White paper

Azure Synapse Analytics
Proof of Concept Playbook

Executive summary

Whether it is an enterprise data warehouse migration, a big data re-platforming, or a green field implementation; each project traditionally starts with a proof of concept.

This Proof of Concept playbook provides a high-level methodology for planning, preparing, and running an effective proof of concept project. An effective proof of concept validates the fact that certain concepts have the potential for real-world production application. The overall objective of a proof of concept is to validate potential solutions to technical problems, such as how systems can be integrated or how results can be achieved through a specific configuration.

This playbook will help you to evaluate the use of Azure Synapse Analytics for the migration of an existing workload. It has been designed with the following readers in mind:

- Technical experts planning their own in-house Azure Synapse proof of concept project
- Business owners who will be part of the execution or evaluation of an Azure Synapse proof of concept project
- Anyone looking to learn more about data warehousing proof of concept projects

The playbook will deliver the following:

- Guidance on what makes an effective proof of concept
- Guidance on how to make valid comparisons between systems
- Guidance on the technical aspects of running an Azure Synapse proof of concept
- A road map to relevant technical content from Azure Synapse
- Guidance on how to evaluate proof of concept results to back business decisions
- Guidance on how to find additional help

Let’s get started.

Data Warehousing with Dedicated SQL Pool

Preparing for your proof of concept

Before going through the process of creating Goals for your Azure Synapse Analytics POC it is worth taking some time to understand the service's capabilities and how they might apply to your POC. To make the most of your POC execution time, read about it in the Service Overview. Additionally, it is worth reading through the Azure Synapse SQL Pools Architectural Overview to familiarize yourself with how SQL pools separate compute and storage to provide industry-leading performance.

Finally, take a look at our videos walking through use cases and new announcements regarding Azure Synapse.

Identify sponsors and potential blockers

Now that you are familiar with Azure Synapse. It is time to make sure that your proof of concept has the necessary backing and will not hit any roadblocks.

Now is the time to:

- Identify any restrictions or guidelines that your organization has about moving data to the cloud.
- Identify executive and business sponsorship for a cloud-based data warehouse project.
- Verify that your workload is appropriate for Azure Synapse. Read more here.

Setting your timeline

A proof of concept is a scoped, time-bounded exercise with specific, measurable goals and metrics of success. Ideally, it should have some basis in business reality so that the results are meaningful.

In our experience, proof of concepts have the best outcome when they are timeboxed to two weeks. This provides enough time for work to be completed without the burden of too many use cases and complex test matrices.

Working within this timeline, you can follow this rough agenda:

- Loading: Three days or less
- Querying: Five days or less
- Value added tests: Two days or less

This agenda is intended to be more of a guideline than a rule.

Here are some tips:

- Make realistic estimates of the time that will be required to complete the tasks in your plan.
- The time to complete your proof of concept will be influenced by the size of its dataset, the number of database objects (for example, tables, views, and stored procedures), the complexity of your database objects, and the number of interfaces you are testing.
- If you find that your proof of concept is estimated to run longer than four weeks, consider reducing its scope to focus on the highest priority goals.
- Get buy-ins from all the lead resources and sponsors for the timeline before continuing.

Now that you have determined that there are no immediate blockers and you have set your timeline, let’s move on to scoping an architecture.

Creating a high-level proof of concept scoped architecture

Your high-level future architecture will likely contain many data sources, numerous data consumers, Big Data components, and possibly machine learning and AI data consumers. In order to keep the goal of your proof of concept achievable within your set timeframe, decide which of these components will be part of the proof of concept and which ones will be excluded from it.

Additionally, if you are already using Azure, you need to identify the following:

- Any resources you already have in Azure (for example, Azure Active Directory, ExpressRoute) that can be used during the proof of concept
- What Azure region(s) your organization prefers
- A subscription for non-production, proof of concept work
- The throughput of your network connection to Azure. Also, check with other business users that your proof of concept can consume some of that throughput without having an adverse effect on production solutions.

Migration considerations

If you are migrating from a legacy data warehouse system to Azure Synapse, here are some questions to consider:

- Are you migrating and want to make as few changes to existing Extract, Transform, Load process (ETL) and data warehouse consumption as possible?
- Are you migrating but want to do some extensive improvements along the way?
- Are you building an entirely new data warehouse environment (green field)?

Next, you need to consider your pain points.

Identifying your current pain points

Your proof of concept should contain use cases to prove potential solutions to address your current pain points. Here are some questions to consider:

- What gaps in our current implementation do we expect Azure Synapse to fill?
- What new business needs are you being asked to support?
- What service level agreements (SLAs) are you required to meet?
- What will be the workloads (for example, ETL, batch queries, analytics, reporting queries, or interactive queries)?

The next step is to set your goals.

Setting the goals

Identify why you are doing a proof of concept and write out clear goals. It is important to know before you start the service what outputs you want from your proof of concept and what you will do with them.

Keep in mind that a proof of concept should be a short and focused effort to quickly prove or test a limited set of concepts. If you have a long list of items to prove, you may want more than one proof of concept with gates between them where you determine whether you need the next one.

Here are some example proof of concept goals:

- We need to know that the query performance for our big complex reporting queries will meet our new SLAs.
- We need to know the query performance for our interactive users.
- We need to know whether our existing ETL processes are a good fit and where improvements need to be made.
- We need to know whether we can shorten our ETL runtimes and by how much.
- We need to know that the enterprise data warehouse (EDW) has sufficient security capabilities to secure our data.

The next step is to plan your testing.

Creating a test plan

Using your goals, identify specific tests to run in order to support those goals and provide the outputs you identified. It is important to make sure that you have at least one test to support each goal and the expected output. Identify specific queries, reports, ETL, and other processes that will be run so that a very specific dataset can be identified.

Refine your tests by adding multiple testing scenarios to clarify any table structure questions that have arisen.

Effective proof of concept execution is usually defined by good planning. Make sure all stakeholders agree to a written test plan that ties each proof of concept goal to a set of clearly stated test cases and measurements of success.

Most test plans revolve around performance and the expected user experience. What follows is an example of a test plan. It is important to customize your test plan to meet your business requirements. Clearly defining what you are testing will pay dividends later in this process.

Goal	**Test**	**Expected Outcomes**
• We need to know that the query performance for our big complex reporting queries will meet our new SLAs	• Sequential test of "complex" queries • Concurrency test of complex queries against stated SLAs	• Queries A, B, and C finished in 10, 13, and 21 seconds, respectively • With 10 concurrent users, queries A, B, and C finished in 11, 15, and 23 seconds, on average
• We need to know the query performance for our interactive users	• Concurrency test of selected queries at an expected concurrency level of 50 users. • Run the preceding with result-set caching	• At 50 concurrent users, average execution time is expected to be under 10 seconds, and without result-set caching • At 50 concurrent users, average execution time is expected to be under five seconds with result-set caching

Goal	**Test**	**Expected Outcomes**
• We need to know whether our existing ETL processes can run within the SLA	• Run one or two ETL processes to mimic production loads	• Loading incrementally onto a core fact table must complete in less than 20 minutes (including staging and data cleansing) • Dimensional processing needs to take less than five minutes
• We need to know that the EDW has sufficient security capabilities to secure our data	• Review and enable network security (VNET and private endpoints), data security (row-level security, Dynamic Data Masking)	• Prove that data never leaves our tenant. • Ensure that PII is easily secured.

The next step is to identify your dataset.

Identify and validate the proof of concept dataset

From the tests scoped you can identify the data that will need in Azure Synapse to execute those tests. Now take some time to review this dataset by considering the following:

- You need to verify that the dataset will adequately represent your future processing on Azure Synapse in both content, complexity, and scale.
- Do not use a dataset that is too small (smaller than 1 TB in size) as you will not see representative performance.
- Do not use a dataset that is too large, as your proof of concept is not intended to complete your full data migration.
- Identify the distribution pattern and indexing option for each table.
- If there are any questions regarding distribution. Indexing, or partitioning, add tests to your proof of concept to answer your questions.

- Remember that you may want to test more than one distribution option or indexing option for certain tables.
- Make sure that you have checked with the business owners for any blockers for moving this data to the cloud.
- Identify any security or privacy concerns.

Next, you need to put together a team.

Assembling your team

Specifically identify the team members needed and the commitment that will be required to support your proof of concept. The team members should include:

- A project manager to run the proof of concept project.
- A business representative to oversee requirements and results.
- An application data expert to source the data for the proof of concept.
- An Azure Synapse specialist.
- An expert advisor to optimize the proof of concept tests.
- Any resources that will be required for specific components of your proof of concept project but are not necessarily required for its entire duration. These resources could include network administrators, Azure administrators, and Active Directory administrators.

Since you are evaluating a new platform, we recommend engaging an expert advisor to assist with your proof of concept. Microsoft's partner community has global availability of expert consultants who are able to demonstrate the features and performance of Azure Synapse Analytics.

Now that you are fully prepared, it is time to put your proof of concept into practice.

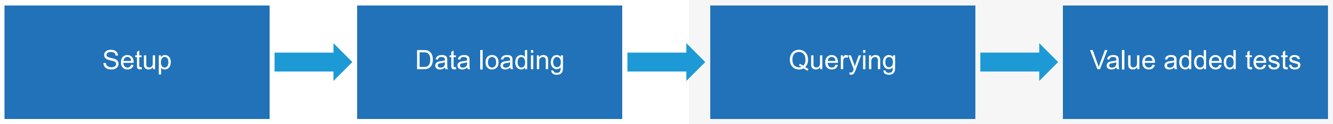
Putting it into practice

It is important to keep the following in mind:

- Implement your proof of concept project with the discipline and rigor of any production project.
- Run it according to plan.
- Have a change request process in place to prevent your proof of concept from growing and changing out of control.

Setting up

Before tests can start, you need to setup the test environment and load the data:



Setting up a proof of concept on Azure Synapse is as easy as clicking a few buttons.

Perform the following steps:

- From the Azure portal, follow this tutorial to create an Azure Synapse SQL pool.
- Ensure that the SQL pool's firewalls are open to your client machine.
- Download and install SQL Server Management Studio (SSMS).

When you set up your SQL pool, you can set the Data Warehouse Units (DWUs). DWUs range from 100 to 30,000 and define the performance characteristics of your SQL pool. This value can be changed at any point by scaling your SQL pool.

We suggest developing code and unit testing at DW500c or below and running load and performance tests at DW1000c or above. At any point, you can pause your SQL pool, which will save on costs.

Data loading

Now that your SQL pool has been created, it is time to load some data.

Do the following:

- If you have not already done so, load some data into an Azure Storage blob. We advise using a General-purpose V2 storage blob with locally redundant storage for a proof of concept. There are several tools for migrating your data to an Azure Storage blob. The easiest way is to use Azure Storage Explorer and copy files into the storage container.
- Now that you have data in your Azure storage container, you can load it into the SQL pool. Azure Synapse supports two T-SQL loading methods: PolyBase and COPY statement.

To load data, you need to connect to your Azure Synapse SQL pool via a tool such as SSMS. Once connected to your database, you can use PolyBase or the COPY INTO statement.

When loading data for the first time into Azure Synapse SQL pools, a common question is what distribution and index to choose. While Azure Synapse SQL pools support a variety of both, it is a best practice to use the defaults of round-robin distribution and clustered columnstore index. From here, you can fine-tune your environment, which we will discuss in a later section.

The following is a COPY INTO example:

```
-- Note when specifying the column list that input field numbers start from 1
COPY INTO test_1 (Col_one default ‘myStringDefault’ 1, Col_two default 1 3)
FROM ‘https://myaccount.blob.core.windows.net/myblobcontainer/folder1/’
WITH (
   FILE_TYPE = ‘CSV’,
   CREDENTIAL = (IDENTITY= ‘Storage Account Key’, SECRET=’<Your_Account_Key>’),
   FIELDQUOTE = ‘"’,
   FIELDTERMINATOR = ‘,’,
   ROWTERMINATOR = ‘0x0A’,
   ENCODING = ‘UTF8’,
   FIRSTROW = 2
)
```

Let’s now move on to querying.

Querying

Deriving analytics from your data is why you are looking at a data warehouse in the first place, so let’s see how to get the most out of your database. Most proof of concepts start by running a small number of representative queries against the data warehouse, first in sequence and then concurrently. This should be defined in your test plan.

Sequential query test

Running queries sequentially is extremely easy with SSMS. It is important to run these tests using a user with a sufficient large resource class. For simple queries, we recommend Static 20. For more complex queries, we suggest Static 40.

The following SQL query uses a query label to keep track of the first query in Dynamic Management Views (DMV). Then it uses sys.dm_pdw_exec_requests to determine query execution duration:

Helpful hint: Using a query label is a great way to keep track of your queries.

```
/* Use the OPTION(LABEL = '') Syntax to add a query label to track the query in DMVs */
SELECT TOP (1000) * FROM [dbo].[Date] OPTION (LABEL = 'Test1')
/* Use sys.dm_pdw_exec_requests to determine query execution duration (ms) */
Select Total_elapsed_time as [Elapsed_Time_ms],
    [label]
FROM sys.dm_pdw_exec_requests
WHERE [label] = 'Test1'
```

Concurrent query test

After baselining single query performance, many customers evaluate running multiple read queries simultaneously. This is intended to mimic a business intelligence scenario running against the SQL pool. The easiest way to run this test is to download a stress testing tool. The most popular tool for this is Apache JMeter.

An expected outcome of this test is a minimum, maximum, and average execution time at a given concurrency level. For example, suppose that you want to mimic a business intelligence workload that generates 100 concurrent queries. You would setup JMeter to run those 100 concurrent queries in a loop and look at the steady state execution. This could be done with result-set caching on and off to show the benefits of that feature.

Document your results that clearly show the results.

Concurrency	# Queries Run	DWU	Min Duration (s)	Max Duration (s)	Median Duration (s)
100	1,000	5,000	3	10	5
50	5,000	5,000	3	6	4

Mixed workload test

Mixed workload testing is simply an addition to the concurrent query test. By adding a data update statement, like something from the loading process, into the workload mix, the workload better simulates an actual production workload.

Tuning your query tests

Depending on the query workload running on Azure Synapse, you may need to fine-tune your data warehouse's distributions and indexes. For guidance, please refer to our best practices guide.

The most common mistakes seen during setup are as follows:

- Large queries run with too small a resource class.
- DWUs are too small for the workload.
- Large tables require hash distribution.

Common tuning changes that can be applied are as follows:

- Materialized views are used to accelerate common aggregations.
- Replicate small dimension tables.
- Hash distribute large fact tables that are joined or aggregated.

Let's now look at value added tests.

Value added tests

After core price performance testing is complete, it is a good time to test specific features to verify that they satisfy your intended use case. More often than not, these features are follows:

- Column-level security
- Row-level security
- Dynamic Data Masking
- Intra-cluster scaling via Workload isolation

Finally, you need to interpret your results.

Interpreting results

Now that you have raw data pertaining to the performance of your data warehouse, it is important to contextualize that data. A common tactic for doing this is to compare the runs in terms of price/performance. Simply put, price/performance removes the differences in price per DWU or service hardware and gives a single comparable number for each performance test.

Let's look at an example:

Test	Test Duration	DWU	$/hr for DWU	Cost of Test
Test 1	10 min	1000	$12/hr	$2
Test 1	30 min	500	$6/hr	$3

The preceding example makes it very easy to see that running Test 1 at DWU1000 is more cost effective, at $2 per test run rather than $3 per test run. This methodology can be used to compare results across vendors in a proof of concept scenario.

In summary, once all the proof of concept tests are completed, the next step is to evaluate the results:

- Begin by evaluating whether the proof of concept goals have been met and the desired outputs collected.
- Make a note of where additional testing is warranted or where additional questions were raised.

Data Lake Exploration with Serverless SQL Pool

Preparing for your proof of concept

A proof of concept (PoC) project can help you make an informed business decision about implementing a big data and advanced analytics environment on a cloud-based platform, leveraging the serverless SQL pool functionality in Azure Synapse.

If you need to explore data in the data lake, gain insights from it or optimize your existing data transformation pipeline, you can benefit from using the serverless SQL pool resource. It is suitable for the following scenarios:

- **Basic discovery and exploration** - Quickly reason about the data in various formats (Parquet, CSV, JSON) in your data lake, so you can plan how to extract insights from it.
- **Logical data warehouse** – Provide a relational abstraction on top of raw or disparate data without relocating and transforming data, allowing always up-to-date view of your data.
- **Data transformation** - Simple, scalable, and performant way to transform data in the lake using T-SQL, so it can be fed to BI and other tools, or loaded into a relational data store (dedicated SQL pools in Azure Synapse, Azure SQL Database, etc.).

Different professional roles can benefit from serverless SQL pool:

- **Data Engineers** can explore the lake, transform and prepare data using this service, and simplify their data transformation pipelines.
- **Data Scientists** can quickly reason about the contents and structure of the data in the lake, thanks to features such as OPENROWSET and automatic schema inference.
- **Data Analysts** can explore data and Spark external tables created by Data Scientists or Data Engineers using familiar T-SQL language or their favorite tools, which can connect to serverless SQL pool.
- **BI Professionals** can quickly create Power BI reports on top of data in the lake and Spark tables.

In support of your business scenarios a serverless SQL pool POC project will identify your key goals and business drivers that a serverless SQL Pool is aligned to support and will test key features and gather metrics to support your implementation decisions.

A proof of concept is a quickly executed project that focuses on key questions and is not designed to be deployed to a production environment but is designed to execute quick tests and then be discarded.

Before you begin planning your serverless SQL Pool POC project do the following:

- Identify any restrictions or guidelines your organization has about moving data to the cloud.
- Identify executive/business sponsorship for a big data and advance analytics platform project and secure support from them for migration to cloud.
- Identify availability of technical SME and business users to support you and provide details during POC execution.

By now you should have determined that there are no immediate blockers, and you can start preparing for your POC. If you are new to Azure Synapse Analytics serverless SQL pool you can refer to the serverless SQL pool documentation where you can get an overview of the features and benefits.

If you are new to serverless SQL Pools in Azure Synapse you should compete the following learning materials:
Build data analytics solutions using Azure Synapse serverless SQL pools

Set the Goals for your POC

A successful POC project requires planning. Identify why you are doing a POC, the real motivations for the POC (like modernization, cost saving, performance improvement, integrated experience etc.) Write down clear goals for your POC and what criteria will define the success of the POC. What do you want as the outputs of your POC and what will you do with those outputs? Who will utilize the outputs? What will define a successful PoC?

Keep in mind that a POC should be a short and focused effort to quickly prove or test a limited set of concepts and capabilities – which is representative of the overall workload. If you have a long list of items to proof, you may want more than one POC with gates between them where you determine if you need the next POC. Given the different professional roles that can make use of a serverless SQL pool and the various scenarios where serverless SQL pool can be used, you may choose to plan and execute multiple POCs including serverless SQL pool; one focused on Data Scientist’s scenarios such as discovery and exploration of data in various formats, another focused on Data Engineering needs such as data transformation and another that explores creation of a Logical Data Warehouse.

As you consider your POC keep some of the following questions in mind to help you shape your goals:

- Are you migrating from an existing big data and advanced analytics platform (whether on-premises or cloud)?
- Are you migrating and want to make as few changes to existing ingestion and data processing as possible?
- Are you migrating but want to do some extensive improvements along the way?
- Are you building entirely new big data and advanced analytics platform (greenfield opportunity)?
- What are your current pain points, if any – like scalability, performance, flexibility etc.?
- What new business needs are you being asked to support?

- What are the SLAs you will be required to meet?
- What will be the workloads – Data Exploration over various data formats, Basic exploration, a logical data warehouse, Data prep and/or transformation, T-SQL interactive analysis, TSQL query of Spark Tables, reporting queries over the data lake?
- What are the skills of the users who will own the project after the POC becomes real?

Some EXAMPLEs of POC Goal Setting

- Why are we doing a POC?
  - We need to know if all of the raw file formats we will be receiving can be explored using serverless SQL pool
  - We need to know if our Data Engineers can quickly evaluate new data feeds
  - We need to know if the performance of querying data from the data lake using serverless SQL pool will need our data exploration needs
  - We need to know if serverless SQL pool is a good choice for some of our visualizations and reports
  - We need to know if serverless SQL pool is a good choice for some of our data ingestion and processing needs
  - Will more move to Azure Synapse meet our cost goals
- At the conclusion of this PoC:
  - We will have the data to identify the data transformations that are well suited to serverless SQL Pool
  - We will have the data to identify where serverless can be best used during data visualization
  - We will have the data to know the ease with which our Data Engineers and Data Scientists will be able to adopt the new platform
  - We will have gained insight to better estimate the effort required to complete the implementation project for complete migration project
  - We will have a list of items that may need more testing

Our POC will be successful if we have the data needed and have completed the testing identified to determine how serverless SQL pool in Azure Synapse will support our cloud-based big data and advance analytics platform. We will have determined if we can move to the next phase or if additional POC testing is needed to finalize our decision. We will be able to make a sound business decision backed up by the datapoints.

Plan your POC Project

Using your goals, identify specific tests to execute to support those goals and to provide the outputs you identified, it is important to make sure that you have at least one test to support each goal and expected output. Identify specific data exploration and analysis tasks, specific transformations, specific existing processing you wish to test during your so that a very specific dataset and codebase can be identified.

Example of the needed level of specificity in planning:

- (Goal) We need to know if the Data Engineering Team can execute the equivalent processing from the existing ETL process "Daily Batch Raw File Validation" within the required SLA.
- (Output) We will have the data to determine if serverless TSQL can be used to execute the "Daily Batch Raw File Validation" requirements of our existing processing and meet its SLA.
  - (Test) Validation queries A, B and C are identified by data engineering team and represents overall data processing needs. Compare the performance of these queries with the benchmark obtained from the existing system.

Evaluate Your POC Dataset

From the specific tests that you have identified, you can now identify the dataset required to support these tests. Take some time to review this dataset. You now need to verify that the dataset will adequately represent your future processing in content, complexity, and scale. Do not use a dataset that is too small - you will not see representative performance. Do not use a dataset that is too big - the POC is not the time to complete the full data migration. Also obtain the appropriate benchmarks from existing systems, which you will use for performance comparisons.

Make sure you have checked with business owners for any blockers for moving this data to the cloud. Identify any security or privacy concerns or any data obfuscation needs to be done before moving data to the cloud.

Create high-level architecture for your POC

Based upon the high-level architecture of your proposed future state architecture, identify the components that will be a part of your POC. Your high-level future state architecture likely contains many data sources, numerous data consumers, big data components and possibly Machine Learning and AI data consumers. Create an architecture for your POC that specifically identifies the components that will be part of the POC and clearly identifies which components will not be part of the POC testing.

If you are already using Azure, identify any resources you already have in place (Azure Active Directory, ExpressRoute, etc.) that can be used during the POC. Also identify what Azure Regions your organization uses. Now is a great time to identify the throughput of your ExpressRoute connection and check with other business users that your POC can consume some of that throughput without adverse effect on production solutions.

Identify POC Resources

Specifically identify the technical resources and time commitments that will be required to support your POC.

- A business representative to oversee requirements and results
- An application data expert, to source the data for the POC and provide knowledge of the existing process/logic
- An Azure Synapse serverless SQL pool expert
- An expert advisor, to optimize the POC tests
- Resources that will be required for specific components of your POC project, but not necessarily required for the duration of the POC. These resources could include network admin resources, Azure Admin Resources, Active Directory Admins, Azure Portal Admins, etc
- Ensure all the required Azure services resources have been provisioned and the required level of access has been provided to these services, including access to storage accounts
- Ensure you have an account which has required data access permission to pull data from all the data sources in the scope of the POC

Since you are evaluating a new platform, we recommend engaging an expert advisor to assist with your POC. Microsoft's partner community has global availability of expert consultants, able to demonstrate the features and performance of Azure Synapse. You can find local partners at Solution Providers Home.

Set POC Timeline

Review the details of your POC planning and business needs to identify a time constraint for your POC. Make realistic estimates of the time that will be required to complete the tasks in your plan. The time to complete your POC will be influenced by the size of your POC dataset, the number of tests, complexity, and the number of interfaces you are testing, etc. If you find your POC is estimated to run longer than 4 weeks, consider reducing the POC scope to keep focus on the highest priority goals. Get buy-in from all the lead resources and sponsors for the timeline before continuing.

Run your POC Project

Execute your POC project with the discipline and rigor of any production project. Execute according to the plan and have a change request process in place to prevent your POC from growing and getting out of control.

Example high level tasks

- Provision Synapse workspace, Storage Accounts, and all Azure resources identified in the POC plan
- Configure Networking and security according to your requirements
- Provide the required access to environment to POC team members
- Load POC dataset
- Implement and configure identified tests and/or migrate existing code to serverless SQL pool scripts and views
- Execute tests
  - Many tests can be executed in parallel
  - Record your results in a consumable and readily understandable format
- Monitor for troubleshooting and performance
- Evaluate Results and Present findings
- Plan Next Steps

Evaluate and Present results

When all the POC tests are completed you can evaluate the results. Begin by evaluating if the POC goals have been met and the desired outputs collected. Note where additional testing is warranted or where additional questions were raised.

Next Steps

Work with technical stakeholders and business to plan for the next phase of the project whether it is going to be a follow up POC or production implementation.

References

- Serverless SQL Pool in Azure Synapse Analytics
- Build data analytics solutions using Azure Synapse serverless SQL pools
- Solution Providers Home

Big Data Analytics with Apache Spark Pool

Preparing for your proof of concept

A proof of concept (PoC) project can help you make an informed business decision about migrating your on-premises big data and advanced analytics platform to a cloud-based big data and advanced analytics service, leveraging Azure Synapse Analytics for Apache Spark workloads.

A Spark POC project will identify your key goals and business drivers that cloud-based big data and advance analytics platform must support and will test key metrics and prove key behaviors that are critical to the success of your data engineering, machine learning model building and training etc. needs. A proof of concept is a quickly executed project that focuses on key questions and is not designed to be deployed to a production environment but is designed to execute quick tests and then be discarded.

Before you begin planning your Spark POC project do the following:

- Identify any restrictions or guidelines your organization has about moving data to the cloud.
- Identify executive/business sponsorship for a big data and advance analytics platform project and secure support from them for migration to cloud.
- Identify availability of technical SMEs or business users to support you and provide details during POC execution.

By now you should have determined that there are no immediate blockers and then you can start preparing for your Spark POC. If you are new to Apache Spark Pools in Azure Synapse Analytics you can refer to this documentation where you can get an overview of the Spark architecture and learn how it works in Azure Synapse.

Develop an understanding of these key concepts:

- Apache Spark and its distributed architecture
- Concepts of RDD and partitions (in-memory and physical) in Spark
- Azure Synapse workspace, different compute engines, pipeline, and monitoring
- Separation of compute and storage in Spark pool
- Authentication and Authorization in Azure Synapse
- Native connectors to integrate with dedicated SQL pool in Azure Synapse, Azure Cosmos DB etc.

Azure Synapse decouples compute resources from storage so that you can better manage your data processing needs and control costs. With the serverless architecture of Spark pool, you can spin up and down as well as grow and shrink your Spark cluster independent of your storage. You can pause (or setup auto-pause) a Spark cluster entirely so that you pay for compute only when in use and when not in use you only pay for the storage. You can scale up your Spark cluster for heady data processing needs or large loads and then scale it down during less intense processing times or shut it down completely when not needed. Scaling and pausing can be used effectively to reduce costs. Your Spark POC tests should include data ingestion and data processing at different scales (small, medium, and large) to compare price and performance at a variety of scale. Read about how to automatically scale Azure Synapse Analytics Apache Spark pools to learn more.

Understanding the difference between different sets of Spark APIs will help to decide what works best for your scenario and you can choose one over other for better performance or ease of use or to take advantage of your teams already existing skill sets. Please read A Tale of Three Apache Spark APIs: RDDs vs DataFrames and Datasets.

Data and file partitioning work slightly differently in Spark and understanding the differences will help to optimize the performance. Read Partition Discovery and Partition Configuration Options to learn more.

Set the Goals for your POC

A successful POC project requires planning. Identify why you are doing a POC, the real motivations for the POC (like modernization, cost saving, performance improvement, integrated experience, etc.) Write down clear goals for your POC and what criteria will define the success of the POC. What do you want as the outputs of your POC and what will you do with those outputs? Who will utilize the outputs? What will define a successful PoC?

Keep in mind that a POC should be a short and focused effort to quickly prove or test a limited set of concepts and capabilities – which is representative of the overall workload. If you have a long list of items to proof, you may want more than one POC with gates between them where you determine if you need the next POC. For Spark POC example, you might have two POCs, first one focused on data engineering (ingestion and processing) while the second one focused on machine learning model development.

As you consider your POC keep some of the following questions in mind to help you shape your goals:

- Are you migrating from an existing big data and advance analytics platform (whether on-premises or cloud)?
- Are you migrating and want to make as few changes to existing ingestion and data processing as possible, for example if it is Spark to Spark migration or if it is Hadoop/Hive to Spark migration?
- Are you migrating but want to do some extensive improvements along the way, for example re-writing MapReduce jobs to Spark jobs, or converting legacy RDD based code to Dataframe/Dataset based code etc.?
- Are you building an entirely new big data and advanced analytics platform (greenfield opportunity)?
- What are your current pain points, if any – like scalability, performance, flexibility etc.?

- What new business needs are you being asked to support?
- What are the SLAs you are required to meet?
- What will be the workloads - ETL, batch processing, stream processing, machine learning model training, analytics, reporting queries, interactive queries?
- What are the skills of the users who will own the project after the PoC becomes real (PySpark vs Scala skills, notebook vs IDE experience etc.)?

Some **EXAMPLE** POC Goal Setting

- Why are we doing a POC?
  - We need to know that the data ingestion and processing performance for our big data workload will meet our new SLAs.
  - We need to know whether near real time stream processing is possible and how much throughput it can support. Will it support our business requirements?
  - We need to know if our existing data ingestion and transformation processes are a good fit and where improvements will need to be made.
  - We need to know if we can shorten our data integration run times and by how much.
  - We need to know if our data scientists can build and train machine learning models and leverage AI/ML libraries as needed in Spark pool and use SQL pool, AML, or Azure Kubernetes for deployment of trained models to do scoring.
  - Will the move to cloud-based Synapse meet our cost goals?
- At the conclusion of this PoC:
  - We will have the data to determine if our data processing (for both batch and real-time stream) performance requirements can be met.
  - We will have tested ingestion and processing of all our different data types (structured, semi and unstructured) that support our use cases.

- We will have tested some of our existing complex data processing and can identify the work that will need to be completed to migrate our portfolio of data integration to the new environment.
- We will have tested data ingestion and processing and will have the datapoints to estimate the effort required for the initial migration and load of historical data as well as estimate the effort required to migrate our data ingestion (ADF, Distcp, Databox etc.).
- We will have tested data ingestion and processing and can determine if our ETL/ELT processing requirements can be met.
- We will have gained insight to better estimate the effort required to complete the implementation project.
- We will have tested scale and scaling options and will have the datapoints to better configure our platform for better price-performance settings.
- We will have a list of items that may need more testing.

Our POC will be successful if we have the data needed and have completed the testing identified to determine if Azure Synapse Analytics will support our cloud-based big data and advance analytics platform. We will have determined if we can move to the next phase or if additional POC testing is needed to finalize our decision. We will be able to make a sound business decision backed up by the datapoints.

Plan your POC Project

Using your goals, identify specific tests to execute to support those goals and provide the outputs you identified as required to make data based decisions. It is important to make sure that you have at least one test to support each goal and expected output. Identify specific data ingestion, batch or stream processing, and all other processes that will be executed so that a very specific dataset and codebase can be identified. This specific dataset and codebase will define the scope of the POC.

Examples of the needed level of specificity in planning:

- Goal) We need to know if our requirement for data ingestion and processing of batch of data can be met under our defined SLA.
- (Output) We will have the data to determine if our batch data ingestion and processing can meet the data processing requirement and SLA.
    - (Test) Processing queries A, B and C are identified as good performance tests as they are commonly executed by data engineering team and represents overall data processing needs.
    - (Test) Processing Queries X, Y and Z are identified as good performance tests as they contain near real time stream processing requirement and represent overall event-based stream processing needs.
    - (Test) Compare the performance of these queries on different scale of the Spark cluster (varying number of worker nodes, size of the worker nodes like small, medium, and large, number and size of executors) with the benchmark obtained from the existing system. Keep the “law of diminishing return” in mind – adding more resources (either by scaling up or scaling out) can help to achieve parallelism however there is certain limit, unique to each scenario, to achieve the parallelism. Find out the sweet spot for each identified use case in your testing. You can consider referring to Appendix section which provides guidelines in identifying that sweet spot.

- (Goal) We need to know if our existing data scientists can build and train machine learning models on this platform
- (Output) We will have tested some of our machine learning models by training on data in Spark or SQL pool and leveraging different machine learning libraries. This will help to determine what machine learning models can be migrated to the new environment
  - (Test) These 2-3 machine learning models (....) will be tested as part of the POC
  - (Test) Test base machine learning libraries which comes with Spark (Spark MLLib) along with additional library which can be installed on Spark (like scikit) to meet the requirement.
- (Goal) We will have tested data ingestion and will have the datapoints to 1) estimate the effort for our initial historical data migration to data lake and/or dedicated pool as well as 2) plan an approach to migrate historical data.
- (Output) We will have tested and determined the data ingestion rate achievable in our environment and can determine if our data ingestion rate is sufficient to migrate historical data during the available time window.
  - (Test) Test different approaches of historical data migration
    Transferring data to and from Azure
    Use cases - Azure Data Box
  - (Test) Identify allocated bandwidth of ExpressRoute and if there is any throttling setup by the infra team
    What is Azure ExpressRoute - Bandwidth options
  - (Test) Test data transfer rate for both online and offline data migration
    Copy performance and scalability achievable using ADF
  - (Test) Test data transfer from data lake to SQL pool using either ADF, Polybase or Copy command
    Data loading strategies for Synapse SQL pool

    Bulk load data using the COPY statement
- (Goal) We will have tested the data ingestion rate of incremental data loading and will have the datapoints to estimate the data ingestion and processing time window to data lake and/or SQL pool.

- (Output) We will have tested data ingestion rate and can determine if our data ingestion and processing requirements can be met with the identified approach
  - (Test) Test the daily update data ingestion and processing
  - (Test) Test the processed data load to SQL pool table from Spark pool
    Import and Export data with Apache Spark
  - (Test) Execute daily load update process concurrently with end user queries.

Refine your tests by adding multiple testing scenarios - The flexibility of Azure Synapse makes it easy to test different scale (varying number of worker nodes, size of the worker nodes like small, medium and large) to compare performance and behavior.

- (Spark pool test) We will execute data processing across multiple node types (small, medium, large) as well as varied number of worker nodes.
- (Spark pool test) We will load/retrieve processed data from Spark pool to SQL dedicated pool with fast SQL pool connector.
- (Spark pool test) We will load/retrieve processed data from Spark pool to Cosmos DB with Azure Synapse Link.
- (SQL pool test) We will execute tests n, m, and p using scale of 500DWU, 1000DWU, 2000DWU.
- (SQL pool test) Test load scenarios A and B using different dynamic and static resource groups.

Evaluate Your POC Dataset

From the specific data ingestion and processing pipeline that you have identified, you will now identify the dataset required to support these tests. Now take some time to review this dataset. You now need to verify that the dataset will adequately represent your future processing on Spark pool in content, complexity, and scale. Do not use a dataset that is too small (< 1TB) - you will not see representative performance. Do not use a dataset that is too big - the POC is not the time to complete the full data migration. Also ask the customer to share benchmarks from existing systems, which you will use for performance comparison.

Make sure you have checked with business owners for any blockers for moving this data to the cloud. Identify any security or privacy concerns or any data obfuscation needs to be done before moving data to the cloud.

Create a high-level architecture for your POC

Based upon the high-level architecture of your future state architecture identify the components that will be a part of your POC. Your high-level future state architecture likely contains many data sources, numerous data consumers, big data components and possibly Machine Learning and AI data consumers. Create an architecture for your POC that specifically identifies the components that will be part of the POC and clearly identifies which components will not be part of the POC testing.

If you are already using Azure, identify any resources you already have in Azure (AAD, ExpressRoute, etc.) that can be used during the POC. Also identify what Azure Regions your organization prefers. Now is a great time to identify the throughput of your ExpressRoute connection and check with other business users and verify that your POC can consume some of that throughput without adverse effect on production solutions.

Learn more about Big data architectures.

Identify POC Resources

Specifically identify the technical resources or SME and time commitment that will be required to support your POC.

- A business representative to oversee requirements and results
- An application data expert, to source the data for the POC and provide knowledge of the existing process/logic
- An Apace Spark and Spark pool expert
- An expert advisor, to optimize the POC tests
- Resources that will be required for specific components of your POC project, but not necessarily required for the duration of the POC. These resources could include network admin resources, Azure Admin Resources, Active Directory Admins, etc.
- Ensure all the required Azure services resources have been provisioned and the required level of access has been provided to these services, including access to storage accounts (Storage Contributor and Storage Blob Data Contributor).
- Ensure you have an account which has required data access permission to pull data from all the data sources in the scope of the POC.

Since you are evaluating a new platform, we recommend engaging an expert advisor to assist with your POC. Microsoft's partner community has global availability of expert consultants, able to demonstrate the features and performance of Azure Synapse. You can find local partners at Solution Providers Home.

Set your POC Timeline

Review the details of your POC planning and business needs to identify a time constraint for your POC. Make realistic estimates of the time that will be required to complete the tasks in your plan. The time to complete your POC will be influenced by the size of your POC dataset, the number data processing jobs, complexity, and the number of interfaces you are testing, etc. If you find your POC is estimated to run longer than 4 weeks, consider reducing the POC scope to keep focus on the highest priority goals. Get buy in from all the lead resources and sponsors for the timeline before continuing.

You can consider checking out "Scoping the Spark POC" section under Appendix for more tips and tricks to better scope, estimate and define timeline for your Spark POC.

Run your POC Project

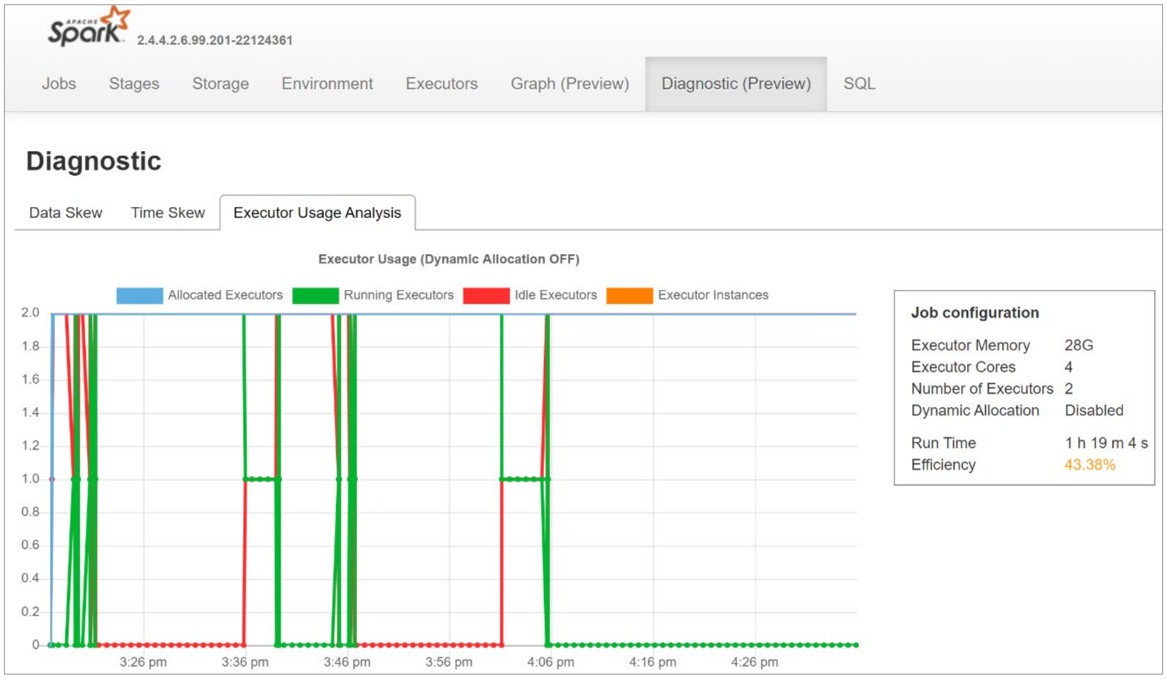
Execute your POC project with the discipline and rigor of any production project. Execute according to the plan and have a change request process in place to prevent your POC from growing and getting out of control.

Example high level tasks

- Provision a Synapse workspace, Spark and SQL pools, Storage Accounts, and all Azure resources identified in the POC plan.
- Load POC dataset
  - Make data available in Azure by extracting from source or creating sample data in Azure as needed.
    Transferring data to and from Azure
    Use cases - Azure Data Box
    Copy performance and scalability achievable using ADF
    Data loading strategies for Synapse SQL pool
    Bulk load data using the COPY statement
  - Test the dedicated connector for Spark and SQL dedicated pool.

- Migrate existing code to Spark
  - If you are migrating from Spark, your migration effort is likely to be straightforward given that Spark pool leverages open-source Spark distribution. However, if you are using vendor specific features on top of core Spark features, you will need to map these features correctly to Spark pool features.
  - If you are migrating from a non-Spark system, your migration effort will vary based on complexity involved.
  - In either case, you can consider checking out “Executing the Spark POC” section under Appendix for more best practices, guidelines, tips and tricks to successfully execute your Spark POC.
- Execute tests
  - Many tests can be executed in parallel (i.e., multiple data ingestion and processing jobs can be run in parallel across multiple Spark pool clusters)
  - Record your results in a consumable and readily understandable format
  - You can consider checking out “Testing Strategy” and “Result Analysis and Read-out” sections under Appendix for more details
- Monitor for troubleshooting and performance
  - Monitor - Apache Spark Activities
  - Monitor - Spark UI or Spark History Server
  - Monitoring resource utilization and query activity in Azure Synapse Analytics

- Monitor data skewness, time skewness and executor usage percentage under Diagnostic tab of Spark History Server:



Job configuration	
Executor Memory	28G
Executor Cores	4
Number of Executors	2
Dynamic Allocation	Disabled
Run Time	1 h 19 m 4 s
Efficiency	43.38%

Evaluate and Present results

When all the POC tests are completed you can evaluate the results. Begin by evaluating if the POC goals have been met and the desired outputs collected. Note where additional testing is warranted or where additional questions were raised.

Next Steps

Work with technical stakeholders and business interests to plan for the next phase of the project whether it's going to be another follow up POC or production migration.

References

- Automatically scale Azure Synapse Analytics Apache Spark pools
- Transferring data to and from Azure
- Copy performance and scalability achievable using ADF
- Import and Export data with Apache Spark
- Monitor - Apache Spark Activities
- Monitor - Spark UI or Spark History Server
- Monitoring resource utilization and query activity in Azure Synapse Analytics
- Manage libraries for Apache Spark in Azure Synapse Analytics
- Azure Cosmos DB Connector for Apache Spark
- Accelerate big data analytics by using the Apache Spark to Azure Cosmos DB connector
- What is Azure Synapse Link for Azure Cosmos DB
- Azure Synapse Apache Spark to Synapse SQL connector
- Choose the data abstraction
- Optimize joins and shuffles

Conclusion

An effective data proof of concept projects start with a well-designed plan and concludes with measurable test results that can be used to make data backed business decisions.

Azure Synapse is a limitless cloud-based analytics service with unmatched time to insight that accelerates the delivery of BI, AI, and intelligent applications for enterprise. You will gain many benefits from Azure Synapse, including performance, speed, improved security and compliance, elasticity, managed infrastructure, scalability, and cost savings.

This guide has provided a high-level methodology to prepare for, and execute a proof of concept to help you use Azure Synapse as a data warehouse with dedicated SQL pool, a data lake with serverless SQL pool, and/or for big data analytics with Apache Spark pool.

All the best with your proof of concept journey!

Get started today

Sign up for an Azure free account

Get more details in a free technical e-book from Packt

Speak to a sales specialist for help with pricing, best practices, and implementing a proof of concept

Microsoft