|  |
| --- |
| **CIS355 HOMEWORK 10: Still More Queries!** |

* You will have to understand the data (both meta-data & raw/end user data) to write good queries, so be curious, experiment, & use the “Object Explorer” as needed.
* All columns should have meaningful names.
* State any assumptions you make, where a question is unclear.
* Note that some questions ask for your code (i.e., “Show the SELECT statement…”) while others just ask for an answer to a question (e.g., “How many rows does the query return” or “why did x happen”).

**Part 1: AdventureWorks2012 Database.  
Refer to the following Database Diagram (from the Person schema) for this section.**

|  |  |
| --- | --- |
| |  | | --- | |  | |

USE AdventureWorks2012

/\* HINT: The PersonType attribute in the Person.Person table is relevant for

the queries in this section. PersonType indicates what kind of person each row

in the table represents. It is a required column and its values are either

SC, IN, SP, EM, VC, or GC. SC = Store Contact, IN = Individual (retail) customer,

SP = Sales person, EM = Employee (non-sales), VC = Vendor contact, GC = General contact

1a. Execute the following query, which produces a list of vendor contact names,

where the contact name consists of only the first name and the last name, both of

which are required (NOT NULL) columns. How many rows are returned and who is

the first contact listed? \*/

SELECT FirstName + ' ' + LastName As "Vendor Contact"

FROM Person.Person

WHERE PersonType = 'VC'

ORDER BY LastName, FirstName

/\*

1b. Execute the following query, which is the same as above except that I've

added middle name (a column that may have NULL values) to the SELECT clause.

The results don't look right! What happened & why? \*/

SELECT FirstName + ' ' + MiddleName + ' ' + LastName As "Vendor Contact"

FROM Person.Person

WHERE PersonType = 'VC'

ORDER BY LastName, FirstName

/\*

1c. How can I fix the above query (and still keep middle name in the result)

so that I get back better results? You can either explain what you would do

or show a revised query. \*/

/\*

2. Show the SQL statement to list the name and address for each salesperson.

You can assume that a salesperson has only one address.

We don't need to see complete addresses -- just the city,

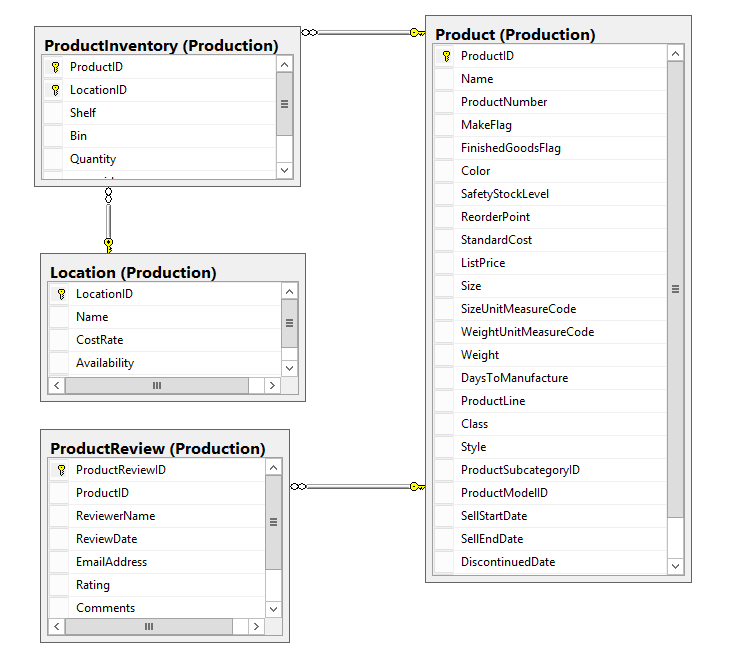
state (or province), and the country.

Use the full name for the state (province) and country -- not the

abbreviations (e.g., Alberta, Canada instead of AB, CA).

Sort the results by country, then state (province), then city. \*/

**Part 2: AdventureWorks2012 Database.  
Use the following Database Diagram (from the Production schema) for this section.**



USE AdventureWorks2012

/\*

3. A user wants to know which products (i.e., product numbers and names) have reviews.

This question can be answered in (at least) two ways: using a JOIN of the Product &

ProductReview tables, or using a subquery. It is similar to the first subquery example

from Thursday’s class (4/12) – in the Class Notes on subqueries. \*/

--a. Show the SELECT statement that answers the user’s question using a JOIN.

--b. Show the SELECT statement that answers the same question using a subquery.

/\*

4. Show the SQL statement to answer the following question:

What products are located in the paint shop? List the shelf, bin, product name,

product number and quantity for every product in the paint shop location.

(Hint: 'Paint shop' is the name of a location in the Production.Location table.) \*/

**Part 3: GI\_Animal Database.  
Use the following Database Diagram for this section.**

|  |
| --- |
|  |

USE GI\_Animal

/\* This table is from a database that tracks livestock animals and their genealogy.

Note that there are two unary relationships on the Animal table, one for fathers and

one for mothers, and two corresponding FKs. Get familiar with the data by executing

the following query: \*/

SELECT \* FROM Animal

/\*

5. Show the SELECT statement that answers the following information request:

List the IDs and names of the sheep who are fathers, along with

the IDs and names of their offspring. Sort the results by father name. \*/

/\*

6. Show the SELECT stateList that answers the following question:

What is the name of Buck's mother?

Note: You can answer this question with either a self-join or a subquery. \*/