ORIE 3120: Industrial Data and Systems Analysis
Spring 2020
Homework #3
Due Date: 2/19/2020 (Wednesday) 2:30pm

1. SQL: Joins and Aggregations (20 points)

Download the database *HW3Q1.sqlite*. In that database, there are two tables: Dept and Empl.

Table "Dept" has four records:

DeptID	DepartmentName
1	Engineering
2	Data Science
3	Product
4	Finance

Table "Empl" has six records:

EmployeeID	DeptID	Name
1001	1	Qing
1002	1	Eugene
1003	2	JR
1004	2	Francisca
1005	3	AJ
1006	6	Arthur

In this question you will write four queries using joins and group bys to produce different query results. For each of the four parts below, write a query that reproduces the same set of records (you do not need to reproduce the order in which they are shown). There are multiple queries that will produce the same records.

Required Submission for each part:

- In your pdf file:
  - Your queries
  - Screenshot of the query results verifying that you were able to reproduce the required records
- In your text file:
  - Your queries

(a) Write a query that produces the following result:

DepartmentName	EmployeeName
Engineering	Qing
Engineering	Eugene
Data Science	JR
Data Science	Francisca
Product	AJ

(b) Write a query that produces the following result:

DepartmentName	EmployeeName
Engineering	Qing
Engineering	Eugene
Data Science	JR
Data Science	Francisca
Product	AJ
NULL	Arthur

(c) Write a query that produces the following result:

DepartmentName	EmployeeName
Engineering	Eugene
Engineering	Qing
Data Science	Francisca
Data Science	JR
Product	AJ
Finance	*NULL*

(d) Write a query that produces the following result:

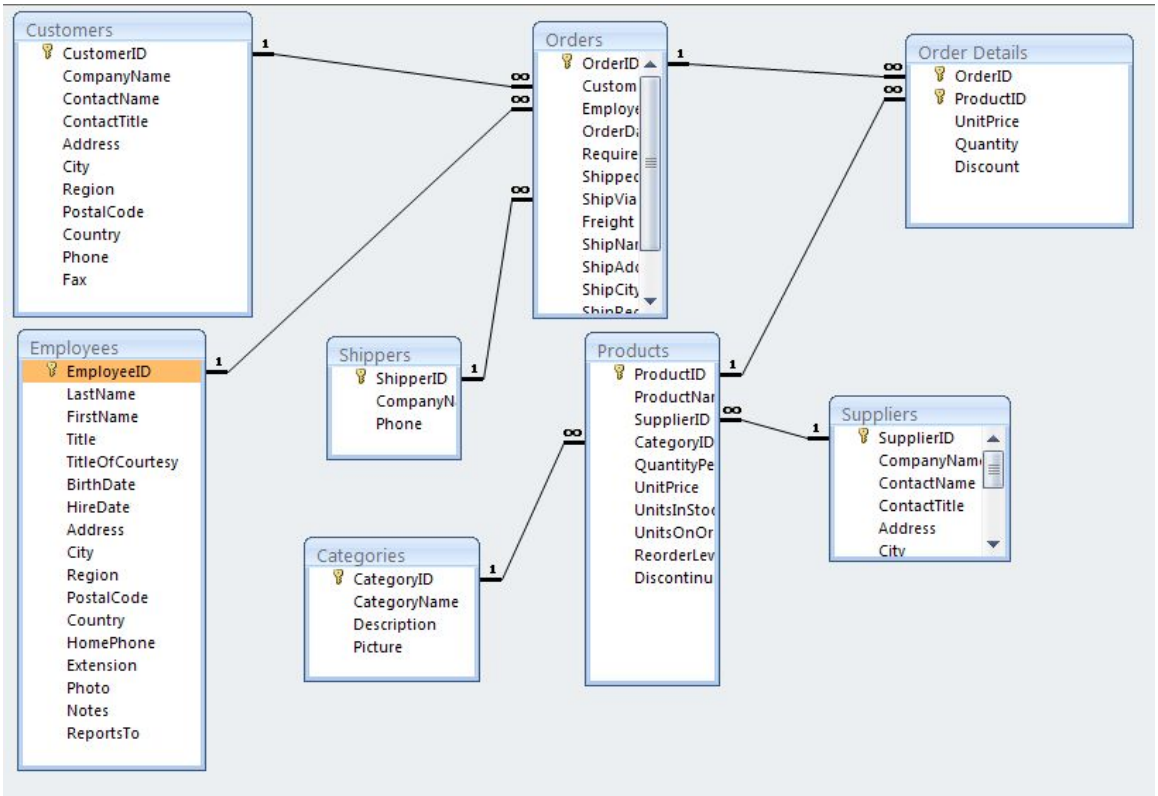
DepartmentName	EmployeeName
Engineering	Qing
Engineering	Eugene
Data Science	JR
Data Science	Francisca
Product	AJ
NULL	Arthur
Finance	*NULL*

(e) Write a query that produces the following result:

DepartmentName	NumEmployees
Data Science	2
Engineering	2
Finance	0
Product	1

2. SQL: Joins and Aggregations (20 points)

The example database used in class is called the "Northwind" database. It has the following schema. Download this database here: *HW3_Northwind.sqlite*.



As an inventory manager for the Northwind company, you are interested in knowing what fraction of your inventory in each category is obsolete (discontinued). If you see a category with a high fraction of obsolete inventory, then you would want to have a meeting with the purchasing manager to discuss the problem. Let's create a set of queries that will compute this for you.

The Products table has most of the data you need. Your total inventory position in a product should be considered the sum of UnitsInStock and UnitsOnOrder (the units on order will arrive and become part of your inventory). It does not make sense to add units of different products together (literally "apples" and "oranges"), so you should multiply by UnitPrice first. That is, InventoryPositionValue equals (UnitsInStock+UnitsOnOrder)*UnitPrice. You could then use a CASE clause based on the Discontinued field to determine the value of your discontinued product. That is, DiscontinuedInventoryPositionValue is equal to InventoryPositionValue if Discontinued is true, and 0 if it is false.

You would then want to compute totals of these two numbers (InventoryPositionValue and DiscontinuedInventoryPositionValue) grouped by CategoryID. From these two totals, for each category, you could compute the fraction of discontinued value to total value. You should avoid dividing by zero (another CASE clause: if the sum of

InventoryPositionValue within a category is strictly less than 0.01 then the fraction is zero.).

Finally, you could join the result with the categories table to get the category name for each categoryID in the result. So, your final query result would include the following fields: CategoryName, SumInventoryPositionValue, SumDiscontinuedInventoryPositionValue, FractionDiscontinuedValue and it will have exactly one row for each category represented in the Products table. You may use shorter or different names if you like.

You may solve this problem with a different sequence of queries but at least one of your queries must use an INNER JOIN.

Required Submission:

- In your pdf file:
  - Your queries
  - Screenshot of the final query result
- In your text file:
  - Your queries

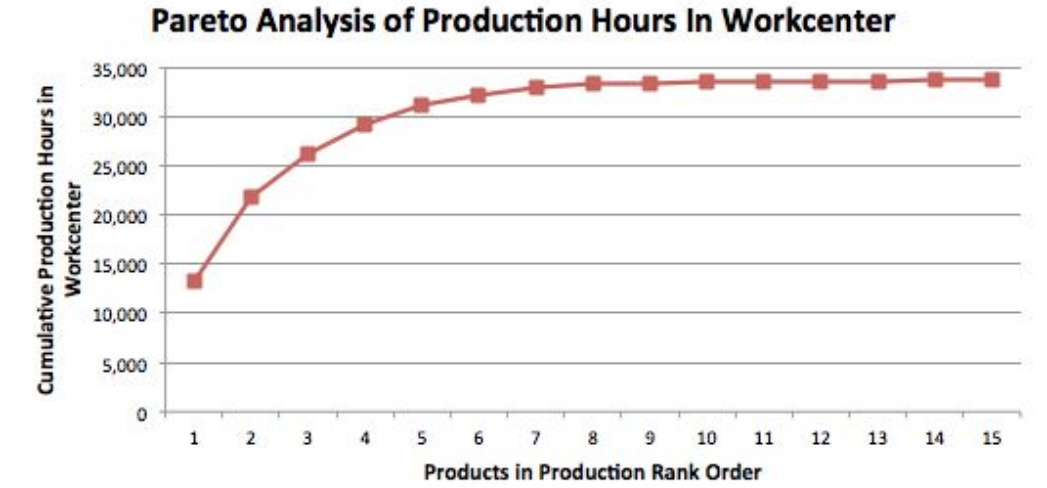
3. SQL: Cross Products and Aggregations (20 points)

The data for this problem are in the database called *HW3_Pareto.sqlite*, which gives a table of production hours (number of hours spent producing the product) within a single workcenter, for each product id:

#	ProductID	ProductionHours
1	A406-030	4385
2	A406-040	8592
3	A500-015	785
4	B127-010	328
5	C120-010	124
6	M820-010	2893
7	M830-010	35
8	N900-010	57
9	R400-020	1136
10	R400-025	1893
11	R400-030	43
12	S830-010	68
13	S830-040	23
14	T103-080	13290
15	X999-000	2

Observe the wide variation in the number of production hours by product. I would like to graph this by ranking the products according to production hours

(largest-producing product first) and then plotting the cumulative production hours by production rank. It will look like this:



This sort of cumulative plot is called a "Pareto Analysis." It helps to dramatize the fact that a small fraction of the products being produced account for most of the production hours in the workcenter. Three of the products in particular (20% of the product numbers) account for more than 25,000 hours (78% of the total). It

#	ProductID	ProductionHours	ProductionRank	CumulativeHours
1	A406-030	4385	3	26267
2	A406-040	8592	2	21882
3	A500-015	785	7	32974
4	B127-010	328	8	33302
5	C120-010	124	9	33426
6	M820-010	2893	4	29160
7	M830-010	35	13	33629
8	N900-010	57	11	33551
9	R400-020	1136	6	32189
10	R400-025	1893	5	31053
11	R400-030	43	12	33594
12	S830-010	68	10	33494
13	S830-040	23	14	33652
14	T103-080	13290	1	13290
15	X999-000	2	15	33654

immediately raises questions such as why the lower volume products are being produced at all. You will want to use this type of chart in presentations to management.

Once you have the data to plot, creating the plot using plotting software (Excel, R, python, etc.) is straightforward. However, you will need to prepare the data for plotting first using SQL. This is a non-trivial task and requires a trick. I created the

data for the above plot using three queries. The results were as follows:

#	ProductID	ProductionHours	GreaterProductID	GreaterProductionHours
1	A406-030	4385	A406-030	4385
2	A406-030	4385	A406-040	8592
3	A406-030	4385	T103-080	13290
4	A406-040	8592	A406-040	8592
5	A406-040	8592	T103-080	13290
6	A500-015	785	A406-030	4385
7	A500-015	785	A406-040	8592
8	A500-015	785	A500-015	785
9	A500-015	785	M820-010	2893
10	A500-015	785	R400-020	1136
11	A500-015	785	R400-025	1893
12	A500-015	785	T103-080	13290
13	B127-010	328	A406-030	4385
14	B127-010	328	A406-040	8592
15	B127-010	328	A500-015	785
16	B127-010	328	B127-010	328
17	B127-010	328	M820-010	2893
18	B127-010	328	R400-020	1136
19	B127-010	328	R400-025	1893

… (there were additional rows in the result of this first query)

#	ProductionRank	CumulativeHours
1	1	13290
2	2	21882
3	3	26267
4	4	29160
5	5	31053
6	6	32189
7	7	32974
8	8	33302
9	9	33426
10	10	33494
11	11	33551
12	12	33594
13	13	33629
14	14	33652
15	15	33654

Sketching my approach:

- I created the first query, saving it as a view, with two copies of the same table (not joined: i.e. all possible combinations) but restricted to combinations for

which production hours in one table were greater than or equal to production hours in the other table. Note that you can use the AS keyword for tables in the same way that you use it for columns.

- Then I performed an aggregation query over the results from this view, grouping on fields from the first copy of the table, counting records from the second copy of the table (to get ProductionRank), and summing production hours in the second copy of the table (to get Cumulative Hours). I saved this query as a second view.
- Finally, I wrote a query selecting only the ProductionRank and CumulativeHours fields and sorting by ProductionRank, saving it as a third view.

There are other ways to accomplish this same effect in SQL. If you prefer, you can use a different approach than the one described above. However, it must result in a correct answer to part (a) below.

(a) Write a query or a series of queries that results in two fields, ProductionRank and CumulativeHours, sorted by ProductionRank, as shown in the last query result above.

(b) Copy or export the final query result to a spreadsheet program like Excel and print out a chart with a format similar to the Pareto chart above. If you want to use different plotting software (R, Python, gnuplot, etc.), that is fine.

Because the query result is short, using copy & paste is easiest (just select them, use the "copy" keyboard shortcut to copy them to the clipboard, and then use paste in Excel). If you prefer to export them, you can on the export icon,

Then select "Query results". SQLiteStudio will then ask you to type in a query whose results you want to save. You can either copy and paste the the query from the third view you created, or you can type the query SELECT * FROM ViewName, where you have substituted the name that you used for your third view. This will let you save the view's results as a csv, which you can then import to Excel or other plotting software.

Required Submission:

- In your pdf file:
  - Your queries
  - Screenshots of the few few lines of the results from each query
  - Screenshot of your plot
- In your text file:
  - Your queries

4. Prerequisite Material on Normal Distributions From ENGRD 2700 (20 points)

We will soon be using material you learned in ENGRD 2700 or an equivalent class, in units on inventory management and statistics. This question will help you remember and review some of this material. If you find this question difficult, that is a sign that you may need to budget extra time in the upcoming weeks to refresh other concepts from ENGRD 2700.

X and Y are two independent normal random variables, each with mean 0 and variance 1. Z=a*X + b*Y, where a and b are two real numbers. Probabilities may be computed using these normal distribution tables, the NORM.DIST function in Excel, or any other method you are comfortable with from ENGRD 2700 or equivalent. Provide answers to each of the following calculations in your pdf file:

(a) Calculate E[Z]
(b) Calculate Var[Z]
(c) Calculate Cov[Z,X]
(d) Calculate P(X>1)
(e) Calculate P(X>1 and Y<0)