Managing the data life cycle using Azure Data Factory

With the massive amounts of app data that companies accumulate over time, how do you efficiently manage and store all this data—with lower costs? How do you provide secure data access, and improve app performance?

Often, much of this data isn't needed regularly, but legal and company compliance policies require you to keep it for a specified time. Having a well-defined strategy for handling app data that's frequently used, infrequently used, or no longer needed, helps companies address user needs and comply with data retention policies.

To manage the data life cycle across apps, Core Services Engineering (formerly Microsoft IT) built a custom, internal solution that uses Microsoft Azure Stack. This solution—called Data Lifecycle Management (DLM)—uses Azure Data Factory to move data from one stage of the data life cycle to the next. Now that we've built and used the solution in our own team, we help other teams at Microsoft—like Sales, Finance, and Human Resources (HR)—to adopt DLM, so that they can:

- Decrease the time that engineering teams spend creating custom data management solutions, and prevent duplicate efforts. Teams at Microsoft that have adopted DLM have reported saving between 600 and 5,800 hours.
- Improve app performance. When you archive data that you don't need regularly, other than for legal compliance, there's less active data to process. Because of DLM, we've seen processing improvements of up to 80 percent for teams that have started using DLM, which in turn increases productivity.
- Automatically comply with data retention policies and easily retrieve and restore archived data.
- Cut storage costs of archived data with options like Azure Blob storage, File storage, and Table storage. So far, teams that have adopted DLM saved a total of about $500,000 as a one-time cost and $100,000 year over year.
- Reduce data storage needs. From July 2016 to January 2017, we've saved about 4.6 TB.

How Microsoft uses DLM

To deliver this solution, we use the latest Microsoft technologies like Azure Data Factory, Azure Active Directory (Azure AD), and Azure Key Vault. DLM is an Azure-based, platform as a service (PaaS) solution, and Data Factory is at its core. The solution manages data that Microsoft employees generate, and that data can live in the cloud (Azure SQL Database) or on-premises (SQL Server). Even if people generate data on-premises, they don't need to archive it there. With Data Factory, they can move archived data to a central, more cost-effective location.

DLM manages data life cycle activities and stages

Teams at Microsoft use our DLM self-service portal to set up and configure their own data management, retention, and archiving policies for life cycle activities like capturing, analyzing, and maintaining the data for a required period, and eventually purging it from data stores. These activities coincide with stages of the data life cycle:

- **Active.** Data that’s used frequently for reports, queries, and transactions. Also known as hot data.
- **Semi-active.** Data that’s used occasionally, such as for historical comparison. Also known as warm data.
- **Archive.** Data that we need to keep for a defined period, so that we comply with legal and company policies. Also called cold data.
- **Purge.** Data that’s obsolete and that the organization no longer needs to store.
DLM automates common data management tasks

Before we created this solution, we researched typical data management tasks industry-wide. One common task is moving archived data to storage that is cheaper than the tier 1 storage that’s used for active data. Another one is moving data from active to semi-active, and eventually purging it.

And we looked at typical data user roles, like:

- A developer who wants to have better app performance, faster queries, and quicker transactions.
- An app owner who wants to categorize data (active, semi-active, archive, purge), manage it, and comply with data retention policies.
- A service engineer who wants to reduce storage costs with tiered storage—like a storage area network for active data, and lower, less expensive tiers for other data.
- A business or data owner who wants data to be secure.

Based on our research, we decided to build a solution that would address any plug-and-play app, automate these tasks, and meet the needs of typical data users.

After the project grew within our team, we decided to make the core solution available to other business groups in the company. We reached out to learn about their data management requirements and to help them adopt DLM for their apps. But because different groups have different apps and specific requirements—such as the type of storage they want to use—we enhanced our core solution in parallel with onboarding. Some key feature enhancements are related to common tasks like:

- Partitioning (like date-based and integer-based).
- Archiving (with different storage types).
- Purging (like date-based and referential integrity), and compression.

Examples of key scenarios that business groups have completed

Of more than 40 teams we’ve talked to so far, about half are now using DLM in production for PaaS and infrastructure as a service (IaaS) scenarios. There are about 30 to 40 different scenarios that these combined groups use DLM for, but some key examples include:

- For a PaaS app in the HR business group, archive 20 percent of their data to table storage.
- For an IaaS finance forecasting app, move data from active to semi-active using customized rules.
- Schedule moving customer service data to semi-active, and eventually to the purge stage.
- For archived HR data, maintain referential integrity. Maintaining referential integrity ensures that the relationship between two tables remains synchronized during updates and deletions.
- Move sales data from active to archive—storing XML in Blob storage and remaining columns in Table storage.
- Use a custom query to move finance forecasting data to semi-active. Use batch processing for tables without a primary/unique key.
Sample views of the DLM self-service portal

The self-service portal makes it easy to set up and configure data policies. In the portal, you register an app, add a dataset, create and assign a life cycle to it, request approval, and then deploy DLM on the app. The following figures give you a partial glimpse at what DLM looks like in action—they aren’t intended to offer the complete, end-to-end steps. Figures 1 and 2 show what you see when you add a new dataset (database) for a registered app—in this case, for a Customer Support and Services-related app, shown on the left side.

You can add a new dataset for either SQL Database (SQL Database – PaaS or SQL Database – hybrid) and SQL Server.

**Figure 1. Adding a dataset for a registered app**

**Figure 2. Adding a dataset for an app—entering information**
Figure 3 shows what you see when you create a data life cycle. Here, you choose the type of life cycle, define how long you want to keep the data, specify the storage tier, and choose the storage type for archiving data.

Benefits we’ve gained with DLM

What are the results of using DLM so far? For teams that are using it, we’ve seen benefits like:

- **Less storage required.** When data is archived, there’s less active data to manage. After archiving data, teams reduced storage required on average of 30 percent, depending on which policies they implement.

- **Cost savings.** Teams have cut the storage costs of archived data with lower cost options like Azure Blob storage, File storage, and Table storage.

- **Less processing time and better performance.** Teams have reported up to 80 percent performance improvement. Reducing the database size leads to less processing time because there’s less active data to process. This is because active and semi-active data are managed, and there’s less active data to process.

- **Increased productivity.** There’s less customization effort for the engineering team and fewer overall data management processes.

- **Easier data retrieval.** When data was stored on disk, retrieving it was hard. It could take days to find the disk and restore data to the server. With options like Table storage, Blob storage, and File storage, data retrieval is fast.
Figure 4 shows the decrease in storage space needed for teams before DLM versus archiving data with DLM.

Table 1 shows a sampling of initial database savings in GBs and dollars, and cumulative amount saved year over year.

**Table 1. Examples of database savings and dollar savings—initial and year over year (YOT)**

<table>
<thead>
<tr>
<th>Business domain</th>
<th>Initial database savings (GB)</th>
<th>YOY database savings (GB)</th>
<th>Dollar savings during initial run ($)</th>
<th>Year-over-year savings ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sales</td>
<td>300</td>
<td>300</td>
<td>25,000</td>
<td>20,000</td>
</tr>
<tr>
<td>Commerce</td>
<td>2,000</td>
<td>100</td>
<td>50,000</td>
<td>20,000</td>
</tr>
<tr>
<td>HR</td>
<td>200</td>
<td>200</td>
<td>25,000</td>
<td>20,000</td>
</tr>
<tr>
<td>Finance</td>
<td>3,000</td>
<td>800</td>
<td>50,000</td>
<td>4,000</td>
</tr>
</tbody>
</table>
Figure 5 shows the average storage savings in TB for FY 2016 (immediate gains) and monthly (incremental gains) for the first half of FY 2017.

![Storage Savings Chart]

**Figure 5. Less storage required**

When you take the immediate gains, and combine them with ongoing, incremental improvements, the return on investment keeps growing over time.

**Technologies and architecture that we use**

To put the DLM solution together, we use the following Microsoft technologies and the Azure Stack in Table 2:

- Visual Studio 2013
- Azure Data Factory, Azure AD, and Azure Key Vault
- SQL Database, SQL Server 2016
- Windows Server 2012
- ASP.NET MVC4

**Table 2. Azure Stack used in DLM**

<table>
<thead>
<tr>
<th>Azure Stack</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web apps</td>
<td>For the user interface</td>
</tr>
<tr>
<td>SQL Database</td>
<td>For metadata storage</td>
</tr>
<tr>
<td>Data Factory</td>
<td>For metadata sync between the client and DLM, and vice versa; also, for data movement between the client and the target database where the data will be archived</td>
</tr>
<tr>
<td>Azure AD</td>
<td>For authorizing user access to DLM</td>
</tr>
<tr>
<td>Key Vault</td>
<td>For encrypting and storing client credentials for secure connections between DLM and client databases</td>
</tr>
<tr>
<td>Worker roles</td>
<td>For tenants that haven’t yet started using Data Factory, data is moved via worker roles in PaaS.</td>
</tr>
</tbody>
</table>

*Note: At this time, Application Insights is only implemented in worker roles. We get telemetry from Application Insights to help us monitor, troubleshoot, and diagnose app performance problems.*
Our architecture supports on-premises and PaaS apps

To develop our solution and create a self-service portal where teams can set up their policies and rules, we use Visual Studio 2013 and ASP.NET MVC4. Our architecture supports data from on-premises and PaaS apps. Data is archived in low-cost storage, rather than in expensive tier 1 storage where active data is generally stored—as follows:

- File Share and File storage for on-premises
- Blob storage for both on-premises and PaaS
- Table storage and Azure Cosmos DB for PaaS

The architecture has built-in security protection with Azure AD and Key Vault. Teams use Azure AD to authenticate and authorize who can access DLM. And Key Vault offers security with industry-standard hardware security modules. Figure 6 shows a high-level view of what happens behind the scenes from start to finish.

![Figure 6. High-level DLM architecture](image)

In short, the data flow and process that we use to manage app metadata and the data life cycle works like this:

1. The user interface is a self-service portal where people can configure and save their data retention policies and rules. These configurations are stored in metadata in SQL Database.

2. After the policies and rules are set up, the service layer schedules and manages life cycle activities in the app.

3. In the data layer—for teams that have PaaS apps—we use Azure web APIs and a Data Factory pipeline to connect the data source of the active data with the target location of the archived data. Data is pushed through the pipeline to the target destination. And for those who have on-premises databases, SQL jobs run automatically, based on the schedule defined for an individual policy.
4. When teams set up policies and rules and identify what data to archive, metadata is synced and moved from the client machine to the target platform. There are associated APIs that log tenant data for tracking purposes, along with events and alerts to help with debugging and troubleshooting.

**Data Factory moves the data from source to destination**

With Data Factory, you create a managed data pipeline that moves data from on-premises and cloud data stores to Table storage, Blob storage, or other stores. You can also schedule pipelines to run regularly (hourly, daily, weekly), and monitor them to find issues and take action. Figure 7 shows how we use Data Factory to move data through its life cycle stages—from the data sources to the target destinations where we archive the data.

![Figure 7. High-level data flow](image)

We take data from data sources, merge it into a staging table, delete the data from its sources, and copy it from a staging table to its target destination—like table storage or Azure Cosmos DB. Then we truncate the staging table, where an entire (rather than partial) dataset is deleted from a table.

**Why do we use Data Factory?**

We evaluated and compared several technologies for moving data. Given the options we looked at, why did we choose Data Factory? Here are some of its advantages:

- Out-of-the-box support for data in both PaaS and on-premises apps and a data management gateway for connecting to on-premises data activities, for people that have on-premises data.
- Cost savings from less expensive storage options—pay only for storage used.
- Optimal app performance.
- Built-in security with Azure AD and Azure Key Vault.
- Ability to move data on specified dates.

**Challenges we faced**

The benefits of using DLM, with Data Factory at the heart of the solution, are clear. But there have been a few hurdles along the path. Addressing the variety of requirements and PaaS and IaaS scenarios that different teams have—for example, related to storage, partitioning, and archiving—can be hard sometimes. We try to understand how teams work, what technologies they want to use, and ensure that our solution is robust.
Some teams wanted to use a technology that we didn’t offer in our solution yet. But we’re always enhancing DLM and adding components, so that we can help teams as much as we can. Despite the range of requirements, though, Data Factory has been very useful because it supports so many different source and target systems.

**Best practices for data life cycle management**

Along the way to the DLM solution, we learned some valuable lessons:

- Define which scenarios to target. Are you interested in just purging the data? Do you only want to archive it? Or do you want to target all stages in the data life cycle?
- Understand which personas you want to target—for example, the service engineer, app owner, and developer.
- To get an idea of where to invest—and to help define the scenarios and personas that you want to focus on—understand your company’s overall technology strategy and vision.
- Decide whether you want to invest in an internal solution or use products that are available outside.
- Have a well-defined archiving strategy. Identify what data should be active, semi-active, archived, or purged.
- Decide what kind of storage you want to use, and how frequently you want to be able to retrieve data, if needed.
- Decide what the archiving destination will be.

**Looking ahead**

It’s been exciting to work on a solution that makes it easier for us and for other business groups in Microsoft to manage their data life cycle. As part of our roadmap, we plan to:

- Learn more about the data requirements of teams at Microsoft, and keep enhancing DLM accordingly.
- To support the growing use of streaming data, extend the scope of DLM. Right now, DLM supports apps that are on SQL Server and SQL Database. But we plan to bring in Azure Data Lake and big data to support streaming data and apps that aren’t on SQL Server or SQL Database.
- Provide DLM outside Microsoft as an external service or pluggable component.

**Supporting data management needs with DLM**

Having a data management strategy is key. Soaring data volumes and data retention regulations affect app performance and storage costs. When it comes to managing data throughout its life cycle, DLM is a comprehensive, policy-based approach that protects data and supports compliance. Whether app data is on-premises or in the cloud, DLM improves app performance, boosts productivity, archives automatically, and cuts storage costs. Based on policies that are configured in the self-service portal, DLM gives guidance throughout the data life cycle.

**For more information**

**Microsoft IT Showcase**

microsoft.com/ITShowcase

**Azure Data Factory**

**IT Showcase data analytics content**

© 2017 Microsoft Corporation. This document is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS OR IMPLIED, IN THIS SUMMARY. The names of actual companies and products mentioned herein may be the trademarks of their respective owners.