Reporting Performance with AR System Applications

By Nicky Madjarov

Contact information: Nicky Madjarov
email: nicky@speedupars.com
phone: 973-202-4278

Version 1.0
March 2009

Check for the latest version of this document and other related articles at http://www.speedupars.com/

1. INTRODUCTION

This article is for those of you who utilize enterprise-wide BMC Remedy ITSM solutions and other Action Request System based custom applications, and are facing poor performance problems. The article concentrates on of-loading the reporting onto archive data sources as one of the ways to increase application performance. The audience could vary from Remedy AR System developers and administrators, dba’s, ITSM service managers.

Archiving is probably the oldest methods to accelerate the search of information in a limited "active" set of data. Every library, government institution, business, even individuals, use archiving in one or other form every day. Let's take for example your taxes. Every year you put together a data set of checks, receipts, incomes, expenses, etc. so you can file your taxes. Once that's done, you put everything nicely in a folder and store it somewhere where it will not take valuable space and you can access it only if needed. The information systems are not much different. There is data that is needed for the day to day operations, e.g. "active" data, and then there is an "archive" data that may be needed once in a while.

While the overall approach for performance improvement of AR System based applications is discussed in the "AR System Performance Improvement Methodology", this article concentrates on of-loading the reporting onto archive data sources, different ways to implement it, improve it, organize it, and as a result, increase the overall application performance.

The amount of data stored in AR System applications database is critical factor for the application performance. The development of performance issues in every AR System application passes through several stages:

1.1. THE NEW APPLICATION

The new, nice and shiny application (custom or out-of-the box), is usually deployed with very limited amounts of data. In most of the cases this is configuration data defining the application behavior. This data, once loaded, will grow and change relatively slowly based on specific business needs. The only exception is the individual user information that can be extensive and can change dynamically based on the update procedures in place.

1.2. THE BEGINNING OF THE ROUTE

Even the most ineffectively designed application will perform well in the beginning of routine use. The reason behind is that the sql server is able to cache most of the frequently used information and provide relatively fast access to it. As the volume of operational data grows, at some point the sql server will reach the a state when it is no longer possible to cache all of the frequently accessed information and the application will enter in its routine use stage. Typical for this stage is that the users can witness fast deteriorating performance in a relatively short period of time. In a matter of few weeks the performance could change from excellent to poor.

1.3. THE ROUTINE USE

All applications are fully deployed and utilized in the organization. Most of the performance problems surface in this stage. It is strictly individual when this will happen, it is affected by environment, available resources, applications in place, quality and quantity of data, etc. It is possible to perform growth analysis for each application and obtain real life results. Using these results one can make projections of the data load in the future.

There is no question that application with less data loaded will perform much better than if there is significant amount of data loaded. A query against table with 1,000 records will run much faster than the same query against the same table with 1,000,000 records.

So, the reduction of the data stored in the application can significantly improve the overall application performance.

Why will one need all this data stored in the application? For reporting (and regulatory) purposes. Each organization has its own reporting requirements and practices. Effective management and capturing trends require vast amount of historical data.

2. IMPLEMENTING EFFECTIVE REPORTING SOLUTIONS

Effective reporting solutions is one way to increase the overall application performance.

The patterns of queering and accessing the application data for reporting purposes are quite different than the patterns of every day operational use. So are the fine tunings in each case, something more, fine tuning aiming operational performance gain could contradict the fine tuning needed for effective reporting. With simple words, certain form may require one set of indexing to perform well from the perspective of every day operations, and entirely different set of indexing for effective reporting. Can we have them both? Not always.

3. ARCHIVING

The process of separating the active application data from the historical data is archiving. There are several steps in designing effective archiving solution:

3.1. SELECT ARCHIVING ARCHITECTURE.

Archiving at the same application server is the simplest and cheapest to implement. There will be copies of the original tables that hold the historical data while the original tables hold the active data only. Once the historical data is moved from the active to the archive form, the entries in the active form could be removed. Separating the historical and active data enables individual fine tuning of each form, including different set of indexes. One of the advantages of this architecture is that these is no need to copy data from all other "shared forms" that could be required for some of the reports.

Archiving on different "reporting" server. In this case there is separate server where the archiving data is stored. There are many possible solutions: from database replication, data warehousing solutions, to Distributed Server Option (DSO) implementations. The latter will be discussed in greater details further in this document.

At the end of this phase you should have clear idea where the archived information is stored.

3.2. DETERMINE THE SCOPE OF ARCHIVING

Not all the form of an application has to be archived. At this step you have to identify a list of forms that will be subject of archiving. This process is affected by the type of architecture selected.

If you are going to implement archiving on the same application server your task is limited to identifying specific forms that are good candidates for archiving. You can use the table growth statistics. Usually the forms with highest growth rate and short life cycle of their entries are the perfect candidates. From ITSM perspective these are incident, change, problem, tasks, work orders, etc. There could be other forms in your applications that contain large amount of information, like asset and personal information. These forms, however, are not good candidates for archiving because most of the information stored there is still active.

Note: You should distinguish between archiving and pruning. In some cases, when the old information is obsolete, you can just prune it from the system (delete the *entries) instead of archiving it. Typical example is the consolidated form(s). In this relation, do not create reports against the consolidated form(s). The information contained there is contained in the original forms as well (incident, problem, change, etc.) and you can extract it from there. If you create reports running against the consolidated form(s) then you'd have to archive it rather than pruning it.*

If you have decided on separate archiving server architecture you have to ensure that all the forms required for your reporting are available on this server. This may require transferring forms in their entirety and keeping them in synch with the originals according to the reporting requirements.

At the end of this phase you should have clear idea what forms will be archived.

3.3. DETERMINE THE ARCHIVING THRESHOLD

The archiving threshold is the amount of time during which the historical data will remain in the active forms before moving it to archive ones. For example, how long a "Closed" incident report will be held in the active incident form before archiving.

You have to understand very well the reporting needs of your organization and ensure that both active and archive contents together will match these needs. To aid this process it is a good idea to sort all reports by period of time covered and make sure that each of them can run either against the active data or against the archived data. This will divide your reports into two groups based on the source they use - active or archive, and will help you restructuring the reporting later in the process.

At the end of this phase you should know what form (and on what server) each report will be running against.

3.4. DETERMINE THE ARCHIVING SCHEDULE

Whether you have decided on archive on the same or on a separate server you have to determine the cycle of transferring the data from the active source to the archived target. Using the considerations about the archiving threshold and the available "transports" specific for your architecture you should determine how frequently the data will be archived. Such schedule could be different for each form.

If it is available as an option you may consider using a delay between transferring the information and removing it. It is allways a good idea to have the transfer process separated from the removal process to avoid triggering workflow on delete.

3.5. PARTIAL DEPLOYMENT

To continue the process you have to implement part of your archiving solution in your development environment. It will not be necessary to follow the same approach in test and production. This is required in order to evaluate the needs of report adjustments in the next step.

3.6. REPORT ADJUSTMENTS

Before jumping into implementation you need to analyze how your reporting will be affected by the archiving solution. You will have to restructure the reporting based on the different data sources that will be used in the future. In some cases this change is simple, in some cases it may require re-designing some of the reports. This, in my opinion, is the most time and resource consuming step in the entire archiving implementation.

At the end of this phase you should have complete, proven set of working reports that utilize the new active and archive data sources.

3.7. PREPARATIONS FOR DEPLOYMENT

This phase is very much affected by the selected architecture and solution specifics, however, here are some common questions to answer while preparing for deployment:

- How much will the reporting change? Will this change affect the current reporting users? Will they need training?
- Solution due-diligence - are there any licenses needed for test and production deployment? Are there any changes required in client and reporting tools settings? If reporting is on separate server, how will be the reporting users authentified and user information maintained?
- Who will maintain and support the new reporting solution (especially if it is on separate server)? Will be any changes in services and the way they are managed? Will that require changes in CMDB?

There is big difference between deploying archiving solution for new application versus adding archiving functions to existing application that contains vast amounts of data. Keep in mind that the latter may require some significant period of time to complete an initial archive feed before deploying routine archiving schedule.

3.8. DEPLOYMENT

Prepare all migration packages, step by step migration procedures, testing plans. Prepare roll-back plan. Prepare user communication plan and training schedules (if applicable). Prepare a change(s), get all necessary resources approved and scheduled. Launch the deployment!

4. VARIATIONS OF NATIVE ACTION REQUEST SYSTEM ARCHIVING SOLUTIONS

As it was mentioned before, there are many, external to AR System, ways to of-load the reporting from the application server: data warehousing, replications, etc. If you have the qualified personnel and some of these is a common practice in your company, you may consider evaluating it.

Since you are already using AR System applications (meaning that you have the knowledge and the personnel already in house) it is certainly an attractive option to use these investments to implement archiving solution. There are several native Action Request System ways to implement archiving. One is based on the form archiving feature, other is based on Distributed Server Option. You can also design custom archiving solution using AR System workflow. The pro's and con's for each solution are discussed further in this document, as well as some helpful hints specific to each of them.

4.1. ARCHIVING USING THE REMEDY AR SYSTEM FORM ARCHIVING FUNCTIONALITY

The AR System provides with straightforward way to implement effective archiving process - the Form Properties>Archive function. Using this function one can implement both archiving and pruning in a matter of minutes. This is the best solution for implementing archiving on the same AR server v.7.0.x and higher.

On the positive side:

- It is out-of-the-box solution and it is cheap to implement - no additional products needed. It is fast to implement, could be implemented form my form, no tight schedules required.
- Creating individual archive forms is highly automated and the fields on source and target are kept in synch automatically as well. Every time a field added or changed in the source, the target is automatically updated.
- Maintaining separate set of indexes for source and target is possible.
- Selecting individual archiving schedule for each form.
- Some flexibility about whether or not to archive diary and attachment fields.
- Ability to Enable/Disable the archiving process individually for each form.
- The AR System automatically blocks any attempt to create or update entries in the archive forms.

On the not that positive side:

- Keeping all field properties in synch updates field permissions as well. All users that access the archiving form should have the same (or similar) permissions as the users that access the original form. The form permissions, however, could be different.
- With the exception of diary and attachment fields, the fields in each table in active and archive data sets are identical. It is not possible to clean-up some fields in the archive if they are not used for reporting. This include all display-only fields.
- Archiving is a scheduled process that could have significant performance impact on the AR server when large amounts of data are to be archived.

This archiving method makes a lot of sense if the archive forms are optimized for reporting. If the reporting is ineffective and individual reports consume a lot of database resources it may affect the overall system performance in a very negative way and it will make the entire archiving implementation pointless.

4.2. ARCHIVING USING CUSTOM WORKFLOW

It is possible and relatively straightforward to create archiving functionality using filters and/or escalations. This solution requires more intellectual labor, however it has its advantages:

- Archiving can be triggered on submit/modify action. This makes possible coping active data into the archiving forms with all the subsequent updates as you go (Real time archiving).
- The archiving forms could be on separate archiving/reporting server.
- Not all the fields have to be present on the archiving form.
- Fields on archiving form could have different permissions that the original ones.
- Maintaining separate set of indexes for active forms and archive forms is possible.

There are some disadvantages:

- There is significant amount of programming needed to implement this method.
- Original forms and archived ones have to be synchronized manually. (adding a field on the original form has to be repeated on the archive form manually).

- If the archive forms are on separate AR server and this server is temporary unavailable the filter processing triggering the archiving action will be broken and the whole user and archiving action will fail.
- There should be a minimum amount of workflow added to the archive forms to prevent modifications or creating of new entries.
- Real time archiving will increase the network traffic and may cause delays in the filter processing phase.

If used within reason this method can bring excellent results. For separate server application high availability environments are recommended, otherwise, temporary unavailability can interrupt the end user routine work with all the unpleasant consequences.

4.3. ARCHIVING WITH REMEDY DISTRIBUTED SERVER OPTION

DSO is separate BMC product that works over Remedy AR System servers. It brings the ability of triggering managed transfers between several AR servers from the workflow. The product itself brings much more value and functionality than the needs of implementing a simple reporting solution, however, for the purposes of this article only the use as transport for reporting solution will be discussed.

The DSO implemented archiving brings together some of the advantages of both methods discussed above:

- Flexible workflow driven transfers makes possible real time archiving implementation.
- Flexible scope of the archived data - archive only the fields needed for reporting.
- Automated transaction handling - the availability of archive destination is not affecting the work on the active server. If archive server is unavailable, the transactions will be queued and processed later.
- Maintaining separate set of indexes for active forms and archive forms is possible.
- Fields on archiving form could have different permissions that the active ones.

The down sides are:

- There is significant amount of programming needed to implement this method.

- There should be a minimum amount of workflow added to the archive forms to prevent modifications or creating of new entries.
- Active and archived forms have to be synchronized manually. (adding a field on the active form has to be repeated on the archive form manually).
- Real time archiving will increase the network traffic.
- Requires additional products - AR System server and DSO for both active and archive servers.
- Requires additional maintenance.

4.4. ADDITIONAL CONSIDERATIONS

There are few helpful considerations that apply to all of the methods discussed in this article:

- Using uniformed naming conventions when creating archiving forms on the same server brings more clarity and makes the work of both users and administrators easier. For example, adding a prefix ARCH_ to all archive forms and using the original name after it will make it easier for everyone to understand that ARCH_HPD:HelpDesk is the archive of HPD:HelpDesk. If the archive is created on separate server, using the same form names is recommended.
- It is always a good idea to have the archiving done in two steps: copy to archive, and then delete after some period of time. This way the workflow on delete will not be triggered at the time of archiving.
- If the applications that are subject of archiving are in routine use and contain large quantities of data it is a good idea to divide the archiving process into two stages:
    - Stage one: Perform gradual initial load of archived data into the archive forms until the routine archiving schedule starts covering the current updates only. (If the archiving threshold is 30 days and the application has been in routine use for a long time there could be millions of entries that will be subject of archiving as soon as the function is enabled. That can significantly affect performance for a long period of time).
    - Stage two: Deploy the routine archiving schedule when it is clear that it will cover only the regular, expected deltas.

5. DECISION MAKING HELP

Selecting the right architecture for implementing archiving/reporting is a complex, tough decision to make. Committing your organization to particular solution will have a long term effects on how business is done. Here are some decision points to consider:

Can you optimize your reports to run effectively when the archive data is stored in separate forms without significant impact on the application performance? In addition to the fine tuning of each report this may include proper scheduling to avoid overlapping of actions that have significant performance impact. If the answer is "Yes" you may consider implementing archiving on the same AR System server.

Is there enough network bandwidth during the regular "peak" hours? If "Yes" implementing of real time archiving becomes an option. If "No" then a sophisticated archiving schedule utilizing the off-hours will be a good option.

6. CONCLUSION

Archiving is a proven way to re-gain application performance. It's effect is immediate. When used in combination with other performance improvement actions it will reveal the true potential of your environment and could prevent extensive spending on additional technologies in attempt to bust the performance of poorly performing applications. Not that I am against the extra technology, but if your application performs 10,000 operations to complete a single function it will still be performing the same number when you move it on that expensive server, it just will be much faster. If you have a way to reduce these 10,000 operations to 1,000, you may not need the more powerful technology after all.

Medium and large size ITSM shops should plan for archiving and off-loading the reporting from the active data sources to archive data sources on same or different server from the very beginning. This way the archiving solution will not be created under the pressure of immediate performance problems.

Designing and implementing archiving solution "when the knife is to the bone" takes more effort and resources. The implementation time is significant.

If you plan your archiving solution at the same time when the new applications are deployed you will save significant amount of both human and financials resources, achieve better results, and avoid pressing problems in the future.