SAP HANA to Azure SQL Migration Handbook



Prepared by

**Data SQL Ninja Team (**[datasqlninja@microsoft.com](mailto:datasqlninja@microsoft.com)**)**

**Disclaimer**

The High-Level Architecture, Migration Dispositions and guidelines in this document is developed in consultation and collaboration with Microsoft Corporation technical architects.  Because Microsoft must respond to changing market conditions, this document should not be interpreted as an invitation to contract or a commitment on the part of Microsoft.

Microsoft has provided generic high-level guidance in this document with the understanding that MICROSOFT MAKES NO WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THE INFORMATION CONTAINED HEREIN.

This document is provided “as-is”. Information and views expressed in this document, including URL and other Internet Web site references, may change without notice.

Some examples depicted herein are provided for illustration only and are fictitious. No real association or connection is intended or should be inferred.

This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes.

© 2020 Microsoft. All rights reserved.

**Note**: The detail provided in this document has been harvested as part of a customer engagement sponsored through the [Data SQL Ninja Engineering](https://microsoft.sharepoint.com/sites/Infopedia_G01/Pages/Data-SQL-Ninja-Engineering.aspx).

Table of Contents

[1 Introduction 4](#_Toc89971626)

[2 Evaluation 8](#_Toc89971627)

[2.1 Evaluate Azure SQL Database 8](#_Toc89971628)

[2.2 Evaluate Azure Synapse 8](#_Toc89971629)

[2.3 Evaluate Azure SQL MI 8](#_Toc89971630)

[2.4 Evaluate Azure SQL Hyperscale 9](#_Toc89971631)

[2.5 Evaluate Azure Cosmos Database 9](#_Toc89971632)

[3 Test Results 10](#_Toc89971633)

[4 Feedback and suggestions 11](#_Toc89971634)

1. Introduction

SAP HANA is an in-memory relational database and application platform that provides high-performance analytics and real-time data processing. It is not lightweight, and hence, it hangs when used on systems that have low memories or loading power. The key feature of HANA is that it stores data in memory and organizes in column-based storage as opposed to row-based storage in traditional databases.

Like other relational databases HANA has database objects such as tables, table types, functions, and stored procedures. In addition, one of the main modeling objects in HANA is calculation views. Calculation views are modeled using the graphical interface of the HANA modeler perspective, and hence no deep SQL knowledge is required. Using calculation views, modeler can extract the required measures from more than one transaction table in the result. A developer can define calculations at different layers, use advanced SQL logic in calculation views and can create joins, projections, unions, and aggregations.

This document focuses on evaluating suitable destination for HANA database among different Azure PaaS offerings. The destination could be Azure SQL database, Managed Instance, Synapse, Hyperscale and Cosmos Database. This document can help migration planners and designers to understand the features used in source database and choose appropriate destination.

This document was prepared based on the information available at the time, for example, the capabilities of HANA Database and Azure PaaS offerings.

Objectives

Evaluate suitability of different Azure data services and identify the best fit to migrate existing on-premises databases from SAP HANA to Azure data service

* Evaluate pros and cons of suitable azure data services
* Recommend azure data service
* Database management
* Performance Evaluation

Success Criteria

* HANA vs Azure – Cost
  + Storage & Compute
  + Development cost optimization
* Speed
  + Concurrent user testing
  + Performance testing – Full data load retrieval testing while slicing and dicing for one feature

HANA to SQL mapping

|  |  |
| --- | --- |
| **HANA Database Objects** | **Equivalent SQL Database Objects** |
| **Hana Tables**  Similar to SQL table | **Tables** |
| **Table Types** Similar to SQL table types | **Table Types** |
| **Table Functions** Similar to SQL Functions | **Functions/Stored Procedures** |
| **Calculation View** GUI based views with projections/nodes | **Views** |
| **Stored Procedures** Similar to SQL Stored Procedures | **Stored Procedures** |

Challenges

The following were the major challenges that we encountered during the conversion:

* HANA was used both as analytical and transactional processing engine. There were fast response times for updates and reads of up to a second in one and half terabyte of data in HANA.
* There were calculation views used which are the major complex database object types in HANA which carried complex business logics/rules.
* Calculation Views alone accounted for majority of the conversion effort as it is GUI based views with nodes in HANA and must be recoded to Azure SQL views/ stored proc.
* Due to being graphical in nature, there is no SQL, but an XML file generated in HANA which is difficult to convert to SQL due to hierarchical complexity.

Environment Setup

**HANA database size:**

* 1.5 TB
* No. Of Active users in the system: 850, actively logged in the past 12 months: 721, actively Logged in last 6 months: 688

**Azure SQL DB:**

* 16 vCores, Business Critical Gen 5

**Azure Synapse:**

* 1000 DWU

**Azure Managed Instance:**

* 16 vCores, Business Critical Gen 5

**Azure Hyperscale:**

* 16 vCores

**Tools:**

JMeter was used as a load testing tool. It was used to test by gradually ramping up users from 1 -100 and stay 2-3 minutes for each ramp up.

Approach

One of the value propositions for modernizing to Azure databases was to minimize administration overhead. We took the approach of choosing four PaaS offerings including Azure SQL database, Hyperscale, Managed Instance and Synapse for our tests. All these automatically handle basic database management functions such as provisioning, monitoring, and backing up data. In this way, customers can avoid the complexities of maintaining a on-premises database environment, while getting a scalable service that supports modern cloud applications. For each of these, we refactored the code for a complex calculation view to make it performant to use the features of the platform more efficiently. After that, we did load testing and tuning to get the best response times.

1. Evaluation
   1. Evaluate Azure SQL Database

Azure SQL Database is a platform as a service (PaaS) offering. For SQL database, General Purpose (GP) uses Azure Premium Storage to store database files for all databases, except for the tempdb database. From the perspective of the database engine, this storage type is remote, i.e., it is accessed over the network, using Azure network infrastructure.

[Business Critical](https://docs.microsoft.com/en-us/azure/sql-database/sql-database-managed-instance#business-critical-service-tier) (BC) instances do not use remote Azure Premium Storage, but instead use local SSD storage. For our tests, we used Business Critical tier due to low latency requirements and refactored code to use indexes. In tests for 100 users response time was 20.5 seconds which was relatively slow and hence, we didn’t consider this as suitable destination for HANA workload we had.

* 1. Evaluate Azure Synapse

Azure Synapse is built for OLAP systems and has been designed as a Massively Parallel Processing (MPP) system, designed to perform analytics on large data. Migrating to Azure Synapse Analytics from SAP HANA required some code changes that took some time to implement.

We refactored the code to implement below best practices:

1. Enable the result set cache.
2. Update statistics on the tables
3. Use [Hash distribution for large tables](https://nam06.safelinks.protection.outlook.com/?url=https%3A%2F%2Fdocs.microsoft.com%2Fen-us%2Fazure%2Fsynapse-analytics%2Fsql%2Fbest-practices-dedicated-sql-pool%23hash-distribute-large-tables&data=04%7C01%7Cguptarahu%40microsoft.com%7C768161f4c6d54880699408d98dbff669%7C72f988bf86f141af91ab2d7cd011db47%7C1%7C0%7C637696677949587662%7CUnknown%7CTWFpbGZsb3d8eyJWIjoiMC4wLjAwMDAiLCJQIjoiV2luMzIiLCJBTiI6Ik1haWwiLCJXVCI6Mn0%3D%7C1000&sdata=P%2FwvUQVvuB1zpXMdqqoq8fnMyN%2BmGoSkRZB0prA%2FlZk%3D&reserved=0).
4. [Optimize clustered columnstore tables](https://nam06.safelinks.protection.outlook.com/?url=https%3A%2F%2Fdocs.microsoft.com%2Fen-us%2Fazure%2Fsynapse-analytics%2Fsql%2Fbest-practices-dedicated-sql-pool%23optimize-clustered-columnstore-tables&data=04%7C01%7Cguptarahu%40microsoft.com%7C768161f4c6d54880699408d98dbff669%7C72f988bf86f141af91ab2d7cd011db47%7C1%7C0%7C637696677949592655%7CUnknown%7CTWFpbGZsb3d8eyJWIjoiMC4wLjAwMDAiLCJQIjoiV2luMzIiLCJBTiI6Ik1haWwiLCJXVCI6Mn0%3D%7C1000&sdata=HseSHAgz9rSZKdRHdyN9r6LS8iOLF5sfXLrvr%2BKFzcU%3D&reserved=0).
5. [Use larger resource class to improve query performance](https://nam06.safelinks.protection.outlook.com/?url=https%3A%2F%2Fdocs.microsoft.com%2Fen-us%2Fazure%2Fsynapse-analytics%2Fsql%2Fbest-practices-dedicated-sql-pool%23use-larger-resource-class-to-improve-query-performance&data=04%7C01%7Cguptarahu%40microsoft.com%7C768161f4c6d54880699408d98dbff669%7C72f988bf86f141af91ab2d7cd011db47%7C1%7C0%7C637696677949602659%7CUnknown%7CTWFpbGZsb3d8eyJWIjoiMC4wLjAwMDAiLCJQIjoiV2luMzIiLCJBTiI6Ik1haWwiLCJXVCI6Mn0%3D%7C1000&sdata=LYwP1RdkyLyTe5kij0jMlit4YCbavK8vlXxtScDRqG4%3D&reserved=0).

After implementing above, individual row inserts and updates were still slow taking an average of 11 seconds. The data volume was 1.5 Terabytes uncompressed which was relatively small by the size of large data warehouses deployed in Synapse. Looking at these drawbacks we didn’t find Synapse as a suitable target for this modernization effort.

* 1. Evaluate Azure SQL MI

Azure SQL Managed Instance is a platform as a service (PaaS) offering that’s nearly 100% compatible with the latest SQL Server Enterprise edition. Like SQL Database, for Managed Instance General Purpose (GP) uses Azure Premium Storage to store database files for all databases, except for the tempdb database. From the perspective of the database engine, this storage type is remote.

Managed Instance [Business Critical](https://docs.microsoft.com/en-us/azure/sql-database/sql-database-managed-instance#business-critical-service-tier) (BC) instances do not use remote Azure Premium Storage, but instead use local SSD storage. For our tests, we used Business Critical tier due to low latency requirements and refactored code to use indexes. After refactoring for 100 users response time was 14.3 seconds which was relatively slow and hence, we didn’t consider this as suitable destination for HANA workload we had.

* 1. Evaluate Azure SQL Hyperscale

Hyperscale is suitable for HTAP (Hybrid Transactional Analytical Performance) workloads with fluid vertical and horizontal compute scaling. The Hyperscale service tier in Azure SQL Database is the newest service tier in the vCore-based purchasing model. This service tier leverages the Azure architecture to scale out the storage and compute resources for an Azure SQL Database substantially beyond the limits available for the General Purpose and Business Critical service tiers. It has higher overall performance due to higher transaction log throughput and faster transaction commit times regardless of data volumes. Hyperscale is a symmetric multi-processing (SMP) architecture and is not a massively parallel processing (MPP) or a multi-master architecture. You can only create multiple replicas to scale out read-only workloads.

Given the 1.5 TB size and HTAP requirements for our workloads, Hyperscale was a great option. Since we were not running data analytics on a large scale with complex queries using Massively Parallel Processing (MPP) data warehouses, Azure Synapse Analytics was not good choice here and it made more sense to use Hyperscale. We ran tests for one and hundred users which also proved the same and found the response time within 1 second and 13.6 second which was much better than response times from Azure SQL, MI or Synapse.

* 1. Evaluate Azure Cosmos Database

Cosmos database is a No-SQL database service that is globally distributed and is suitable for real time processing. Going with this option would have taken more time, costs and effort therefore it was not considered.

1. Test Results

Graphical user interface

Description automatically generated

The above test results show that Hyperscale gives optimal performance as compared to other PaaS offerings.

1. Feedback and suggestions

If you have feedback or suggestions for improving this data migration asset, please contact the Data Migration Jumpstart Team (askdmjfordmtools@microsoft.com). Thanks for your support! Note: For additional information about migrating various source databases to Azure, see the Azure Database Migration Guide.