

## Re: Newbie help

**Source:**

<http://www.tech-archive.net/Archive/SQL-Server/microsoft.public.sqlserver.programming/2004-09/5015.html>

---

**From:** Steve (*sedmyer\_at\_indy.rr.com*)

**Date:** 09/23/04

Date: Thu, 23 Sep 2004 02:30:24 GMT

Mike,

You hit the nail on the head (pun intended).

Yes this is exactly what I am attempting to do. I do not however have unlimited resources and therefore the purchase of specialized products for this hobby is out of the question. I did consider creating my own custom engine but decided that was more effort than this hobby deserved since I think I can probably get what I need (turn this screw) using SQL Server (the hammer I have). Granted the manner in which I will use SQL Server (more or less as a flat file) will probably always irk the DBA purists out there. But that is really not a concern of mine.

Thanks to all for the info and insight.

Steve

"Michael D. Long" <michael.d.long-nospam@comcast.net> wrote in message news:u1FbqeEoEHA.3396@tk2msftngp13.phx.gbl...

> *This is ultimately a case where you are getting recommendations for a  
> hammer to turn a screw. Do a little research on time series data archival  
> products, such as OSI PI and the Intellution Data Historian. Industrial  
> data collection poses the problem of storing massive amounts of  
> time-series data (just like your application) that must be analyzed in  
> order to be useful. The collected data must be stored efficiently in  
> order to conserve resources and keep manufacturing costs low. A  
> traditional general purpose DBMS is not the optimal solution to the  
> problem, though such products can be used.*  
>  
> *To efficiently store and retrieve data for your needs, the approach used  
> by time series repositories is best. You only store changed values (as  
> you originally proposed), updating the timestamp of the relevant entry when  
> the data remains unchanged between intervals. This minimizes seek times  
> for retrieval, but does pose a problem that must be addressed during data  
> retrieval. When you request a value for a specific time, it is just as  
> likely not to be found as it is to exist. In the case where the entry  
> doesn't exist, you must interpolate the value. The correct value for your  
> application would be the record having the appropriate symbol with the*

> next earliest timestamp.  
>  
> You can achieve much better performance for such applications with a  
> custom engine than a commercial DBMS, but you need a solid background in  
> data storage and retrieval techniques. Based on experience in this area,  
> unique thought patterns also help a lot.  
>  
> --  
> Michael D. Long  
>  
>  
> "Steve" <sedmyer@indy.rr.com> wrote in message  
> news:f\_K3d.76291\$787.48515@fe2.columbus.rr.com...  
>> Well here is a little bit more back ground on why I do what I do :)  
>>  
>> I am writing a program to gather "real-time" market data on 3000+  
>> symbols. I quoted the word real-time because in this case real-time means  
>> I tick every 15 seconds. For each of these ticks I pull thirty different  
>> values. Once I have gathered a sufficient amount of data (I would like to  
>> keep a six month rolling window of data) I hope to be able to find data  
>> that act as indicators of activity. The next step is obvious, use those  
>> indicators to guide future trading.  
>>  
>> I originally thought I would simply store the data in text files to  
>> conserve space. Further to prevent the file from being too large (and  
>> since I am only interested in trends lasting a day or less) I thought  
>> each day I would create a new file. However, it turns out that a single  
>> day results in +/- 4 million rows of data which is appx. 1 gig on disk.  
>> Once I realized the size of the data I was gathering, and the difficulty  
>> of locating data in that file, it became obvious that I would not be able  
>> to quickly do any type of analysis on a flat file of this size. For that  
>> reason I decided to store the data in a SQL Server DB. Further I decided  
>> that since I am only looking for short term trends, I would continue to  
>> break the data into daily tables in order to minimize query times. So the  
>> programs first task in the morning (on trading days) is to create a table  
>> named for the day (MM-DD-YYYY) with one column for each data element  
>> being gathered. Then as the day goes on all tick data is simply inserted  
>> into the days table in the DB. I was still hoping to minimize the space  
>> required by this data. I have heard (but I do not know if it is true or  
>> not) that in SQL Server a column of type varchar only takes up as much  
>> space as is actually used. Since much of the data I gather does not  
>> change as rapidly as I pull it I felt I could save a considerable amount  
>> of space by not saving any data which has not changed and simply leave  
>> unchanged columns null for that tick. So if I have duplicate data from 1  
>> tick to the next I would save to the DB only those columns which actually  
>> changed. This presented me with a problem though, since most (if not  
>> all) of the records are incomplete (due to the saving of null values in  
>> place of duplicate data) how do I get a complete record for any given  
>> time to use in the analysis. It is for this purpose that I created the  
>> procedure that I posted.  
>>

>> *This DB has no other tables (at this time) than these daily data tables.*  
>> *There are no relationships in this DB and there are no users (other than*  
>> *myself and my application). Therefore data integrity and worries about*  
>> *what users might be able to do to the DB are complete non-issues.*  
>>  
>> *"Joe Celko" <jcelko212@earthlink.net> wrote in message*  
>> *news:eqnUp81nEHA.3520@TK2MSFTNGP15.phx.gbl...*  
>>>> *I was hoping that I might get some feedback on it. <<*  
>>>  
>>> *Always a bad thing to ask for in this newsgroup :)*  
>>>  
>>>> *The purpose of the procedure is to populate a single record [sic]*  
>>> *with the most recent (as compared to the time parameter) non-null values*  
>>> *for each column in the specified table. <<*  
>>>  
>>> *What you are doing is writing metadata tools in SQL and you are not*  
>>> *supposed to do that in an application.*  
>>>  
>>> *The whole idea of this procedure is a violation of basic software*  
>>> *engineering principles. This is **\*\*much\*\*** more fundamental than SQL*  
>>> *programming. Don't you remember all that stuff about cohesion and*  
>>> *coupling?*  
>>>  
>>> *Did you mean "as determined by the time parameter"? I do not see such a*  
>>> *parameter; if it were done properly it would be a temporal data type and*  
>>> *not a string. Dynamic SQL is a sign of failure on the part of the DB*  
>>> *programmer, so he has to let the end user build the system he was not*  
>>> *able to.*  
>>>  
>>> *The right answer is never pass a table name as a parameter. You need to*  
>>> *understand the basic idea of a data model and what a table means in*  
>>> *implementing a data model. Go back to basics. What is a table? A model*  
>>> *of a set of entities or relationships. EACH TABLE SHOULD BE A DIFFERENT*  
>>> *KIND OF ENTITY. What having a generic procedure works equally on*  
>>> *automobiles, octopi or Britney Spear's discology is saying that your*  
>>> *applications a disaster of design.*  
>>>  
>>> *1) This is dangerous because some user can insert pretty much whatever*  
>>> *they wish -- consider the string 'Foobar; DELETE FROM Foobar; SELECT \**  
>>> *FROM Floob' in your statement string.*  
>>>  
>>> *2) It says that you have no idea what you are doing, so you are giving*  
>>> *control of the application to any user, present or future. Remember the*  
>>> *basics of Software Engineering? Modules need weak coupling and strong*  
>>> *cohesion, etc.*  
>>>  
>>> *3) If you have tables with the same structure which represent the same*  
>>> *kind of entities, then your schema is not orthogonal. Look up what*  
>>> *Chris Date has to say about this design flaw.*  
>>>  
>>> *4) You might have failed to tell the difference between data and*

microsoft.public.sqlserver.programming: Re: Newbie help

>>> *meta-data. The SQL engine has routines for that stuff and applications  
>>> do not work at that level, if you want to have any data integrity.  
>>>  
>>> Yes, you can write a program with dynamic SQL to kludge something like  
>>> this. it will last about a year in production and then your data  
>>> integrity is shot.  
>>>  
>>> You use square brackets, global temporary table, TOP and other  
>>> proprietary syntax edven when standard syntax is allowed. Your names  
>>> violate ISO-11179 naming rules with those silly 1960's BASIC prefixes.  
>>>  
>>> --CELKO--  
>>> Please post DDL, so that people do not have to guess what the keys,  
>>> constraints, Declarative Referential Integrity, datatypes, etc. in your  
>>> schema are. Sample data is also a good idea, along with clear  
>>> specifications.  
>>>  
>>>  
>>> \*\*\* Sent via Developersdex <http://www.developersdex.com> \*\*\*  
>>> Don't just participate in USENET...get rewarded for it!  
>>  
>>  
>  
>*