TIPS FOR BULLETPROOF DATA PLATFORM IN AZURE

Deploy secure, cost-effective and maintainable Data Analytics solutions in Azure

VALDAS MAKSIMAVIČIUS

Here is a document to help you speed up the implementation of the Modern Data Platform in Azure. Either you are a single team or an enterprise, you want to ensure your environment is secure, cost-effective and maintainable. My goal is to help you not only deliver a successful minimal viable product, but also ensure the solution is extensible to onboard new teams or use cases without any major refactoring. Let's investigate how to organize resources in Azure, how to link Azure Key Vault, Azure Databricks and Azure Data Factory, and many more.

> *"Your early decisions make the biggest impact on the eventual shape of your system. The earliest decisions you make can be the hardest ones to reverse later. These early decisions about the system boundary and decomposition into subsystems get crystallized into the team structure, funding allocation, program management structure, and even timesheet codes. Team assignments are the first draft of the architecture.*
>
> *It's a terrible irony that these very early decisions are also the least informed. This is when your team is most ignorant of the eventual structure of the software in the beginning, yet that is when some of the most irrevocable decisions must be made."*
>
> *Michael T. Nygard, "Release It!: Design and Deploy Production-Ready Software"*

Table of contents

There is no way this document can withstand the test of time as the services described here are changing fast. Hence, I will do my best to keep it up to date and notify all my subscribers.

Any inputs from your end are highly welcome at valdas@maksimavicius.eu

1. About the Author

Hi there!

I am a software architect specializing in data analytics and cloud computing with ten years of experience. I have been using Azure Cloud components since 2014 and working with data platforms since 2010.

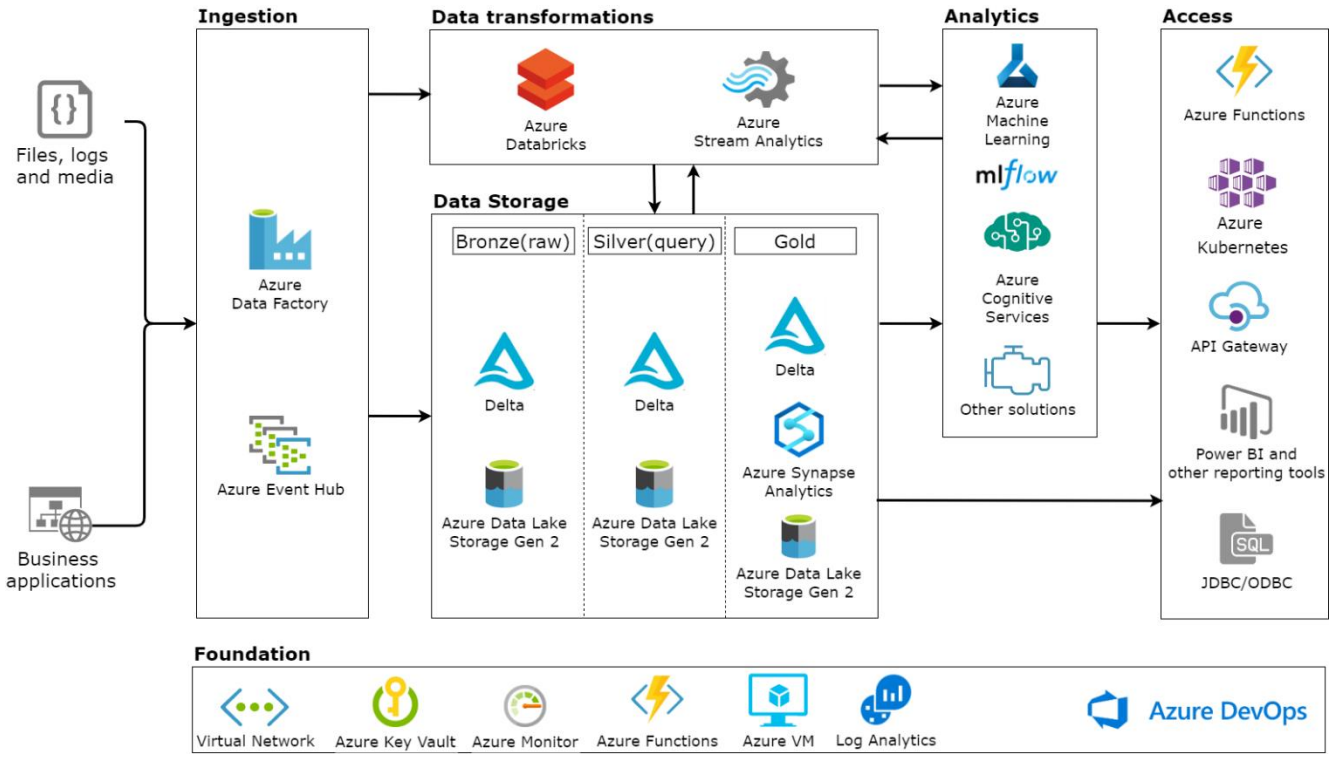
For the last four years, I have been leading Data Engineering teams using the latest Azure Data and AI services. I worked on Data Lake and Data Science platform implementations for various sectors in the Nordics.

I enjoy sharing my lessons learned at conferences and meetups. I am a founder of Vilnius Microsoft Data Platform Meetup and a frequent speaker at IT conferences.

Let's get in touch!
Valdas Maksimavičius
valdas@maksimavicius.eu
https://www.linkedin.com/in/valdasm/

https://www.valdas.blog
https://www.dataplatformschool.com

2. Architecture Overview



This reference architecture is inspired by on an official modern data platform architecture (see reference) and Delta lake (see reference).

Services we are going to cover in this document:

- **Azure Data Factory**
  - It is a hybrid data integration and orchestration service.
  - Read more: https://docs.microsoft.com/en-us/azure/data-factory/
- **Azure Event Hub**
  - It is a Big Data streaming platform and event ingestion service.
  - Read more: https://docs.microsoft.com/en-us/azure/event-hubs/
- **Azure Data Lake Storage Gen 2**
  - It is a scalable object storage for any type of unstructured data, with hierarchical namespace.
  - Read more: https://docs.microsoft.com/en-us/azure/storage/blobs/data-lake-storage-introduction
- **Azure Databricks**
  - It is an Apache Spark-based analytics platform.
  - Read more: https://docs.microsoft.com/en-us/azure/azure-databricks/

- **Delta Lake**
    - It is an open-source storage layer that brings ACID transactions to Apache Spark™ and big data workloads.
    - Read more: https://delta.io/
- **Azure Synapse Analytics**
    - It is a cloud-based multi-node relational database service enabling insights at petabyte scale, with built-in notebook, transformation and visualization capabilities.
    - Read more: https://azure.microsoft.com/en-us/services/synapse-analytics/
- **Azure Machine Learning**
    - It is a managed solution to train, deploy, and manage machine learning models, Auto ML experiments, and pipelines at scale.
    - Read more: https://azure.microsoft.com/en-us/services/machine-learning/#documentation
- **Azure Key Vault**
    - It is a centralized storage of application secrets and their distribution.
    - Read more: https://docs.microsoft.com/en-in/azure/key-vault/

3. Azure Resources Best Practices

1. **Get your team on board by teaching fundamental concepts and terms used in Azure**
    - Most data teams consist of cross-functional roles. (i.e. Data Engineer, Data Scientist, Business Analyst, DevOps, ...). Make sure you all understand the fundamental building blocks of Azure as this will simplify communication, access management, and increase understanding of ongoing work.
    - Read more:
        - https://docs.microsoft.com/en-us/azure/cloud-adoption-framework/ready/considerations/fundamental-concepts
2. **Start with at least two subscriptions: production and non-production workloads.**
    - Microsoft has different pricing offerings for dev/test workloads. You can get discounted rates on Azure services and licensing.
    - Your production and non-production environments will likely have different Azure policies
    - You might have different data policy allowance (e.g. sensitive data only in production subscription, anonymized in non-production subscription)
    - Read more:
        - https://docs.microsoft.com/en-us/azure/cloud-adoption-framework/decision-guides/subscriptions/
3. **Scale your subscriptions only when required**
    - There are certain limitations per subscription, which are often referred to as “quotas”. For example, the number of virtual networks in a subscription is limited. Some quotas can be increased by opening a customer support request. But in some cases, you’ll need multiple subscriptions.
    - When onboarding new users, best practice is to split subscriptions based on functional patterns (functional lines, such as finance, sales), business unit pattern or geographic patterns.
    - Read more:
        - https://docs.microsoft.com/en-us/azure/cloud-adoption-framework/ready/azure-best-practices/scaling-subscriptions

4. **Use Resource Groups as a logical container for Azure resources**
    - Group resources that share the same lifecycle, permissions, and policies. For instance, if Databricks is using Key Vault, Storage, then it makes sense to group these resources in the same resource group.
    - Read more:
        - https://docs.microsoft.com/en-us/azure/cloud-adoption-framework/ready/azure-readiness-guide/organize-resources?tabs=AzureManagmentGroupsAndHierarchy
5. **Define naming conventions for Azure resources**
    - Each resource or service type in Azure enforces a set of different naming rules. Some services require unique names within your subscription, others need unique names within a datacenter, or even globally. To avoid creating an unmaintainable solution, define naming conventions of your resources at the beginning.
    - Read more:
        - https://docs.microsoft.com/en-us/azure/architecture/best-practices/resource-naming
        - https://docs.microsoft.com/en-us/azure/cloud-adoption-framework/ready/azure-best-practices/naming-and-tagging
6. **Apply metadata tags to add additional information that couldn't be included in the resource name**
    - Use tags to include context about the ownership information, budget, environment, application name, etc.
    - Read more:
        - https://docs.microsoft.com/en-us/azure/cloud-adoption-framework/ready/azure-readiness-guide/organize-resources?tabs=ResourceTags
7. **Create Azure AD security groups to give access to Azure resources**
    - Instead of assigning access to Azure resources (or other services such as ADLS Gen2) to a single user, create groups and add AD users or Service Principal users.
    - Read more:
        - https://docs.microsoft.com/en-us/azure/active-directory/fundamentals/active-directory-manage-groups

8. **Script your infrastructure and make it deployable**
    - Creating individual resources in Azure Portal is simple. But the more advanced solution you build, the more complex security and networking requirements get. Automated infrastructure deployment is a must.
    - In Azure, you can choose from ARM templates, PowerShell scripts, CLI. All options have certain limitations, choose one or all of them depending on the services you plan to use.
    - Some decide to go with Terraform or Ansible
    - Read more:
        - https://github.com/Azure/azure-quickstart-templates
9. **Create Azure budgets to track resource consumption or even stop if required**
    - There are tools like a pricing calculator to estimate costs, budget and cost alerts, reviewing costs against your latest invoice. But all these are just soft methods to remind you about your cloud activity rather than hard stop rules.
    - If you want to shut down or delete resources based on your spending, I suggest creating budget action group triggers (e.g. trigger a script to shut down or delete resources when you reach 100% of your budget)
    - Read more:
        - https://docs.microsoft.com/en-us/azure/cost-management/tutorial-acm-create-budgets
10. **Be up to date with the latest Azure releases**
    - Azure releases new upgrades quite frequently. Even a small change for a platform as a service offering can simplify and enhance your productivity (or cause issues 😊)
    - Read more:
        - https://azurecharts.com/

4. Azure Data Factory Best Practices

11. **Parameterize as much as you can**
    - You can parameterize Azure Data Factory objects and pass dynamic values at run time. For example, you can have one linked service per type, one dataset per linked service and one pipeline per ingestion pattern.
    - Read more:
        - https://docs.microsoft.com/en-us/azure/data-factory/control-flow-expression-language-functions
12. **Load metadata from configuration repositories**
    - Metadata should run the whole show. Make sure to derive all sources, destinations, transformations from a file, SQL table or Cosmos DB document. This approach will ultimately reduce development and maintenance efforts.
    - Read more:
        - https://docs.microsoft.com/en-us/azure/data-factory/control-flow-lookup-activity
13. **Consider building metadata and restartability framework**
    - Continuation of the previous point. Create a framework to have a better control of pipelines and control flow
    - Read more:
        - https://github.com/mrpaulandrew/ADF.procfwk
14. **Setup continuous integration & delivery pipeline for ADF**
    - Create environments (development, test, production) and automate the Data Factory pipeline movement.
    - Read more:
        - https://docs.microsoft.com/en-us/azure/data-factory/continuous-integration-deployment#overview
15. **Store credentials in environment-specific Key Vault**
    - Azure Data Factory retrieves the credentials from Azure Key Vault when executing an activity. Azure Key Vault simplifies secrets regeneration, access management, expiration handling and more.
    - Read more:
        - https://docs.microsoft.com/en-us/azure/data-factory/store-credentials-in-key-vault
16. **Define naming conventions for ADF components**
    - Define rules how do you name datasets, linked services, pipelines. For example, how to name multiple Data Factory instances, what prefixes are needed for Data Factory objects, etc.
    - Read more:
        - https://docs.microsoft.com/en-us/azure/data-factory/naming-rules

17. **Look through ADF limitations**
    - Did you know you can have a maximum of 40 activities per pipeline? There are other limitations that you need to get familiar with before you "discover" them in production
    - Read more:
        - https://github.com/MicrosoftDocs/azure-docs/blob/master/includes/azure-data-factory-limits.md
18. **Set up alerts and configure ADF pipelines monitoring**
    - There are built-in diagnostics, alerts, notifications, Gantt chart. Also, you can integrate ADF with Azure Monitor and Log Analytics to discover correlations with other applications.
    - Read more:
        - https://docs.microsoft.com/en-us/azure/data-factory/monitor-using-azure-monitor
19. **Set up an integrated runtime to satisfy advanced security requirements**
    - You need integrated runtime to establish connecting with on-premise or other sources that have restrictions on access, such as IP restriction or other networking rules
    - Read more:
        - https://docs.microsoft.com/en-us/azure/data-factory/create-self-hosted-integration-runtime
20. **Utilize integration with Azure Functions or Logic Apps to create custom functionality**
    - The number of predefined ADF activities is limited. But you can extend it with Azure Functions or Logic Apps. For example, use Azure Functions to collect and send custom logging information to other Azure services (Event Hubs, Cosmos DB).
    - Read more:
        - https://docs.microsoft.com/en-us/azure/data-factory/control-flow-azure-function-activity
        - https://www.mssqltips.com/sqlservertip/5718/azure-data-factory-pipeline-email-notification--part-1/
21. **Azure Synapse data integration will replace Azure Data Factory over time**
    - Continue using Azure Data Factory for now. When the new functionality of data integration within Azure Synapse becomes generally available, Microsoft will provide the capability to import Azure Data Factory pipelines into an Azure Synapse workspace
    - Note that Azure-SSIS Integration Runtime (IR) will not be supported in Synapse

22. **Fetch lineage automatically with Azure Data Catalog Gen 2**
    - Azure Data Catalog Gen 2 has out of the box functionality to collect lineage from connected Data Factory instances
23. **Set up automated testing for Azure Data Factory**
    - Write tests using NUnit and build them into Azure DevOps pipelines
    - Read more:
        - https://richardswinbank.net/adf/set_up_automated_testing_for_azure_data_factory
24. **Build sophisticated data pipelines using multiple Data Factory instances**
    - Consider using cross-factory pipeline orchestration using web activities
    - Read more:
        - https://cloudblogs.microsoft.com/industry-blog/en-gb/technetuk/2020/03/19/enterprise-wide-orchestration-using-multiple-data-factories/

5. Azure Event Hub Best Practices

25. **Decide on partitions as the partition count isn't changeable**
    - Partitions are a data organization mechanism that relates to the downstream parallelism required in consuming applications. The number of partitions in an event hub directly relates to the number of concurrent readers you expect to have.
    - Read more:
        - https://docs.microsoft.com/en-us/azure/event-hubs/event-hubs-faq#best-practices
        - https://docs.microsoft.com/en-us/azure/event-hubs/event-hubs-features#partitions
26. **Consider using hosted Kafka server for CDC scenarios**
    - Event hub namespaces have a hard limit of 10 hubs.
    - Analogy with Kafka: a namespace is a Kafka server and a hub is a Kafka topic. You must split topics per Kafka server based on Azure's 10 hubs hard limit, which is counterintuitive and complicates setup.
    - With CDC scenarios each hub is a topic which is just a SQL table.
    - Read more:
        - https://feedback.azure.com/forums/911458-event-hubs/suggestions/36989824-allow-more-than-10-hubs-per-event-hub-namespace
27. **Split big messages to overcome message maximum size limitation**
    - Maximum size of Event Hubs events is 256 KB (Basic plan) and 1MB (Standard)
    - You might consider as well using other queue services.
    - Read more:
        - https://docs.microsoft.com/en-us/azure/event-grid/compare-messaging-services
28. **Look through other Event Hub limitations**
    - There is a maximum retention period of 7 days? There are other limitations that you need to get familiar with before you "discover" them in production
    - Read more:
        - https://docs.microsoft.com/en-us/azure/event-hubs/event-hubs-quotas

6. Azure Databricks Best Practices

29. **Create separate Databricks workspace for each business team**
    - Assign workspaces based on a related group (business unit, delivery team) of people working together. It helps in streamlining the access within the workspaces (folders, notebooks, etc.)
30. **Put your Databricks workspace in Azure Virtual Network**
    - Databricks creates a virtual network behind the scenes. It is recommended to specify your own virtual network at the start and use it. Calculate how big subnets are required and reserve the required IP address ranges.
    - You can reuse virtual network by multiple workspaces, but there should be unique subnets.
    - Read more:
        - https://docs.azuredatabricks.net/administration-guide/cloud-configurations/azure/vnet-inject.html
31. **Create user groups in Databricks**
    - Databricks integrates with Azure Active Directory to enable single sign-on for users. AAD groups are not present in Databricks. But there is an internal user group functionality to manage users within workspace.
    - Use Databricks groups to limit access to clusters and notebooks.
    - Read more:
        - https://docs.azuredatabricks.net/administration-guide/users-groups/groups.html#manage-groups
32. **Create a dedicated batch user for Databricks**
    - At some point in time, you'll need to automate your notebooks, and you don't want personal tokens in production. Hence, create a batch user in the Azure Active Directory tenant and add it to Databricks.
    - Read more:
        - https://docs.azuredatabricks.net/administration-guide/users-groups/users.html
33. **Use Azure Key Vault backed secrets**
    - Forbid users from using plain secret values in their notebooks. Instead, connect Azure Key Vault and fetch secrets from there. After linking the scope, give users/user groups access rights to the appropriate scopes (Azure Key Vault).
    - Read more:
        - https://docs.azuredatabricks.net/security/secrets/secret-scopes.html

34. **Use other storages instead of DBFS**
    - You can't restrict access to DBFS folders (unless you enable password pass-through functionality, then DBFS is not reachable at all)
    - The lifecycle of default DBFS is tied to the Workspace. Deleting the workspace will also delete the default DBFS and permanently remove its contents.
35. **Keep data as close as possible to reduce latency**
    - Even though you can find connectors to any storage (ADLS, SQL, NoSQL) and run analysis on top of it, try to select the data storage location as close to you as possible to reduce latency.
    - Keep in mind push-down calculations. Spark can handle most of the transformations way faster. For large datasets it is worth loading the data from RDBMS to data lake before performing the data manipulation.
36. **Use direct access to Azure Data Lake Gen 2 instead of mounts**
    - Once you mount storage, all users having access to the workspace will be able to use the mounted storage with identical access. If you set up direct access, there are ways to control and monitor who can access what. For example, by using Data Lake Storage ACLs.
37. **Enable Azure Data Lake Storage Credential Passthrough**
    - Credential passthrough allows users to authenticate to Azure Data Lake Storage from Azure Databricks clusters using the same Azure Active Directory identity that they use to log into Azure Databricks.
    - There are certain limitations of password passthrough (e.g. unavailable Python I/O). You can have a cluster dedicated to Spark workloads with enabled password passthrough. Then, another cluster for non-Spark workloads (if required) with a dedicated Service Principal user to access the data.
    - Read more:
        - https://docs.azuredatabricks.net/administration-guide/access-control/credential-passthrough.html
38. **Use ephemeral job clusters**
    - Let each job create a separate cluster for its execution, with a short life tied to the job lifecycle.
    - Such clusters are cheaper and easier to manage as it’s running in an isolated environment.
    - Read more:
        - https://docs.azuredatabricks.net/jobs.html

39. **Use instance pools to initiate clusters faster**
    - To reduce cluster's start time, you can attach a cluster to a predefined pool of idle instances. When attached to a pool, a cluster allocates its driver and worker nodes from the pool.
    - Azure Databricks does not charge DBUs while instances are idle in the pool, but Azure VM charges still apply.
    - Read more:
        - https://docs.azuredatabricks.net/clusters/instance-pools/index.html
40. **Avoid cluster init scripts**
    - Databricks runtimes have many pre-installed libraries and components. Make sure to check what's available before you decide to customize the clusters.
    - Treat Init scripts with caution, because they can lead to cluster launch failures. If you really need them, please use the Cluster Scoped execution as it's easier to find failures in the logs
    - Read more:
        - https://docs.azuredatabricks.net/clusters/init-scripts.html
41. **Install Python libraries and create an environment scoped to a notebook session**
    - You can install Python libraries and create an environment scoped to a notebook session. Hence you can share a cluster with others without interference. The libraries are available both on the driver and on the executors, so you can reference them in UDFs.
    - Read more:
        - https://docs.microsoft.com/en-us/azure/databricks/dev-tools/databricks-utils#--library-utilities
42. **Enable diagnostics for Databricks workspace**
    - You can turn on Azure Diagnostics to send all user activities and logs into Log Analytics for detailed analysis.
    - Read more:
        - https://docs.azuredatabricks.net/administration-guide/account-settings/azure-diagnostic-logs.html
43. **Monitor and trace performance and resource usage**
    - To get the most out of the Databricks cluster, make sure to optimize Spark workloads, including the performance and resource usage on the host and JVM, as well as Spark metrics and application-level logging.
    - You can use built-in Ganglia metrics or send detailed logs to Azure Monitor.
    - Read more:
        - https://docs.microsoft.com/en-us/azure/architecture/databricks-monitoring/application-logs
        - https://docs.azuredatabricks.net/release-notes/product/2019/july.html#ganglia-metrics

44. **Use Databricks CLI for administrative tasks**
    - Databricks CLI is required for CI/CD pipelines, granting access to secret scopes, interacting with workspace, and more.
    - Read more:
        - https://docs.databricks.com/dev-tools/databricks-cli.html
45. **Put your artifacts in source control**
    - Databricks has a native notebooks source control integration. Another approach is to use Databricks CLI to download your artifacts (notebooks, clusters, configuration) and add to source control from your PC.
    - Read more:
        - https://docs.azuredatabricks.net/notebooks/github-version-control.html
46. **Setup continuous integration & delivery pipeline for Databricks**
    - There are various approaches on how to split environments (development, test, production) in Databricks. Either you can create workspace folders (dev, test, prod) with different access levels or create dedicated workspaces. For smaller deployments, I like the former option more.
    - Read more:
        - https://databricks.com/blog/2017/10/30/continuous-integration-continuous-delivery-databricks.html
        - https://databricks.com/session/devops-for-applications-in-azure-databricks-creating-continuous-integration-pipelines-on-azure-using-azure-databricks-and-azure-devops
47. **Separate exploration, machine learning and production clusters**
    - Create separate clusters for data transformation/exploration workloads, machine learning, and production pipelines.
    - Read more:
        - https://docs.databricks.com/administration-guide/capacity-planning/cmbp.html
48. **Parametrize your notebooks with widgets**
    - There is a possibility to add parameters to your notebooks (for example during notebook execution from Azure Data Factory inject pipeline date value to your notebook).
    - Read more:
        - https://docs.databricks.com/notebooks/widgets.html

49. **Be up to date with the latest Databricks runtimes**
    - Databricks releases new versions quite frequently. New runtimes include new functionality and new/upgraded libraries.
    - Read more:
        - https://docs.databricks.com/release-notes/runtime/supported.html#release-notes
50. **Decide when to use Azure Data Factory and when Databricks**
    - It depends on several different factors such as performance, cost, preference, security, feature capability and more. Simple recommendation is to use both.
    - Read more:
        - https://www.mssqltips.com/sqlservertip/6438/azure-data-factory-vs-ssis-vs-azure-databricks/
51. **Check other in-depth Databricks recommendations**
    - Read more:
        - https://github.com/Azure/AzureDatabricksBestPractices/blob/master/toc.md
52. **Use Azure AD to authenticate instead of personal tokens**
    - In the past, the Azure Databricks API has required a Personal Access Token (PAT), which must be manually generated in the UI. This complicates DevOps scenarios. A new feature in preview allows using Azure AD to authenticate with the API
    - Read more:
        - https://cloudarchitected.com/2020/01/using-azure-ad-with-the-azure-databricks-api/
        - https://docs.microsoft.com/en-us/azure/databricks/dev-tools/api/latest/aad/

7. Databricks Delta Best Practices

53. **Use Delta format instead of Parquet when Databricks is the main data engine**
    - Delta is a file format build on top of Parquet with support to ACID transactions, performance improvements and delta tables support.
    - Read more:
        - https://delta.io/
54. **Avoid Delta if you plan to access it outside Databricks Runtime**
    - There are two cases to consider: external writes and external reads.
    - External writes: Delta Lake maintains additional metadata in the form of a transaction log to enable ACID transactions and snapshot isolation for readers. In order to ensure the transaction log is updated correctly and the proper validations are performed, writes must go through Databricks Runtime.
    - External reads: Delta tables store data encoded in an open format (Parquet), allowing other tools that understand this format to read the data. For information on how to read Delta tables, see Integrations.
55. **Choose the right partition column**
    - You can partition a Delta table by a column. The most commonly used partition column is date
    - Read more:
        - https://medium.com/@aravinthR/partitioned-delta-lake-part-3-5cc52b64ebda
56. **Compact underlying files**
    - Delta table will over time accumulate many files, especially if you add data in small batches. You can compact a table by repartitioning it to smaller number of files
    - Read more:
        - https://docs.delta.io/latest/best-practices.html#compact-files
57. **Provide data location hints**
    - If you expect a column to be commonly used in query predicates and if that column has high cardinality (that is, many distinct values), then use Z-ORDER BY. Delta Lake automatically lays out the data in the files based on the column values and use the layout information to skip irrelevant data while querying.
    - Read more:
        - https://docs.databricks.com/delta/optimizations/file-mgmt.html#delta-zorder

58. **Use same functions for both real-time and batch pipelines**
    - My favorite point of Delta lake – code reusability!
    - One you apply a function that cleans the data and aggregates it to the real-time stream, you can reuse the same function for the batch job.
    - Read more:
        - https://www.element61.be/en/resource/best-practice-modern-data-platform-azure-databricks-and-delta
59. **Use Delta to simplify GDPR and CCPA compliance**
    - Enable quick and easy search, modification, and cleanup of your data using standard SQL DML statements like DELETE, UPDATE, and MERGE INTO.
    - Read more:
        - https://docs.microsoft.com/en-us/azure/databricks/security/privacy/gdpr-delta

8. Azure Data Lake Store Gen 2 Best Practices

60. **Define zones (directory layout) and characteristics**
    - What data will you keep in ADLS Gen 2? How the data will differ (based on access rights, content, purpose, etc.). Define the characteristics and folder structure that you plan to use.
    - Read more
        - https://www.sqlchick.com/entries/2017/12/30/zones-in-a-data-lake
        - https://www.sqlchick.com/entries/2016/7/31/data-lake-use-cases-and-planning
61. **Plan your security methodology before you load your first file**
    - Plan your security methodology before you load your first file. Map it out what user groups you need, how do you give access to new teams, how do you separate batch users from end-users, etc.
    - Read more:
        - https://adatis.co.uk/Implementing-Enterprise-Security-in-Azure-Databricks-Part-2/
62. **Define access mechanisms**
    - Depending what is your scenario, you might want to read ADLS Gen 2 data from PowerBI, Databricks, HDInsight, SQL Datawarehouse and more. Unfortunately, all of these have different capabilities and limitations.
    - Read more:
        - https://www.jamesserra.com/archive/2019/09/ways-to-access-data-in-adls-gen2/
63. **Apply access rights (ACLs) to ADLS Gen 2 objects**
    - Always use Azure AD security groups as the assigned principal in ACLs. Resist the opportunity to directly assign individual users or service principals. Using this structure will allow you to add and remove users or service principals without the need to reapply ACLs to an entire directory structure.
    - There is limited support for end-to-end ACLs setup from Storage Explorer, so you might end up giving permissions via APIs
    - Read more:
        - http://sql.pawlikowski.pro/2019/03/10/connecting-to-azure-data-lake-storage-gen2-from-powershell-using-rest-api-a-step-by-step-guide/

64. **Apply data retention to your data**
    - You can always implement custom logic with Databricks, ADF, or other tools. Else, you can use built-in storage lifecycle management to get rid of old files.
    - Read more:
        - https://docs.microsoft.com/en-us/azure/storage/blobs/storage-lifecycle-management-concepts?tabs=azure-portal
65. **Separate long-term storage from explorative sandboxes and staging storages**
    - Assuming your ADLS Gen 2 represents your long-term storage, consider creating separate instances for user exploration sandboxes or staging. Such storages would have different retention policies, integrated CI/CD process, and different responsible persons.
66. **Choose between Premium, Hot, Cool & Archive plans**
    - Optimize pricing and performance by choosing the right plan for your needs. You need to take into consideration data size, read & write operations, data transfer
    - Read more:
        - https://azure.microsoft.com/en-us/pricing/details/storage/data-lake/
67. **Enable the Data Lake Storage Gen2 firewall**
    - By default, the firewall is disabled which can make your data vulnerable. Limit the access by specifying virtual networks and IP addresses of your data readers.
    - Read more:
        - https://docs.microsoft.com/en-us/azure/storage/common/storage-network-security
68. **Enable diagnostics and audit**
    - Collect usage and other metrics in blob storage. There are still some limitations and you might need to use PowerShell to enable it.
    - Read more:
        - https://docs.microsoft.com/en-us/azure/storage/common/storage-analytics-logging

9. Azure Synapse Best Practices

69. **Automate pause and scale to reduce costs**
    - Make sure to pause your instance if no one is using it. You might create an API call, Azure Automation workbook to run at scheduled intervals or on-demand.
    - Read more:
        - https://docs.microsoft.com/en-us/azure/sql-data-warehouse/sql-data-warehouse-manage-compute-overview
70. **Set proper distribution to achieve high performance**
    - A hash distributed table can deliver the highest query performance for joins and aggregations on large table columns.
    - A round-robin distributed table distributes data evenly across the table, but without any further optimization. A round-robin table is fast for storing data, but query performance can often be improved with hash distributed tables
    - A replicated table caches a full copy of the table on each compute node. Replicated tables are best utilized and only recommended for small tables (lookups)
71. **Monitor and optimize your queries**
    - SQL Datawarehouse uses a different distribution, indexing and statistics mechanisms than classic SQL databases. Hence, monitor performance closely to ensure you are using resources that you are paying for.
    - Read more:
        - https://docs.microsoft.com/en-us/azure/sql-data-warehouse/sql-data-warehouse-manage-monitor
72. **Speed up data loading with PolyBase**
    - Use PolyBase to load data faster into SQL Datawarehouse via an intermediate blob storage.
73. **Avoid using external tables for queries**
    - While Polybase with external tables can be the fastest way to load data, it is slow for queries. During the query, data gets loaded to tempdb before reading the data. Hence, if you have multiple queries, consider using a local table with preloaded data.

74. **Use smaller resource class to increase concurrency and larger resource class for query performance**
    - Resource classes are resource limits in Azure SQL Data Warehouse that govern compute resources and concurrency for query execution. It helps to manage workloads by setting limits for users on the number of queries that run concurrently, and on the compute-resources assigned to each query.
    - Read more:
        - https://docs.microsoft.com/en-us/azure/sql-data-warehouse/resource-classes-for-workload-management
75. **Separate static and dynamic resource classes**
    - Assign static resource classes for user/group based on their usage. Static resources classes, which are well suited for increased concurrency on a data set size that is fixed
    - Dynamic resource classes, which are well suited for data sets that are growing and need increased performance as the service level is scaled up. Do not assign dynamic resource class for known usage
    - Larger resource classes take precedence over smaller resource classes, if user assigned multiple resource classes.
76. **Enable firewall on a server or database level**
    - Limit the access by specifying virtual networks and IP addresses of your consumers. For example, Azure Databricks clusters and Azure Data Factory integrated runtime IP ranges.
    - Read more:
        - https://docs.microsoft.com/en-us/azure/sql-database/sql-database-firewall-configure
77. **Prefer Azure Active Directory users over SQL users**
    - Azure Active Directory users and groups should be used for all users. However, there are some limitations, where only SQL users will work (access from Databricks) but keep it to a minimum.
    - Read more:
        - https://docs.microsoft.com/en-us/azure/sql-data-warehouse/sql-data-warehouse-authentication

78. Enable results cache in the user database for repetitive use

- This allows subsequent query executions to get results directly from the persisted cache so recomputation is not needed. Result set caching improves query performance and reduces compute resource usage.
- Once enabled, results are cached for all queries until the cache is full, except, for example: Queries using user defined functions, tables with row level security or column level security enabled, and there are more exceptions.
- Read more:
  - https://docs.microsoft.com/en-us/azure/synapse-analytics/sql-data-warehouse/performance-tuning-result-set-caching

10. Azure Key Vault Best Practices

79. **Create Azure Key Vault instances per application per environment**
    - It is required to prevent from sharing secrets across environments and it reduces the threat in case of a breach.
    - You assign permissions on the Key Vault level, not on the keys/secrets level. Hence users get access to all elements inside.
80. **Enable the Azure Key Vault firewall and set up access policies**
    - Create access policies for every vault, use the least privilege access principal to grant access and turn on the firewall.
    - To fetch secrets from Azure Data Factory or Azure Databricks, you'll need to grant access to these applications in access policies.
    - Read more:
        - https://docs.microsoft.com/en-us/azure/key-vault/key-vault-secure-your-key-vault
81. **Enable logging**
    - Monitor how and when your key vaults are accessed, and by whom.
    - Read more:
        - https://docs.microsoft.com/en-us/azure/key-vault/key-vault-logging
82. **Develop a key expiry notification & auto-rotation system**
    - Minimize manual intervention needed for secret changes.
    - Read more:
        - https://medium.com/@cprosenjit/azure-databricks-user-token-management-we-can-end-up-developing-a-key-expiry-notification-f65828b66401

11. Bonus

The official Azure documentation often takes a siloed approach and misses out more advanced Big Data / Machine Learning end-to-end scenarios. I collected a list of awesome blog posts on Azure Databricks, Azure Data Factory, Azure Data Lake and other related topics. Understand the bigger picture, build configurable end-to-end pipelines, automate deployment, set up accesses and permissions.

Building Modern Data Platform in Azure - Useful Resources (last updated in July 2020)
https://www.valdas.blog/2019/07/13/azure-useful-links/