

Forum:[Hack Forums](#)

Thread: [\[SQL\] Storing transactions in a database](#)

Chappie - February 3, 2016, 5:35 pm

Well, I am using a Database in a C# project. I decided to take my question here because my issue is nothing to do with C# in general. I want to store transactions in a SQL database.

Example in pseudo visual
[Table dbo.Customers]

[Product c1]	[Transaction ID c2]	[Date c3]	[Customer ID]
Phone	001	2/3/16	100
Pepsi	001	2/3/16	100
Bagel	002	2/4/16	100
Apple	002	2/4/16	100
Coffee	003	2/5/16	100

[/Table]

Now Customer 100 comes in and complains the phone he bought for his daughters birthday is broken. Sales person needs to pull up his past purchases given the transaction ID on the receipt. I want it to pull up everything he bought so we will query every item with the ID of 001 - which is a Pepsi and a phone.

Now this is fine because I have 5 rows of data. However over time would this become an issue? With over a thousand transactions and each product stored individually?

If on average a customer bought 3 items and I have over 1000 transactions, then I would have over 3000 rows of data. 10,000 == 30,000 and so on. I feel like there is a more efficient way to design the transactions and products, I just am not sure.

Anyny0 - February 3, 2016, 6:03 pm

This would work fine even with tons of data, this is what SQL is built for :)

As for a more efficient way, I would split the transactions into different tables:

The first one would be transaction, with only an id and a date.

The second one would be the items of this transaction, and would contain the quantity of each item as well as its price.

TransactionItem would also have a foreign key to a Transaction and to an Item.

This gives us a result like this:

Code Transaction:

id, date, invoiceNumber?

TransactionItem:

id, Transaction_id, Item_id, ItemQty, ItemPrice

Item:

id, Name, Description, Price

It would not be a huge improvement but would allow you to store less redundant data, such as the transaction date at the moment.

JIM NASIUM'S GYMNASIUM - February 3, 2016, 7:17 pm

^ This is good advice. However, wouldn't it also be redundant to store the `ItemPrice` as well?

Chappie - February 3, 2016, 7:33 pm

Item price and stuff wouldn't need to be in SQL as long as I have the product name or it's ID because I can just pull it from the SQL and compare it to a "pricebook" xml I have.

However, I have another way I'm looking at it and I want your guys' input on it.

Currently, I have set it up to store transactions in an XML file. When a new month is detected it creates a new XML file and saves the transactions for that new month in there and stores the old transaction history file into a library folder - which can be accessed if date specified. Doing this is much easier and I don't have to worry about security in regards to my database. However, how much can XML support if broken up into files based on months?

Ideally 5,000 - 10,000 transactions a month. Can XML support that?

Annabelle Lee - February 3, 2016, 9:15 pm

(Feb 3, 2016 08:33 PM)Peefer Wrote: [\[CITE\]](#) Item price and stuff wouldn't need to be in SQL as long as I have the product name or it's ID because I can just pull it from the SQL and compare it to a "pricebook" xml I have.

However, I have another way I'm looking at it and I want your guys' input on it.

Currently, I have set it up to store transactions in an XML file. When a new month is detected it creates a new XML file and saves the transactions for that new month in there and stores the old transaction history file into a library folder - which can be accessed if date specified. Doing this is much easier and I don't have to worry about security in regards to my database. However, how much can XML support if broken up into files based on months?

Ideally 5,000 - 10,000 transactions a month. Can XML support that?

XML is simply a markup language that organizes the data so it can be easily prased. As it's more or less in a textfile, you don't have to worry about it being supported. As long as you have a way to read it, you'll be fine.

Chappie - February 3, 2016, 9:23 pm

(Feb 3, 2016 10:15 PM)AnnabelleLee Wrote: [\[CITE\]](#) XML is simply a markup language that organizes the data so it can be easily prased. As it's more or less in a textfile, you don't have to worry about it being supported. As long as you have a way to read it, you'll be fine.

I wasn't sure if using XML to store a lot of data would become inefficient after a certain point.

Annabelle Lee - February 3, 2016, 9:31 pm

(Feb 3, 2016 10:23 PM)Peefer Wrote: [\[CITE\]](#) I wasn't sure if using XML to store a lot of data would become inefficient after a certain point.

Personally I believe it'd just be better to keep using SQL for the prices as well, it'd be easier to keep track of and it's designed to handle large sets of data. I'm sure that at a point xml would get more computationally expensive, but don't know how to calculate that point.

Anyny0 - February 4, 2016, 5:40 am

(Feb 3, 2016 08:17 PM)JIM NASIUMS GYMNASIUM Wrote: [\[CITE\]](#) ^ This is good advice. However,

wouldn't it also be redundant to store the `ItemPrice` as well? (Feb 3, 2016 08:33 PM)Peefer Wrote: [\[CITE\]](#) Item price and stuff wouldn't need to be in SQL as long as I have the product name or it's ID because I can just pull it from the SQL and compare it to a "pricebook" xml I have. Grabbing the price at the date of purchase allows us to deal with specials and price changes.
If a customer bought a phone at 100\$ and returns it when the price is now at 150\$, we wouldn't be able to track this change without storing the price at the transaction date.

(Feb 3, 2016 08:33 PM)Peefer Wrote: [\[CITE\]](#) Currently, I have set it up to store transactions in an XML file. When a new month is detected it creates a new XML file and saves the transactions for that new month in there and stores the old transaction history file into a library folder - which can be accessed if date specified. Doing this is much easier and I don't have to worry about security in regards to my database. However, how much can XML support if broken up into files based on months?

Ideally 5,000 - 10,000 transactions a month. Can XML support that? Xml could support that (It would be a huge text file), but I wouldn't recommend saving it this way.

What you could do instead is keeping the SQL model and exporting it daily/weekly/monthly into a readable format such as csv or xml. This would keep you a fast access (Sql) with backups in a readable format (Xml)

You could then potentially delete the old transactions from the database, but I would instead move them into another table as "Legacy" transactions if we ever need them.

(Feb 3, 2016 10:23 PM)Peefer Wrote: [\[CITE\]](#) I wasn't sure if using XML to store a lot of data would become inefficient after a certain point. It is inefficient for us, humans, to go through. We have to use a text editor and search manually for a certain transaction, and if you only have the default text editors (Notepad), the file will be slow to open and edit.