# What is the best strategy for retainment of large data sets?

Asked 16 years, 4 months ago Modified 9 years, 2 months ago

Viewed 755 times

▲

9

▼

I'm leading a project where we'll be recording metrics data. I'd like to retain the data for years. However, I'd also like to keep the primary table from becoming bloated with data that, while necessary for long term trending, isn't required for short term reporting.

What is the best strategy for handling this situation? Simply archive the old data to another table? Or "roll it up" via some consolidation of the data itself (and then store it off to a different table)? Or something else entirely?

*Additional info: we are using SQL Server 2005.*

database-design dataset

Share

Improve this question

Follow

edited Oct 19, 2015 at 10:15

Brian Tompsett - 汤莱恩
5,875 ● 72 ● 61 ● 133

asked Aug 21, 2008 at 12:31

Scott Marlowe
8,025 ● 12 ● 46 ● 52

## 5 Answers

Sorted by: Highest score (default)

▲

4

▼

✓

We use both methods at my work, but slightly different, we keep all sales data in the primary table for 30 days, then at night (part of the nightly jobs) the days sales are rolled up into summaries (n qty of x product sold today ect) in a separate table for reporting reasons, and sales over 30 days are archived into a different database, then once a year (we go on tax years) a new archive database is started. not exactly perfect but..

this way we get the summaries data fast, keep all current sales data at hand and have an unlimited space for the detailed archive data. we did try keeping it all in one database (in different tables) but the file size of the database (interbase) would grow so large that it would drag the system down.

the only real problem we have is accessing detailed data that spans several database, as connecting and disconnecting is slow, and analysis has to be done in code rather than sql

Share Improve this answer

Follow

answered Aug 21, 2008 at 12:54

Re0sless

**10.9k** ● 7 ● 54 ● 68

---

4

If you are using SQL server 2005, this may be a good candidate for using [partitioned tables](partitioned tables).

Share Improve this answer

Follow

answered Aug 21, 2008 at 13:02

wcm

**9,241** ● 7 ● 41 ● 65

---

2

Depending on constraints like budget, etc, this sound like a perfect candidate for a data warehouse application. This would typically introduce a new server for use as a data warehouse. SQL Server 2005 supports a lot of this activity out of the box, further you might be able to utilize additional SQL Server services (e.g. Analysis Services, Reporting Services) to provide additional value to your users. (see http://www.microsoft.com/technet/prodtechnol/sql/2005/dwsqlsy.mspx)

Share Improve this answer

Follow

answered Aug 21, 2008 at 13:47

Peter Meyer

**26.1k** ● 1 ● 35 ● 53

2

@Jason - I don't see how keeping data in plain old text files will allow you to do long term trending analysis easily on the data.

@Jason - I guess my point is that if any sort of ad-hoc analysis (i.e. trending) needs to be done on the data by business people, rolling up or archiving the data to text files really doesn't solve any problems. Of course writing code to consume a text file is easy in many languages, but that problem has been solved. Also, I would argue that today's RDBMS's are all extremely durable when setup and maintained properly. If they weren't why would you run a business on top of one (let alone archive data to it)? I just don't see the point of archiving to a plain text file because of the claim that durability of text files is superior to that of databases.

Share Improve this answer Follow

edited Aug 21, 2008 at 14:22

answered Aug 21, 2008 at 13:39

Peter Meyer

26.1k ● 1 ● 35 ● 53

---

1

Either of those options are excellent, but it really depends on the problem domain. For things like cash balances or statistical data, I think that rolling up records and consolidating them is the best way, you can then move the rolled up records into a parallel archive table, keying them in such a way that you can "unroll" if necessary. This keeps your primary data table clean and quick, but allows you to retain the extra data for auditing or whatever. The key question is, how do you implement the "roll-up" process. Either automatically, via a trigger or server side process, or by user intervention at the application level?

Share Improve this answer Follow

answered Aug 21, 2008 at 12:36

ninesided

**23.2k** ● 16 ● 87 ● 109