Yi (Isabel) Pan
Written on 03/2015
Updated on 12/2018

ACCESS SQL PROJECT

1) The business I created the database for is a Music Instruments store. Initially I wanted to create a database for "Accent Music" which is a famous music instruments store in Delaware but after contacting them through email I found out that they already have a well-developed database, so instead I decided to create my own business based on "Accent Music".

The name of my business is MusikU, it's a merchandising business that sells musical instruments. Since I externally procure the inventory, the business deals with two sets of external parties; the vendors and customers. The business has multiple vendors, each supplying a different line of products. My inventory consists of the following categories:

1. Bass Instruments
2. String Instruments
3. Woodwinds
4. Keyboards

The database is necessary to keep an electronic record of all the business transactions. The item table holds the entire inventory (both in the warehouse and on the shelves), specifying the item code, description, item type, standard selling price and quantity on hand. Since all of the products has barcodes attached to them, every time make a sale the electronic record is automatically updated, giving the data in real time that is used to make promptly business decisions. For instance, the quantity on hand tells how much of an item should be ordered, reducing the need for a physical count of the inventory. The business needed an integrated information system that would connect the different functional areas of business. The database links the activities of the different departments, for example, the item table is connected to

the vendors table via purchaeline and purchase. It's the responsibility of the supply chain department to procure goods from the appropriate vendors but once they make the purchase, invoices are made by the accounting department, the purchaseLine and purchase holds important accounting data such as price, date of purchase, employees and vendors involved etc.

The business also wants to keep track of the items sold and the customers they are sold to. The invoiceLine states the Invoice total as one of the attributes of the table which could be compared to the standard price to find out if any discounts have been given to the customer. This information is necessary to track any fraudulent activities. Employees are assigned different tolerance levels; a limit on the maximum discount an employee can authorize or the amount of purchase he can be involved in, so if a transaction exceeds the maximum amount of discount, it can be inquired about to ensure fair business practices. The database also enables to keep track of the returns. It's important for the business to know which items have the most returns and why. For instance, if I noticed that most of the returns are because of poor quality of the product, I can switch to a different vendor or if the reason is "different from description", I can improve website descriptions.

The database integrates the warehouse, accounting, Sales and distribution departments; assisting in the smooth flow of business processes across the functional areas of business. When the sales department receive an order, they would have real time data from the warehouse to make sure they have enough on hand to accept the order. The invoiveLine table contains revenue related data while the the purchaseLine contains data related to Cost of Sales which can be combined together using SQL queries to produce weekly/monthly/ yearly financial reports. Both the tables contain dates for every transaction, I can make financial reports and do financial ratios even for each day to assess the profitability of the business.

Finally I am interested in the demographics of the customers which can be used to do targeted marketing.

Yi (Isabel) Pan
Written on 03/2015
Updated on 12/2018

2) The database helps the company look at how their business is run. For example, the company needs to know where to supply their inventory so they must keep track of their vendors and the transactions they make with them when they purchase instruments. They need to know how much is expensed on supplying inventory as well as how much is purchased so they know how much is available to sell. Often, instruments are purchased from vendors in bulk as well so it is beneficial to use the data to see which vendors supply the best quality for price for which instruments. This way, the company can decide on cost-efficient ways of purchasing instruments. The database will record which vendors the company interacts with, the price and quantity of inventory, and the instruments purchased, information which will all come from the purchase order obtained during the transaction.

Once the company has inventory, it can now sell them to customers. The company wants to record and track transactions with customers to calculate the revenue they are bringing in. They can also use the data to analyze which instruments are most popular, expensive to supply, and profitable for the company. When customers are dissatisfied with their purchases, they will make returns and the database will record these transactions to see how much inventory is returned and how much money has to be returned to the customer. All this data will come from the sale invoices from customers as well as return receipts.

The company has many employees that take care of purchases, returns, and sales. Employees are part of different departments and some manage other employees. The database will keep track of how much each employee is paid (salary), what transactions employees are responsible for and how employees are managed and other employee information. The data will help management see if there are any trends in how employees are performing (for example, one employee may be especially good at sales) and will

help in decisions regarding bonus systems or commissions. This data will come from the employee's working papers and background checks.

3)
ITEM: This table describes the types of instruments the company has on hand. This is important to keep track of how much is available for the company to sell. It has a many to one relationship with the INVOICELINE table and PURCHASELINE table.

VENDOR: This table describes who the company buys the instruments from. This is important to track where the company gets their inventory supplied. It has a one to many relationship with the PURCHASE table.

EMPLOYEE: This table lists all the employees of the company and other information relating to them such as salary and departments. This is important to track how employees are functioning in the company. This table also has a unary relationship with itself, one to many relationship. It also has a one to many relationship with the SALE and PURCHASE tables.

CUSTOMER: This table lists the customers who have bought at least one instrument from the company and their information. This is important to see where I am getting revenues from. This table has a one to many relationship with the SALE table.

RETURNS: This table lists the transactions where customers return instruments to the company. This is important to track because the company needs to record changes in inventory and revenue. This table has a one to many relationship with the SALE table.

PURCHASE: This table records what transaction existed between the vendor and company. It is important to track purchases because these are responsible for expenses. This table has a one to many relationship with the PURCHASE LINE table.

PURCHASELINE: This is an associative table that connects the ITEM and PURCHASE table because there is a many to many relationship between the two. The relationship this table has with the other two are many to one. It describes what items where bought in a purchase, the price and quantity. This is important to track because the company often buys in bulk and they need to record the amount of expenses incurred and inventory bought.

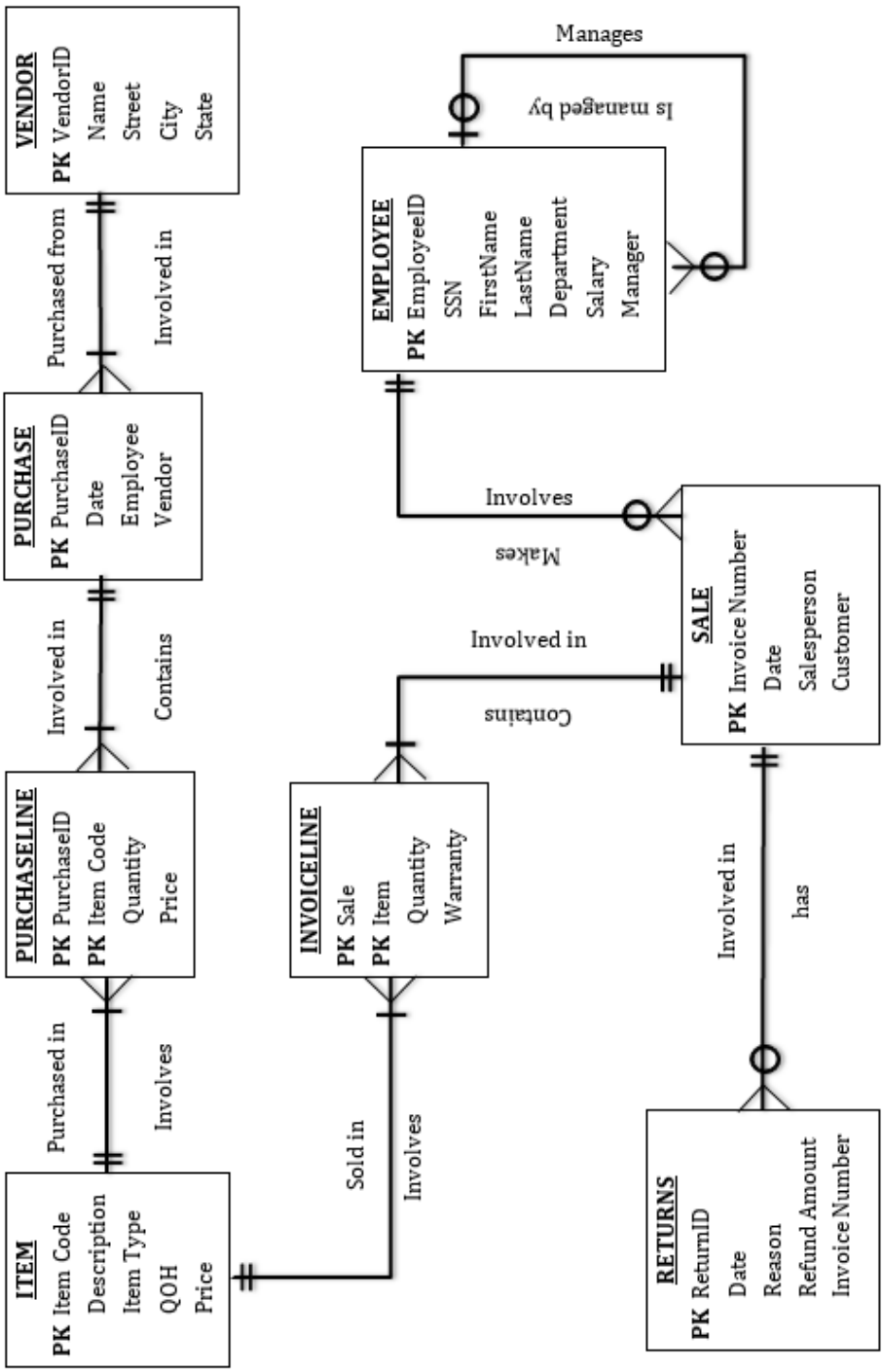
SALE: This table records what transaction existed between the customer and company. It is important to track sales because it is where the company gets its revenue.

INVOICELINE: This is an associative table that connects the ITEM and SALE table because there is a

many to many relationship between the two. This table has a one to many relationship with the other two tables. It describes what items were sold in a sale, at what price and quantity, as well as the warranty that is specific to each sale.

4. ER Diagram

Yi (Isabel) Pan
Written on 03/2015
Updated on 12/2018



Yi (Isabel) Pan
Written on 03/2015
Updated on 12/2018

5.

Query 1: Sales revenue for December of 2012.

```
SELECT Sum(item.price*Invoiceline.quantity) AS SaleRevenue
FROM Sale INNER JOIN (item INNER JOIN invoiceline ON item.itemcode = invoiceline.item)
ON Sale.InvoiceNumber = invoiceline.Sale
where sale.date >= #12/01/2012# and sale.date < #1/01/2013#;
```

This query will help give the managers a prediction of how much they will sell this December, also this query will come in handy if they are curious of past month revenues. This query would be useful to run at the beginning and end of every month.

Query 2: Cost of goods sold for December of 2012.

```
SELECT Sum(invoiceline.quantity*Purchaseline.price) AS COGS
FROM Sale INNER JOIN ((Item INNER JOIN InvoiceLine ON Item.ItemCode =
InvoiceLine.Item) INNER JOIN PurchaseLine ON Item.ItemCode = PurchaseLine.ItemCode) ON
Sale.InvoiceNumber = InvoiceLine.Sale
WHERE Sale.date>=#12/1/2012# And Sale.date<#1/1/2013#;
```

This query has similar purposes as the first query and would be good to use these side by side at the beginning and end of every month.

Query 3: List of employees that sold above $40,000.

```
SELECT Employee.EmployeeID, Employee.FirstName, Employee.LastName,
Sum(InvoiceLine.InvoiceTotal) AS TotalSales
FROM (Employee INNER JOIN Sale ON Employee.EmployeeID = Sale.Salesperson) INNER
JOIN InvoiceLine ON Sale.InvoiceNumber = InvoiceLine.Sale
GROUP BY Employee.EmployeeID, Employee.FirstName, Employee.LastName
HAVING Sum(InvoiceLine.InvoiceTotal)>40000;
```

This query would be important for finding which of employees sell the most. The managers could give rewards to employees once they reach this amount. This query could be run at the end of every month.

Yi (Isabel) Pan
Written on 03/2015
Updated on 12/2018

Query 4: Lists the employees that live in Delaware

```
SELECT FirstName, Lastname
FROM Customer
WHERE State = "DE";
```

FInding the employees that live in a certain area could come in handy for the manager or marketing people for when they want to send out flyers to their current customers who live nearby.

Query 5: Percentage of returns

```
SELECT Count(Returns.InvoiceNumber)/Count(Sale.InvoiceNumber) AS
Percent_of_Sales_Returned
FROM Sale LEFT JOIN Returns ON Sale.InvoiceNumber = Returns.InvoiceNumber;
```

Finding the percentage of returns will help accounting and the upper management predict future revenue. This query would be helpful to run monthly and to include on any internal financial reports.

Query 6: Average Salary for employees in the Sale department

```
SELECT Department, avg(Salary) AS AVG_Salary
FROM Employee
GROUP BY Department
having Department = "Sale"
```

This query would be useful for HR so that when they are trying to hire or are in interviews, they can give the potential employee an estimate of how much he will be paid.

Query 7: Item with highest gross margin

```
SELECT item.description, item.itemtype, item.qoh, item.price-purchaseline.price AS
Gross_Margin
FROM Item INNER JOIN Purchaseline ON item.itemcode = Purchaseline.itemcode
WHERE item.price-purchaseline.price = (SELECT Max(Item.Price-PurchaseLine.Price) As
GrossMargin
From Item inner join Purchaseline on item.itemcode = Purchaseline.itemcode);
```

Knowing which item provides the greatest gross margin will help employees to know which items to push the most onto customers. This query should be ran anytime products are added or prices are changed.

Query 8: How many items each instrument type are sold.

```
SELECT item.itemtype, count(invoiceline.item) AS Number_Sold
FROM item INNER JOIN invoiceline ON item.itemcode = invoiceline.item
GROUP BY item.itemtype
having item.itemtype = "bass"
```

This query helps tell the management and the advertising department how well the itemtype is being sold and could use this to compare versus other item types. Giving the company an idea of which type is their best seller.

Query 9: Finding retained earning

```
SELECT (SELECT Sum(InvoiceLine.InvoiceTotal) AS SumOfInvoiceTotal FROM InvoiceLine)-
(SELECT Sum([PurchaseLine]![Price]*[PurchaseLine]![Quantity]) AS Expr1 From PurchaseLine)
AS RentainedEarnings
FROM (SELECT COUNT(*) FROM INVOICELINE);
```

Finding the retained earnings is useful for any business to find where their business stands financially. It is also needed for the accountants for them to make their different reports. This query would be used on a monthly basis and also as requested.

Query 10: Which employee was responsible for customer Bob loblaw on July 4th 2010.

```
SELECT Employee.FirstName, Employee.LastName
FROM Employee INNER JOIN (Customer INNER JOIN Sale ON Customer.CustomerID =
Sale.Customer) ON Employee.EmployeeID = Sale.Salesperson
WHERE Customer.FirstName="Bob" AND Customer.LastName="Loblaw" AND
Sale.Date=#7/4/2010#;
```

A query like this could come in handy when a customer complains or compliments his sales representative. This query would be ran on an on-demand basis.

Query 11: List of items returned because they were broken.

Yi (Isabel) Pan
Written on 03/2015
Updated on 12/2018

```
SELECT Item.Description, Returns.Reason, item.price
FROM Item INNER JOIN ((Sale INNER JOIN Returns ON Sale.InvoiceNumber =
Returns.InvoiceNumber) INNER JOIN InvoiceLine ON Sale.InvoiceNumber = InvoiceLine.Sale)
ON Item.ItemCode = InvoiceLine.Item
GROUP BY Item.Description, Returns.Reason, item.price
HAVING Returns.reason = "Broken"
```

This query can help the company see if there is a trend with products being returned that were broken. This query would be benificial to run at the very least at the end of each month and before the company makes any large orders to replace their stock.

Query 12: List the Customer service managers and how many employees are under each.

```
SELECT manager, Department, count(manager) AS number_of_employees
FROM employee
GROUP BY manager, Department
HAVING department = "Customer Service";
```

This will help the company get an idea whether a certain department is in need of more or less managers. This query would not have to be run often, but would be beneficial to look at whenever you are about to hire new employees or thinking about giving someone a raise or promotion.

Query13: Customer who has purchased more than $15000 from the company

```
SELECT Customer.FirstName, Customer.LastName, Sum(InvoiceLine.InvoiceTotal) AS
TotalPurchased
FROM (Customer INNER JOIN Sale ON Customer.CustomerID = Sale.Customer) INNER JOIN
InvoiceLine ON Sale.InvoiceNumber = InvoiceLine.Sale
GROUP BY Customer.FirstName, Customer.LastName
HAVING Sum(InvoiceLine.InvoiceTotal)>=15000;
```

Finding the customers who have purchased more than $15000 worth of goods would be useful for marketting department to see who most loyal customers are. They could use this info to send the customers special deals and other information regarding the company. This query could be ran any time the company is about to have a special sale or have a special event for these customers.

Query14: List the total salary of each department.

Yi (Isabel) Pan
Written on 03/2015
Updated on 12/2018

Select Department, sum(Salary) as Cost From Employee Group by Department;

This will help the company understand how much each department within the company costs based on the number of employees and their salaries. This query could be used monthly and could be ran whenever the accountants make their reports.

Query15: List the employee information for each Return

```
SELECT Returns.ReturnID, Returns.Reason, Returns.Date, Employee.EmployeeID,
Employee.FirstName, Employee.LastName
FROM (Employee INNER JOIN Sale ON Employee.EmployeeID = Sale.Salesperson) INNER
JOIN Returns ON Sale.InvoiceNumber = Returns.InvoiceNumber;
```

This will help us find the employee that is responsible for the return. The manager can give the employee punishment base on the information.This query would be ran monthly and while evaluating employees.

Using Subquery: Q7, Q9
Two-Table-Query: Q5, Q7, Q8
Three-Table-Query: Q1, Q2, Q3, Q10, Q11, Q13, Q15
GROUPBY And HAVING: Q3, Q6, Q8, Q11, Q12, Q13
Other: Q4, Q14