**Table of Content**

1. Summary of the data

2. Describe relational schema, draw ER Diagram.

3. Describe start schema, draw ER Diagram.

* calendar dimension table
* customers dimension table
* sales representative employees dimension table
* productLine dimension table
* products dimension table
* shippedOrders fact table: calendar, customer, employee, product key, productline

4. App for analysis:

* Select employee/productLine then print the sales of each month
* Select employee/productLine then print the sales of each quarter
* Sales of each productLine per location
* Draw diagrams
* Pie chart to compare sales of each product line each quarter

**Methods**: steps to create databases locally

1. Create two new schemas: pinnacle\_db and pinnacle\_wh
2. Run DemoAppMain.py
3. If the data for pinnacle\_db have not been created, push the button ‘Initialize Operational Database’
4. Click the button ‘Perform ETL for Warehouse’ to get data from pinnacle\_db to pinnacle\_wh
5. The buttons “Employee’s Performance” and “Productlines' Performance” will populate the dashboard for sales performance of each employee and each product line.

NOTE: Change the two .ini files if needed

**Application:** Run DemoAppMain.py

1. **Summary of the data**

For our data, we use the **classicmodels** database from MySQL sample databases. The data source can be found using this link: [MySQL Sample Database (mysqltutorial.org)](https://www.mysqltutorial.org/mysql-sample-database.aspx). Initially, we did some ETL to the operational database by deleting some irrelevant information in mySQL Workbench. For example, the productlines table contained columns like text descriptions, html descriptions and images of each product line. We cannot do meaningful analysis on such descriptions, and both html descriptions and image columns only contain null values anyways. So, we decided to drop the entire productlines table, as well as any foreign keys associated with the table. We also deleted columns from other tables, such as credit limit from the customers table, phone extension from the employees table, comments from the order table.

To briefly summarize our operational database, we have the “customerNumber” as the primary key, customers’ names, phone numbers, and addresses in the customer table. It has a foreign key “salesRepEmployeeNumber” linked to the employees. The employees have names, email addresses, office code, which links to the offices table, and job titles. Since job positions are hierarchical, there is a “reports to” foreign key that links to the primary key “employeeNumber” in the table itself. The orderdetails table contains the primary key “orderNumber'', foreign key “productCode'', as well as quantity and price per item. Similarly, the orders table contains dates and status, and the foreign key “customerNumber”. The payments table uses “customerNumber” as the primary key, and has columns checkNumber, payment date and the payment amount. Last but not least, the products table uses “productCode” as the primary key, and multiple feature columns. The major focus of that table should be productLine, quantityInStock, and buyPrice.

1. **Operational Database**

**EER Diagram before (Taken from the data source):**

**Diagram

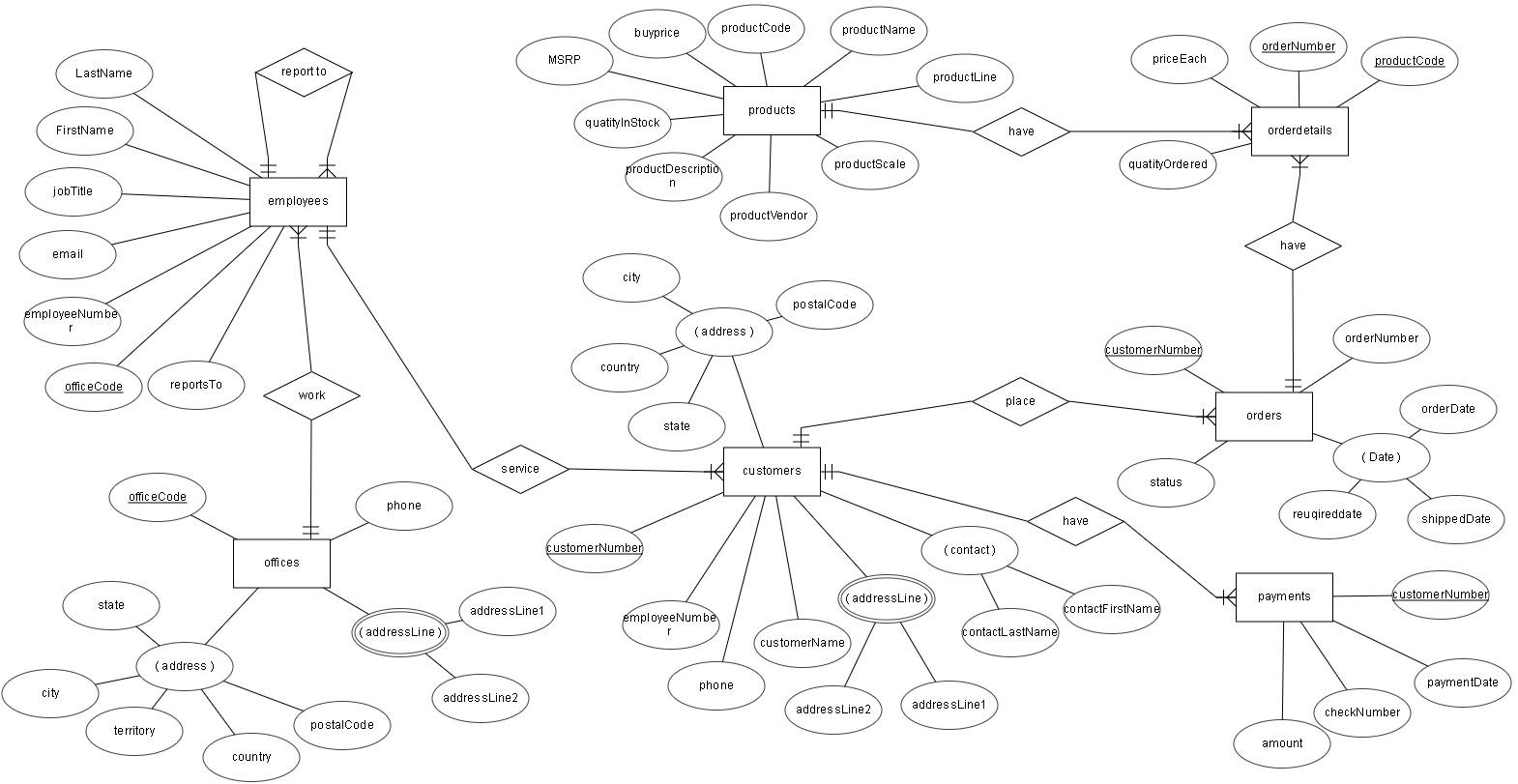
Description automatically generated with medium confidence**

**EER Diagram after dropping unnecessary columns:**

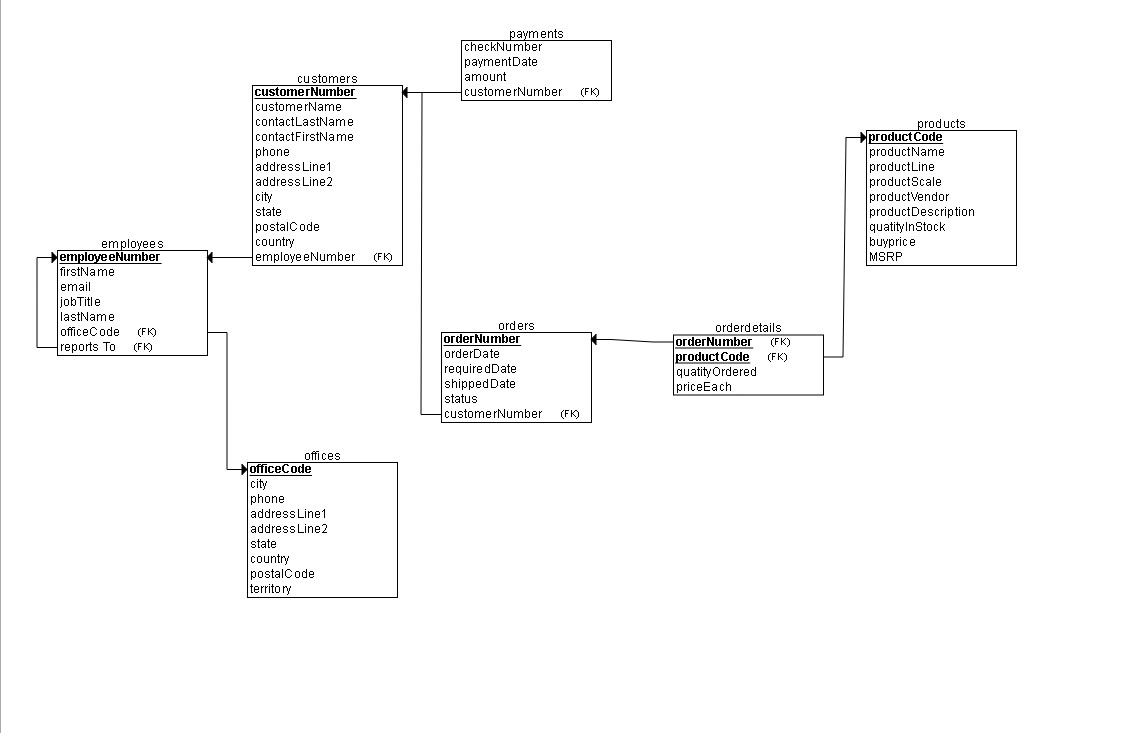
**Diagram

Description automatically generated**

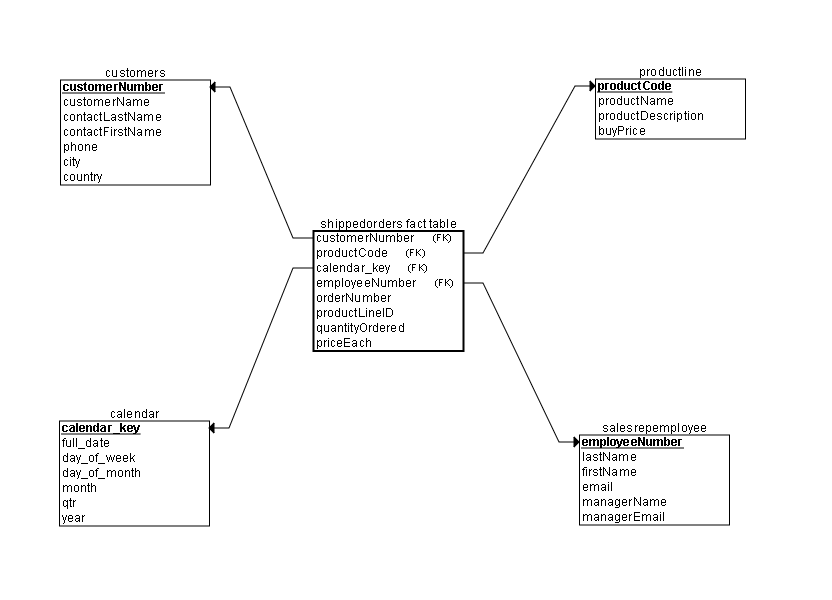
**ER diagram in ERDPlus**

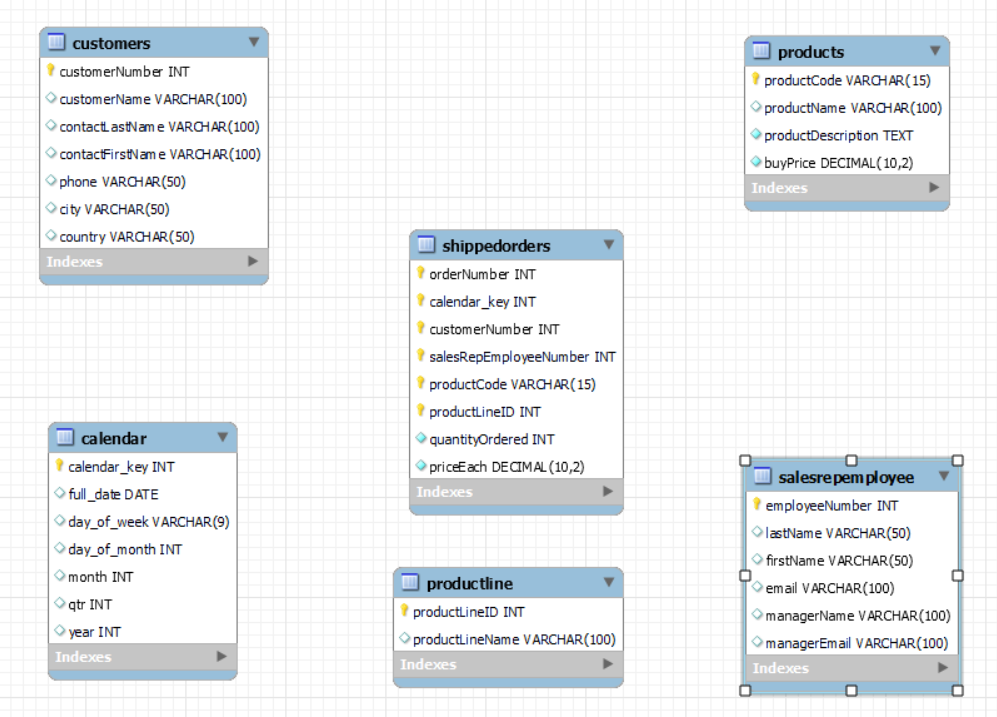
****

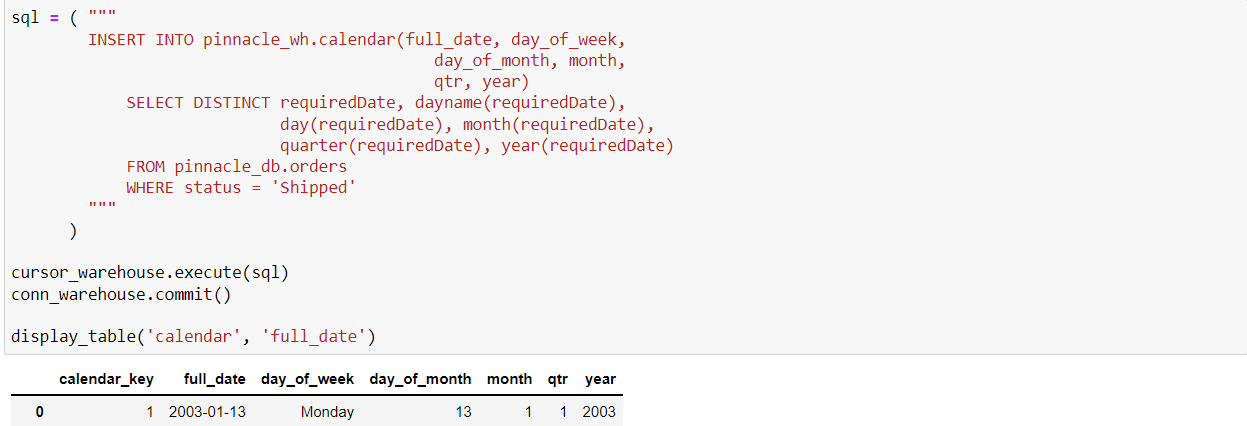
**Relational Schema in ERDPlus**

****

1. **Data Warehouse/Star Schema**

****

****

****

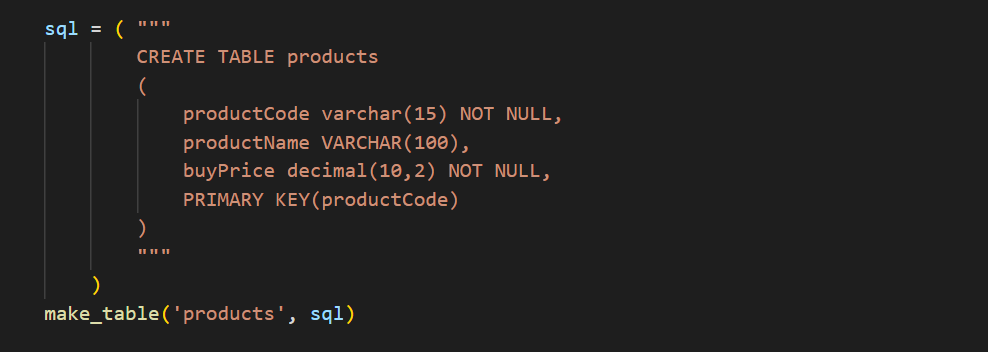
For our analytical data warehouse, the first dimension table we made was the calendar table. We generated a primary key by incrementing automatically. We used the orders table from before, and we only selected the required dates of the shipped products. In addition, we broke up the date information into several columns like day of the week, quarter, and year.



Remember that the employees table was self-referred, such that each sales representative reports to a manager. For our sales representatives employee dimension table, we replaced the “reportsTo” foreign keys with the managers’ first and last names, and their respective email addresses.



The productLine table is self-explanatory, where we just made a primary key for each product line.



For the products table, we filtered out useless information from the relational table, and we were left with productCode, productName, and most importantly buyPrice. Notice that we only selected products that were actually ordered when constructing this table.

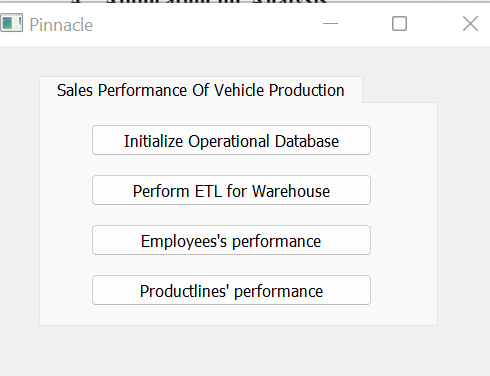


For the customers table, we filtered out addresses, and we were left with city and country as our geographic data. In addition, we were only interested in customers that were helped by sales representatives.

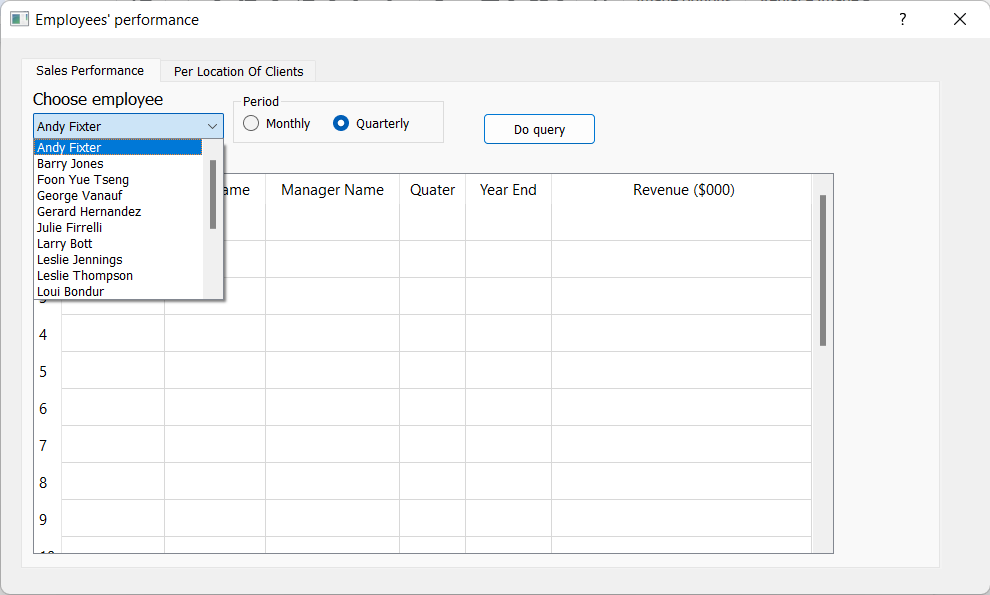


Lastly, for the sales fact table, we joined all the dimension tables described above. At the end, it consisted of all the primary keys and two target columns: quantityOrdered and PriceEach.

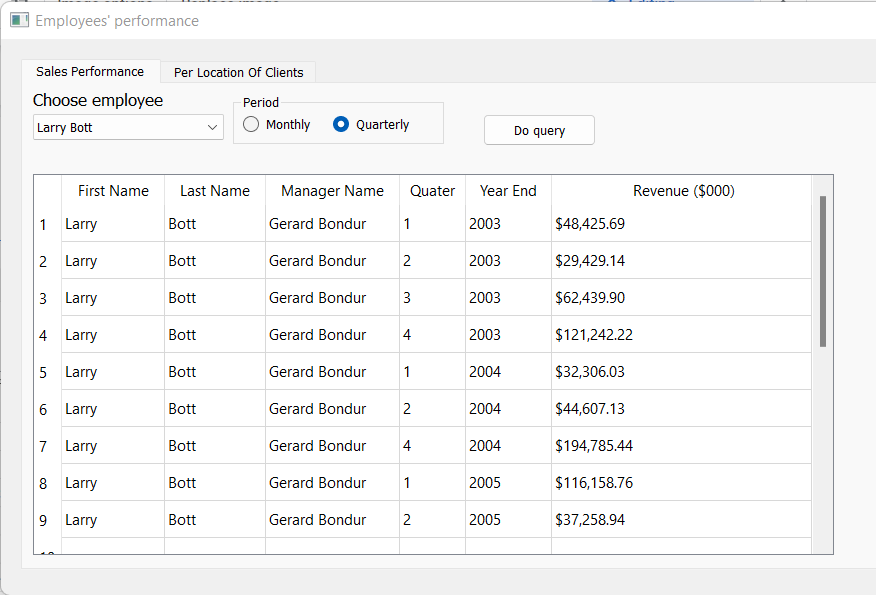
1. **Application for Analysis**

****

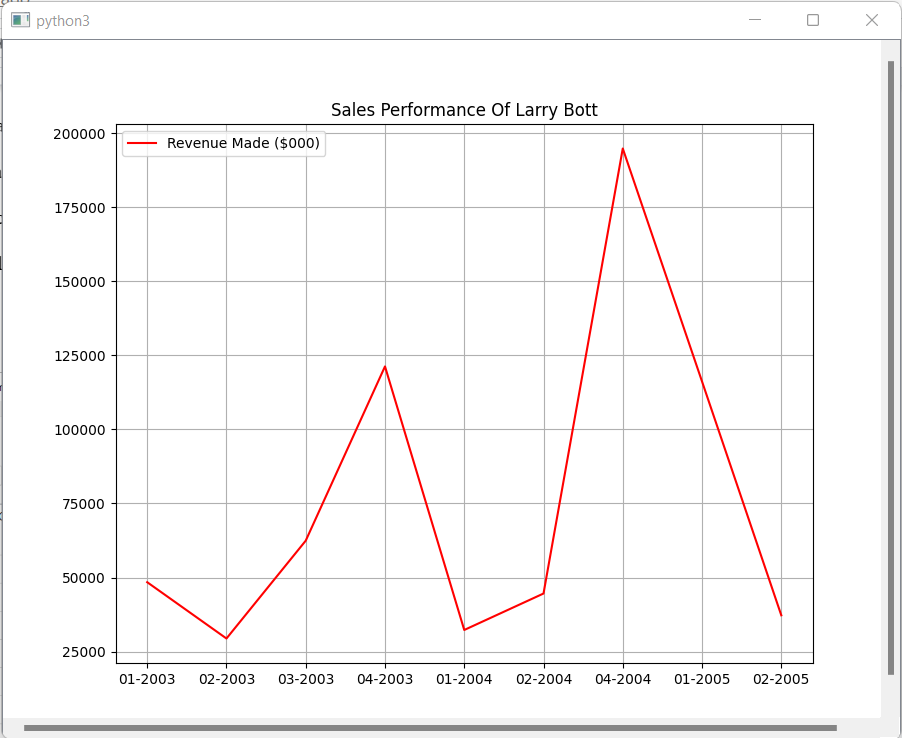
This is the main menu of our GUI application. There are four buttons that we can click on, and the last two take us to a different interface. One purpose of our application is to analyze the sales representatives through sales records and performance graphs, and determine whether they deserve a raise or layoff. Another purpose is to analyze the product lines through sales records and performance graphs, and determine whether the trend is going up or down. We have also implemented a pie chart, where we can select a particular country and year to analyze the proportions of product lines. The goal is to find out where we should allocate the resources for different product lines as the company by examining the pie charts.



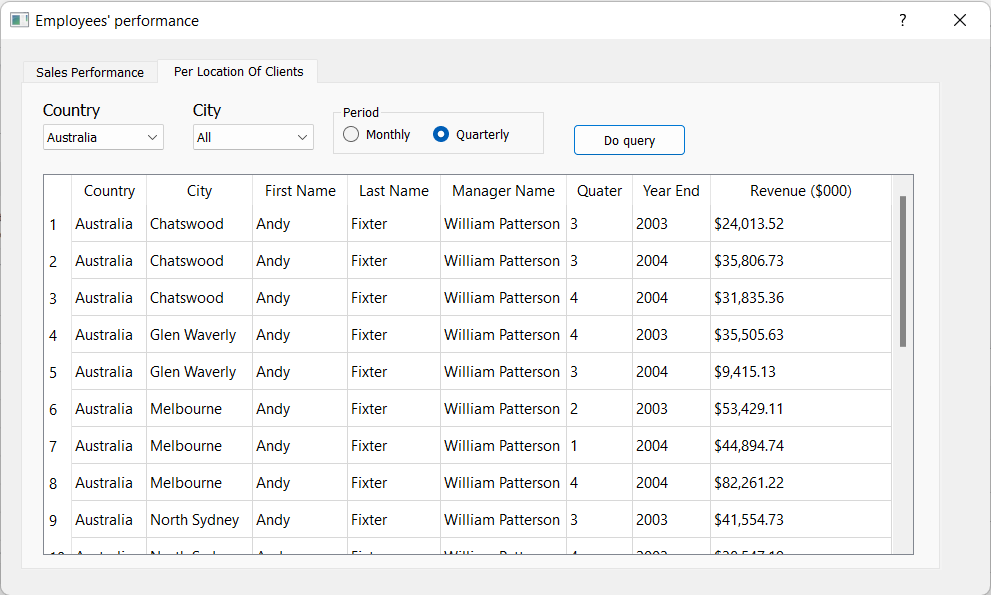
In the Employees’ performance UI window, we can see that there are four major components in the first tab. They are a drop-down menu that contains all employee names alphabetically, a radio button that lets us select the time period, a query button and a table that displays the query results. Let’s say that we want to see how Larry Bott did in terms of sales quarterly.



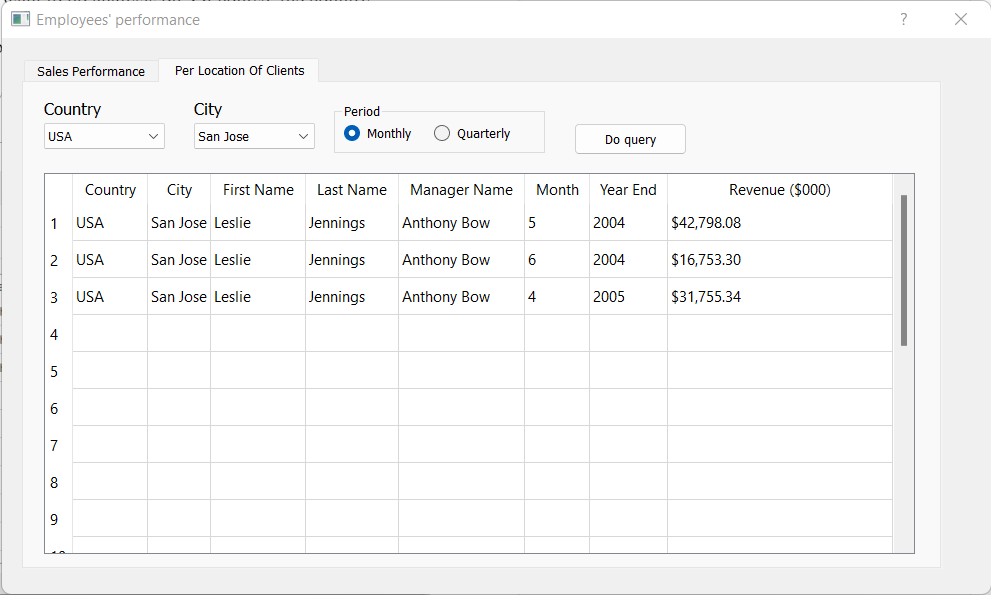
These are the query results, encapsulated in the table above. When we were writing the code, we ordered it by first name, last name, year, and quarter. Since we are only inquiring about one person, we only need to focus on the year and the quarter. Notice how 2003 comes before 2004 and 2005, then quarters are ordered numerically. There is some implied information from this table as well. We can see that Larry Bott either didn’t work or didn’t generate any revenue in the third quarter of 2004. Or simply because the data didn’t get collected. Since the majority of our data only ranges from 2003 to 2005, there is no way to tell whether an employee worked before 2003 or stopped working after 2005.

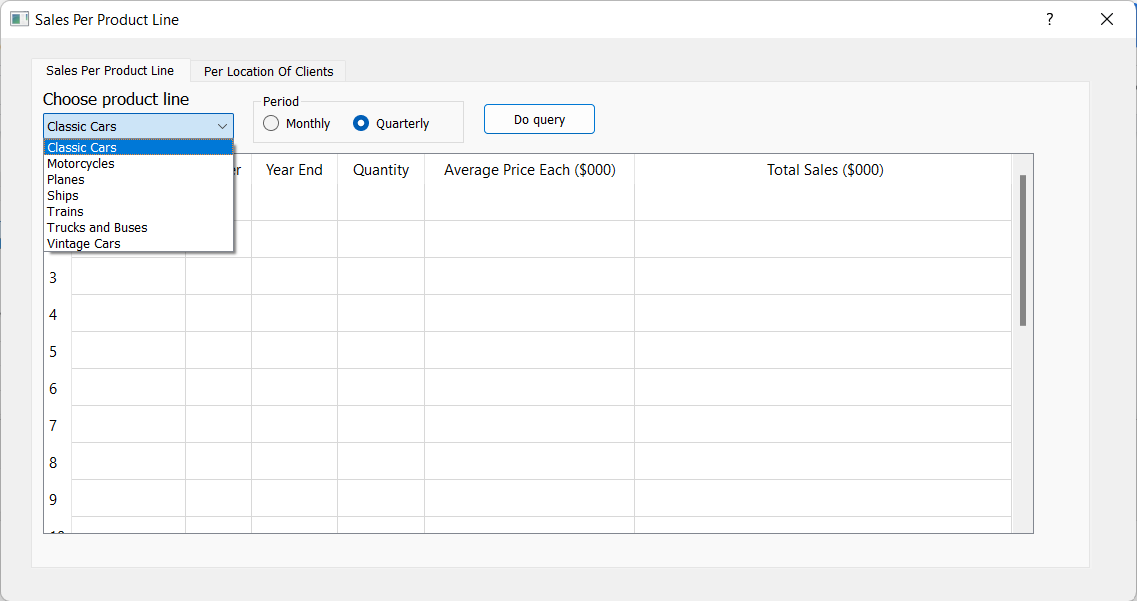


From the graph above, we can see that Larry Bott peaked in the 4th quarter of 2004 and started going downhill in 2005.



This is the second tab of Employees’ performance. Now there are two drop-down menus that let us choose the location we want to do analysis on. Or we can select “all” to see every city in that country. The country and city menus are both in alphabetical order. Let’s say that we want to find out about employees’ monthly revenues in San Jose, USA:





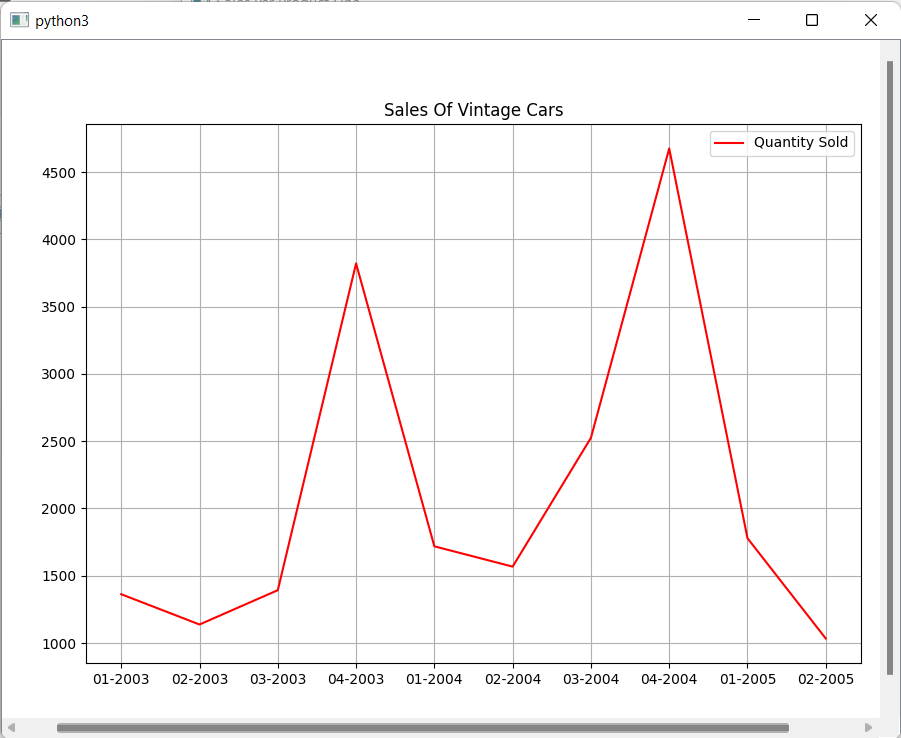
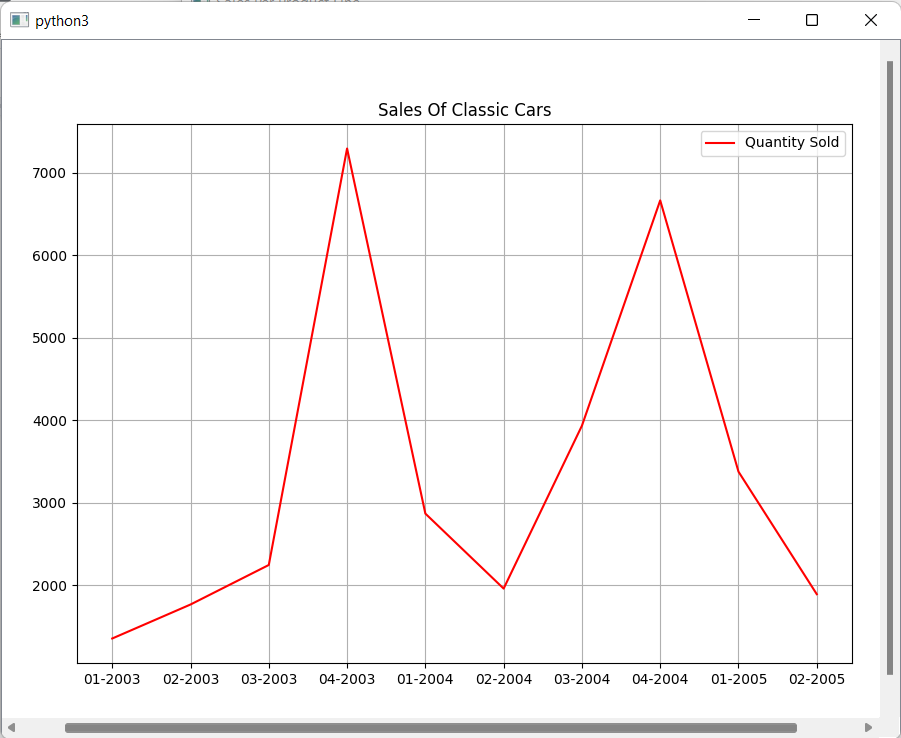
Now let’s turn our attention to the Product Lines’ performance UI window. The structure is the same as before, a drop-down menu selecting the product line, time period radio button, query button and table. Let us try to compare Classic Cars and Vintage Cars in our analyses, to find out which is more popular and profitable.

Graphical user interface, table

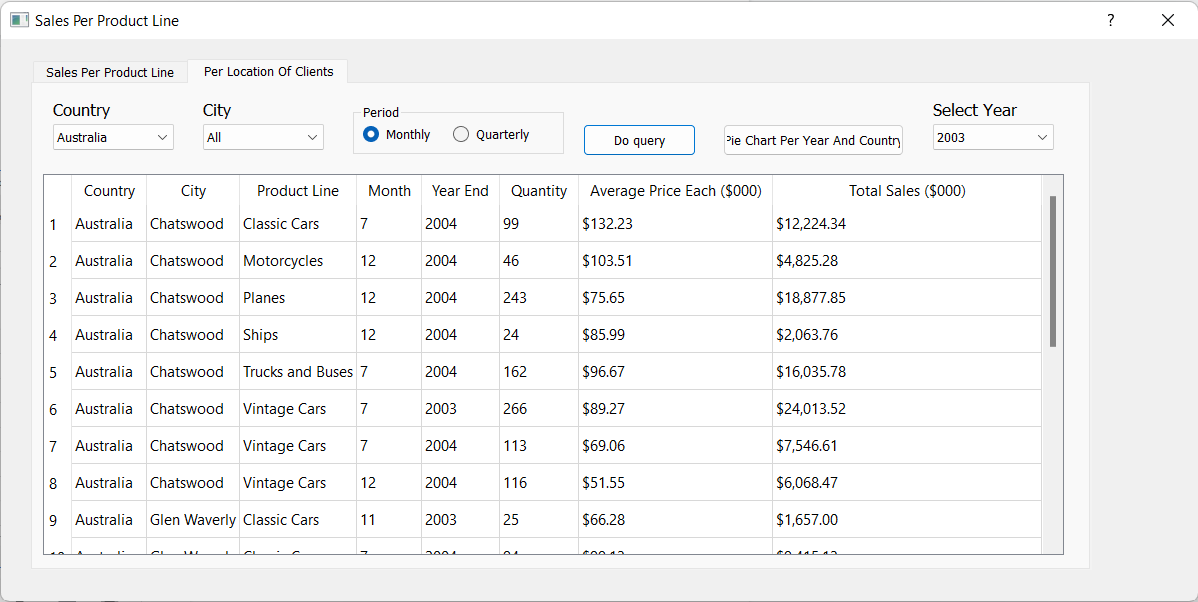
Description automatically generated

Graphical user interface, table

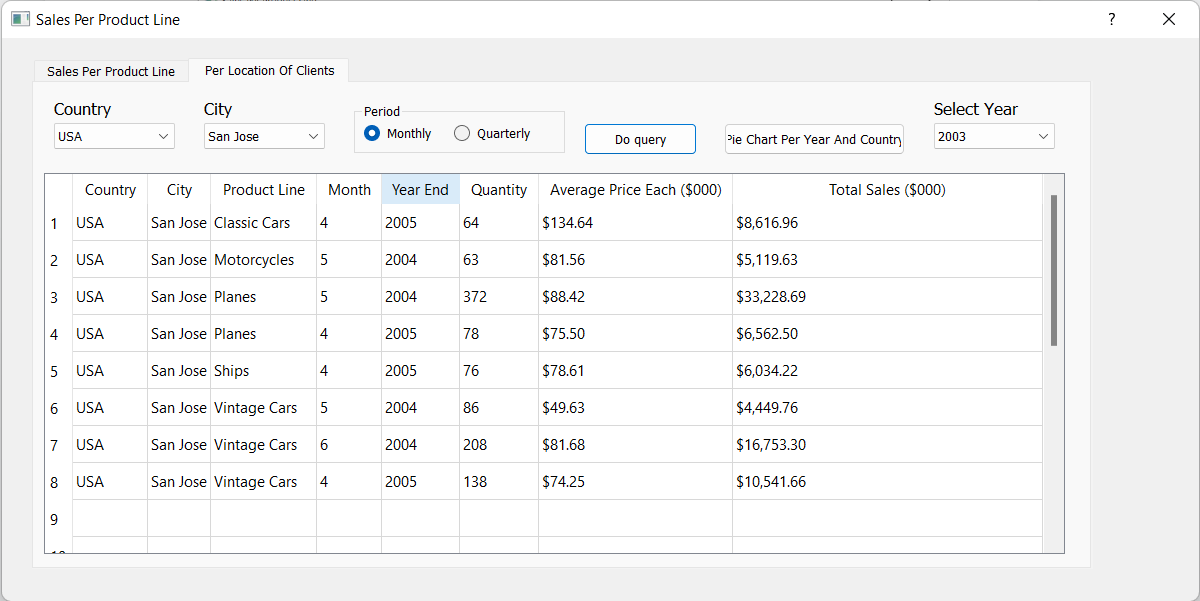
Description automatically generated

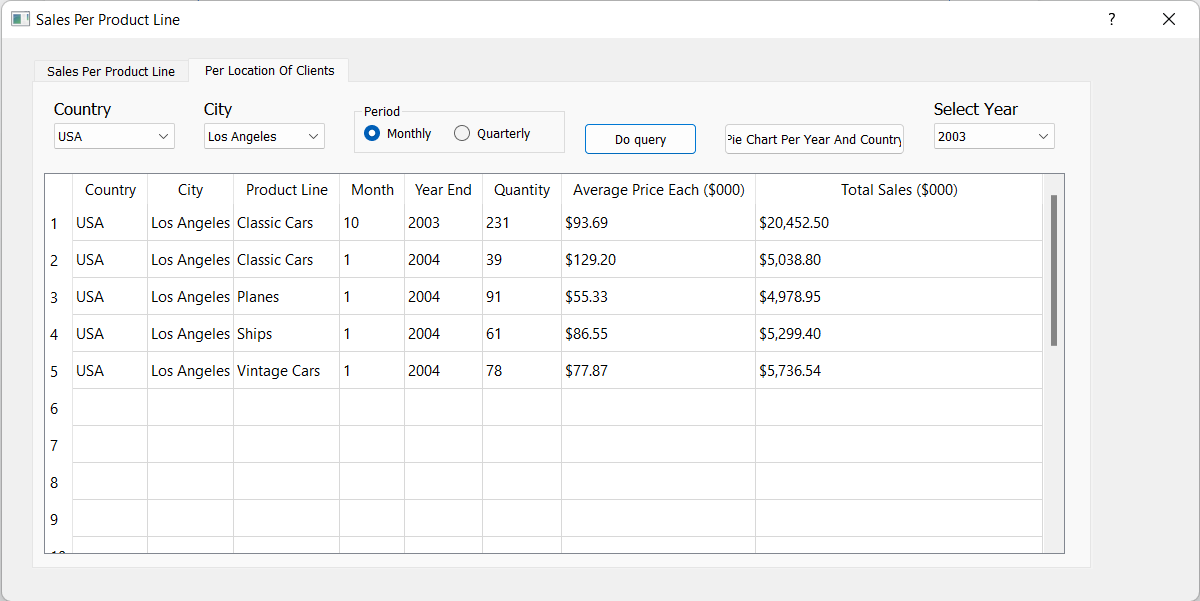


It is hard to compare the two since both graphs look the same visually (except for the y-axis), but it becomes much clearer once we look at the query tables. Not only do Classic Cars have a higher average price each than Vintage Cars, but they also have higher quantities sold across all quarters and years. In the end, the total sales of Classic Cars are clearly far larger than the total sales of Vintage Cars. This can also be confirmed by common sense, that the vintage car parts are outdated, and people favor new cars over vintage cars.

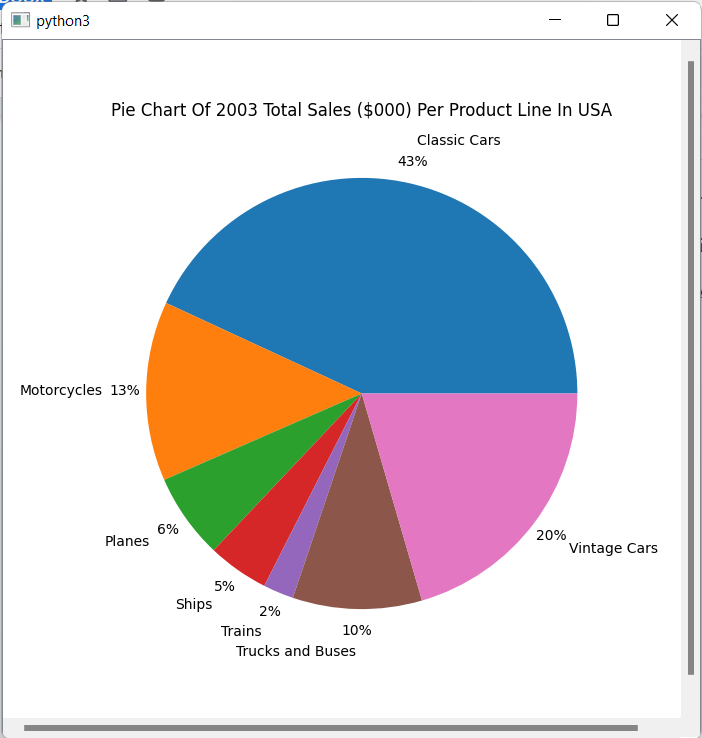


The second tab of Product Lines’ performance shares the same functionality as the second tab of Employees’ performance, with the addition of a pie chart button and an option to select the year for the pie chart. Again, we order by Product Line alphabetically, then year and quarter/month chronologically. All we have to do is select the location we intend to do analysis on. Let’s say we want to compare how the sales of the San Jose office is doing vs. the Los Angeles office, monthly:

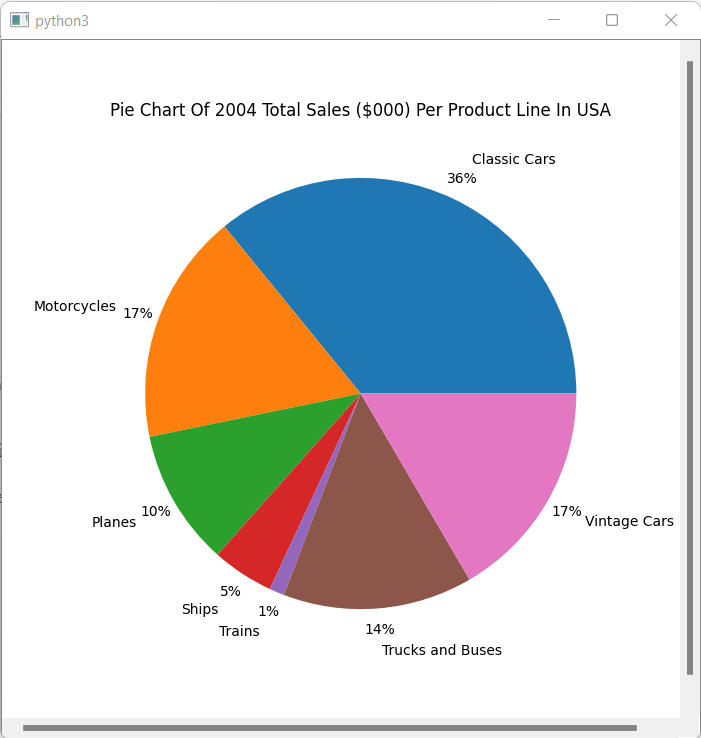




Well, this is straightforward even without graphs. The San Jose office is clearly more profitable than the Los Angeles office. Also, the San Jose office sold a lot more different types of products. We can only make some educated guesses, maybe the San Jose office is a big headquarters and maybe there are more affluent people living in San Jose than in LA.



This pie chart pops up if we click the “Pie Chart per Year and Country” button, given we have already selected the country and year we desired. Let’s compare the total sales per product line in the USA in 2003 and the total sales per product line in the USA in 2004.



There are definitely some noticeable changes in the two pie charts. First of all, both classic cars sales and vintage cars sales declined from 43% to 36% and from 20% to 17%, respectively. Instead, motorcycles, trucks and buses, and planes increased in sales going from 2003 to 2004. There can be multiple circulations drawn from this observation. One possible explanation is that the car prices have gone up in 2004 and less people are willing to buy cars that year. Another possible factor is the economic state that changes from year to year.