmodels	employees	extension	NO	varchar
classicmodels	employees	email	NO	varchar
classicmodels	employees	officeCode	NO	varchar
classicmodels	employees	reportsTo	YES	int
classicmodels	employees	jobTitle	NO	varchar
classicmodels	offices	officeCode	NO	varchar
classicmodels	offices	city	NO	varchar
classicmodels	offices	phone	NO	varchar
classicmodels	offices	addressLine1	NO	varchar
classicmodels	offices	addressLine2	YES	varchar
classicmodels	offices	state	YES	varchar
classicmodels	offices	country	NO	varchar

TABLE_SCHEMA	TABLE_NAME	COLUMN_NAME	IS_NULLABLE	DATA_TYPE
classicmodels	offices	postalCode	NO	varchar
classicmodels	offices	territory	NO	varchar
classicmodels	orderdetails	orderNumber	NO	int
classicmodels	orderdetails	productCode	NO	varchar
classicmodels	orderdetails	quantityOrdered	NO	int
classicmodels	orderdetails	priceEach	NO	decimal
classicmodels	orderdetails	orderLineNumber	NO	smallint
classicmodels	orders	orderNumber	NO	int
classicmodels	orders	orderDate	NO	date
classicmodels	orders	requiredDate	NO	date
classicmodels	orders	shippedDate	YES	date
classicmodels	orders	status	NO	varchar
classicmodels	orders	comments	YES	text
classicmodels	orders	customerNumber	NO	int
classicmodels	payments	customerNumber	NO	int
classicmodels	payments	checkNumber	NO	varchar
classicmodels	payments	paymentDate	NO	date
classicmodels	payments	amount	NO	decimal
classicmodels	productlines	productLine	NO	varchar
classicmodels	productlines	textDescription	YES	varchar
classicmodels	productlines	htmlDescription	YES	mediumtext
classicmodels	productlines	image	YES	mediumblob
classicmodels	products	productCode	NO	varchar
classicmodels	products	productName	NO	varchar
classicmodels	products	productLine	NO	varchar
classicmodels	products	productScale	NO	varchar
classicmodels	products	productVendor	NO	varchar
classicmodels	products	productDescription	NO	text
classicmodels	products	quantityInStock	NO	smallint
classicmodels	products	buyPrice	NO	decimal
classicmodels	products	MSRP	NO	decimal

In [10]:
```
%%sql
use classicmodels;
with
    customer_orders_details as
        (
            select customerNumber, orderNumber, status, orderDate, shippedDate,
                   productCode, quantityOrdered, priceEach
            from orders natural join orderdetails
        ),
     customer_orders_totals as
         (
             select customerNumber, orderNumber,
                    concat(
                            '$',
                            format(sum(priceEach * quantityOrdered), 2)
                        ) as order_value
                from customer_orders_details
                group by customerNumber, orderNumber
         )
select * from customer_orders_totals LIMIT 10;
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
10 rows affected.
```

Out[10]:

customerNumber	orderNumber	order_value
103	10123	$14,571.44
103	10298	$6,066.78
103	10345	$1,676.14
112	10124	$32,641.98
112	10278	$33,347.88
112	10346	$14,191.12
114	10120	$45,864.03
114	10125	$7,565.08
114	10223	$44,894.74
114	10342	$40,265.60

World, Country, State, City

- Having definitive information about countries, cities, etc. is useful for data engineer and data insight.
- We will use information from [Darshan Gada's GitHub project (https://github.com/dr5hn)](https://github.com/dr5hn). For convenience, I have copied SQL scripts into the homework directory.
- Use DataGrip to create a schema `world_city_state` .

- Select the newly created schema and right click to choose `Run SQL Script` to run the scripts:
  - `world_city_state_countries.sql`
  - `world_city_state_states.sql`
  - `world_city_state_cities.sql`

Copy Information

- We want to preserve the original data. So, we will copy the data and **structure** into the HW 2 B database.
- Set the current database to `S22_W4111_HW2_B` .
- Create tables in the database for every table in `classicmodels` and `world_city_state` .
- Load the data into the new tables from the original tables.
- The tables in `S22_W4111_HW2_B` **MUST** have the same column names, types, constraints, etc.
- You **MUST** perform this task by executing SQL statements in cells below.
- This task may seem overly tedious and complex. But, if you think about it, you will realize that writing many of the statements from scratch is not necessary.

In [11]:
```
%%sql

use S22_W4111_HW2_B;
select 1;
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
1 rows affected.
```

Out[11]:

1
1

In [12]:
```
%%sql
drop table if exists s22_w4111_hw2_b.customers;
create table if not exists s22_w4111_hw2_b.customers
(
    customerNumber         int            null,
    customerName           varchar(50)    null,
    contactLastName        varchar(50)    null,
    contactFirstName       varchar(50)    null,
    phone                  varchar(50)    null,
    addressLine1           varchar(50)    null,
    addressLine2           varchar(50)    null,
    city                   varchar(50)    null,
    state                  varchar(50)    null,
    postalCode             varchar(15)    null,
    country                varchar(50)    null,
    salesRepEmployeeNumber int            null,
    creditLimit            decimal(10, 2) null
);

insert into s22_w4111_hw2_b.customers select * from classicmodels.customers
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
0 rows affected.
122 rows affected.
```

Out[12]:
```
[]
```

In [13]:
```
%%sql
drop table if exists s22_w4111_hw2_b.employees;
create table if not exists s22_w4111_hw2_b.employees
(
    employeeNumber int          null,
    lastName       varchar(50)  null,
    firstName      varchar(50)  null,
    extension      varchar(10)  null,
    email          varchar(100) null,
    officeCode     varchar(10)  null,
    reportsTo      int          null,
    jobTitle       varchar(50)  null
);

insert into s22_w4111_hw2_b.employees select * from classicmodels.employees
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
0 rows affected.
23 rows affected.
```

Out[13]:
```
[]
```

In [14]:
```
%%sql
drop table if exists s22_w4111_hw2_b.offices;
create table if not exists s22_w4111_hw2_b.offices
(
    officeCode   varchar(10) null,
    city         varchar(50) null,
    phone        varchar(50) null,
    addressLine1 varchar(50) null,
    addressLine2 varchar(50) null,
    state        varchar(50) null,
    country      varchar(50) null,
    postalCode   varchar(15) null,
    territory    varchar(10) null
);
insert into s22_w4111_hw2_b.offices select * from classicmodels.offices
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
0 rows affected.
7 rows affected.
```

Out[14]: []

In [15]:
```
%%sql
drop table if exists s22_w4111_hw2_b.orderdetails;
create table if not exists s22_w4111_hw2_b.orderdetails
(
    orderNumber     int            null,
    productCode     varchar(15)    null,
    quantityOrdered int            null,
    priceEach       decimal(10, 2) null,
    orderLineNumber smallint       null
);
insert into s22_w4111_hw2_b.orderdetails select * from classicmodels.orderdetails
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
0 rows affected.
2996 rows affected.
```

Out[15]: []

In [16]:
```
%%sql
drop table if exists s22_w4111_hw2_b.orders;
create table if not exists s22_w4111_hw2_b.orders
(
    orderNumber    int         null,
    orderDate      date        null,
    requiredDate   date        null,
    shippedDate    date        null,
    status         varchar(15) null,
    comments       text        null,
    customerNumber int         null
);
insert into s22_w4111_hw2_b.orders select * from classicmodels.orders
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
0 rows affected.
326 rows affected.
```

Out[16]:
```
[]
```

In [17]:
```
%%sql
drop table if exists s22_w4111_hw2_b.payments;
create table if not exists s22_w4111_hw2_b.payments
(
    customerNumber int            null,
    checkNumber    varchar(50)    null,
    paymentDate    date           null,
    amount         decimal(10, 2) null
);
insert into s22_w4111_hw2_b.payments select * from classicmodels.payments
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
0 rows affected.
273 rows affected.
```

Out[17]:
```
[]
```

In [18]:
```
%%sql
drop table if exists s22_w4111_hw2_b.productlines;
create table if not exists s22_w4111_hw2_b.productlines
(
    productLine     varchar(50)   null,
    textDescription varchar(4000) null,
    htmlDescription mediumtext    null,
    image           mediumblob    null
);
insert into s22_w4111_hw2_b.productlines select * from classicmodels.productlines
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
0 rows affected.
7 rows affected.
```

Out[18]: []

In [19]:
```
%%sql
drop table if exists s22_w4111_hw2_b.products;
create table if not exists s22_w4111_hw2_b.products
(
    productCode        varchar(15)    not null,
    productName        varchar(70)    not null,
    productLine        varchar(50)    not null,
    productScale       varchar(10)    not null,
    productVendor      varchar(50)    not null,
    productDescription text           not null,
    quantityInStock    smallint       not null,
    buyPrice           decimal(10, 2) not null,
    MSRP               decimal(10, 2) not null
);

insert into s22_w4111_hw2_b.products select * from classicmodels.products
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
0 rows affected.
110 rows affected.
```

Out[19]: []

In [20]:

```
%%sql
drop table if exists s22_w4111_hw2_b.countries;
create table if not exists s22_w4111_hw2_b.countries
(
    id              mediumint unsigned null,
    name            varchar(100)       null,
    iso3            char(3)            null,
    numeric_code    char(3)            null,
    iso2            char(2)            null,
    phonecode       varchar(255)       null,
    capital         varchar(255)       null,
    currency        varchar(255)       null,
    currency_name   varchar(255)       null,
    currency_symbol varchar(255)       null,
    tld             varchar(255)       null,
    native          varchar(255)       null,
    region          varchar(255)       null,
    subregion       varchar(255)       null,
    timezones       text               null,
    translations    text               null,
    latitude        decimal(10, 8)     null,
    longitude       decimal(11, 8)     null,
    emoji           varchar(191)       null,
    emojiU          varchar(191)       null,
    created_at      timestamp          null,
    updated_at      timestamp          null,
    flag            tinyint            null,
    wikiDataId      varchar(255)       null
);
insert into s22_w4111_hw2_b.countries select * from world_city_state.countries
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
0 rows affected.
250 rows affected.
```

Out[20]:

```
[]
```

In [21]:
```
%%sql
drop table if exists s22_w4111_hw2_b.states;
create table if not exists s22_w4111_hw2_b.states
(
    id           mediumint unsigned null,
    name         varchar(255)       null,
    country_id   mediumint unsigned null,
    country_code char(2)            null,
    fips_code    varchar(255)       null,
    iso2         varchar(255)       null,
    type         varchar(191)       null,
    latitude     decimal(10, 8)     null,
    longitude    decimal(11, 8)     null,
    created_at   timestamp          null,
    updated_at   timestamp          null,
    flag         tinyint            null,
    wikiDataId   varchar(255)       null
);
insert into s22_w4111_hw2_b.states select * from world_city_state.states
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
0 rows affected.
4963 rows affected.
```

Out[21]:
```
[]
```

In [22]:
```
%%sql
drop table if exists s22_w4111_hw2_b.cities;
create table if not exists s22_w4111_hw2_b.cities
(
    id           mediumint unsigned null,
    name         varchar(255)       null,
    state_id     mediumint unsigned null,
    state_code   varchar(255)       null,
    country_id   mediumint unsigned null,
    country_code char(2)            null,
    latitude     decimal(10, 8)     null,
    longitude    decimal(11, 8)     null,
    created_at   timestamp          null,
    updated_at   timestamp          null,
    flag         tinyint            null,
    wikiDataId   varchar(255)       null
);
insert into s22_w4111_hw2_b.cities select * from world_city_state.cities
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
0 rows affected.
148061 rows affected.
```

Out[22]:
```
[]
```

In [23]:
```
%sql select * from s22_w4111_hw2_b.cities limit 10;
```

```
 * mysql+pymysql://root:***@localhost
10 rows affected.
```

Out[23]:

id	name	state_id	state_code	country_id	country_code	latitude	longitude	created_at
1	Andorra la Vella	488	07	6	AD	42.50779000	1.52109000	2019-10-05 23:58:06
2	Arinsal	493	04	6	AD	42.57205000	1.48453000	2019-10-05 23:58:06
3	Canillo	489	02	6	AD	42.56760000	1.59756000	2019-10-05 23:58:06
4	El Tarter	489	02	6	AD	42.57952000	1.65362000	2019-10-05 23:58:06
5	Encamp	487	03	6	AD	42.53474000	1.58014000	2019-10-05 23:58:06
6	Ordino	491	05	6	AD	42.55623000	1.53319000	2019-10-05 23:58:06
7	Pas de la Casa	487	03	6	AD	42.54277000	1.73361000	2019-10-05 23:58:06
8	Sant Julià de Lòria	490	06	6	AD	42.46372000	1.49129000	2019-10-05 23:58:06
9	la Massana	493	04	6	AD	42.54499000	1.51483000	2019-10-05 23:58:06
10	les Escaldes	492	08	6	AD	42.50729000	1.53414000	2019-10-05 23:58:06

Data Transformation

- The query below shows some information from `classicmodels.customers` .

In [24]:
```
%sql select * from classicmodels.customers limit 10;
```

```
 * mysql+pymysql://root:***@localhost
10 rows affected.
```

Out[24]:

customerNumber	customerName	contactLastName	contactFirstName	phone	addressLine1	a
103	Atelier graphique	Schmitt	Carine	40.32.2555	54, rue Royale	
112	Signal Gift Stores	King	Jean	7025551838	8489 Strong St.	
114	Australian Collectors, Co.	Ferguson	Peter	03 9520 4555	636 St Kilda Road	
119	La Rochelle Gifts	Labrune	Janine	40.67.8555	67, rue des Cinquante Otages	
121	Baane Mini Imports	Bergulfsen	Jonas	07-98 9555	Erling Skakkes gate 78	
124	Mini Gifts Distributors Ltd.	Nelson	Susan	4155551450	5677 Strong St.	
125	Havel & Zbyszek Co	Piestrzeniewicz	Zbyszek	(26) 642-7555	ul. Filtrowa 68	
128	Blauer See Auto, Co.	Keitel	Roland	+49 69 66 90 2555	Lyonerstr. 34	
129	Mini Wheels Co.	Murphy	Julie	6505555787	5557 North Pendale Street	
131	Land of Toys Inc.	Lee	Kwai	2125557818	897 Long Airport Avenue	

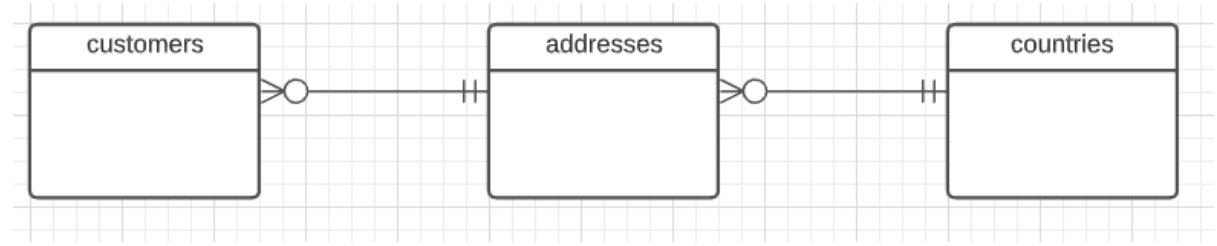
- There are several problems with this table definition, but we will focus on two.

1. Directly storing values like a country's or city's name is error prone. For example, different users and applications could enter various values:
   - Country: "United States," "USA," "US," etc.
   - City: "NYC," "New York," etc.

2. Having address information in rows with company information can cause errors and ambiguity over time, e.g.
   - There are cases where multiple companies have the same address, or a company has multiple addresses.
   - Just because a company "goes away" does not mean the address "went away."

- To fix these problems, you must transform the schema and data. This task will also require some data cleanup.

- The conceptual model you should implement is:



Customers-Address Conceptual Model

- You have to determine how to connect/link the tables. While you may include columns in one table that contain values in another table, do not worry about formally setting foreign key constraints. The important think is that you understand how they're linked.
- A good design would also handle ambiguity over city, state, etc. names. You do not need to worry about anything other than removing addresses from customers and handling countries.
- In the cells below, enter your SQL statements for creating and modifying tables, and modify data.

In [25]:
```
%%sql
drop table if exists s22_w4111_hw2_b.kari_customers;
create table if not exists s22_w4111_hw2_b.kari_customers
(
    customerNumber         int            not null
        primary key,
    customerName           varchar(50)    null,
    contactLastName        varchar(50)    null,
    contactFirstName       varchar(50)    null,
    phone                  varchar(50)    null,
    addressLine1           varchar(50)    null,
    addressLine2           varchar(50)    null,
    salesRepEmployeeNumber int            null,
    creditLimit            decimal(10, 2) null
);

insert into s22_w4111_hw2_b.kari_customers
select customerNumber, customerName, contactLastName, contactFirstName, phone,
addressLine1, addressLine2, salesRepEmployeeNumber, creditLimit from classicmodel
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
0 rows affected.
122 rows affected.
```

Out[25]:
```
[]
```

In [26]:
```
%%sql
drop table if exists s22_w4111_hw2_b.kari_addresses;
create table if not exists s22_w4111_hw2_b.kari_addresses
(
    addressLine1 varchar(50) not null
        primary key,
    addressLine2 varchar(50) null,
    city         varchar(50) null,
    state        varchar(50) null,
    postalCode   varchar(15) null,
    country      varchar(50) null
);

insert into s22_w4111_hw2_b.kari_addresses
select addressLine1, addressLine2, city, state, postalCode, country
from classicmodels.customers
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
0 rows affected.
122 rows affected.
```

Out[26]:
```
[]
```

In [27]:
```
%%sql
drop table if exists s22_w4111_hw2_b.kari_countries;
create table if not exists s22_w4111_hw2_b.kari_countries
(
    country varchar(50) not null
        primary key
);

insert into s22_w4111_hw2_b.kari_countries select distinct country from classicmc
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
0 rows affected.
27 rows affected.
```

Out[27]:
```
[]
```

In [28]:
```
%%sql
update s22_w4111_hw2_b.kari_addresses
  set city = "New York"
  where city in ("NYC", "NY");
update s22_w4111_hw2_b.kari_addresses
  set country = "United States"
  where country in ("USA", "US");
update s22_w4111_hw2_b.kari_addresses
  set country = "United Kingdom"
  where country in ("UK");
update s22_w4111_hw2_b.kari_countries
  set country = "United States"
  where country in ("USA", "US");
update s22_w4111_hw2_b.kari_countries
  set country = "United Kingdom"
  where country in ("UK");
```

```
 * mysql+pymysql://root:***@localhost
5 rows affected.
36 rows affected.
5 rows affected.
1 rows affected.
1 rows affected.
```

Out[28]:
```
[]
```

- Put SQL statements in the cell below to return information about customers, including address.

In [29]:
```
%sql describe s22_w4111_hw2_b.kari_customers
```

```
 * mysql+pymysql://root:***@localhost
9 rows affected.
```

Out[29]:

Field	Type	Null	Key	Default	Extra
customerNumber	int	NO	PRI	None	
customerName	varchar(50)	YES		None	
contactLastName	varchar(50)	YES		None	
contactFirstName	varchar(50)	YES		None	
phone	varchar(50)	YES		None	
addressLine1	varchar(50)	YES		None	
addressLine2	varchar(50)	YES		None	
salesRepEmployeeNumber	int	YES		None	
creditLimit	decimal(10,2)	YES		None	

In [30]:
```
%sql describe s22_w4111_hw2_b.kari_addresses
```

```
 * mysql+pymysql://root:***@localhost
6 rows affected.
```

Out[30]:

Field	Type	Null	Key	Default	Extra
addressLine1	varchar(50)	NO	PRI	None	
addressLine2	varchar(50)	YES		None	
city	varchar(50)	YES		None	
state	varchar(50)	YES		None	
postalCode	varchar(15)	YES		None	
country	varchar(50)	YES		None	

In [31]:
```
%sql describe s22_w4111_hw2_b.kari_countries
```

```
 * mysql+pymysql://root:***@localhost
1 rows affected.
```

Out[31]:

Field	Type	Null	Key	Default	Extra
country	varchar(50)	NO	PRI	None	