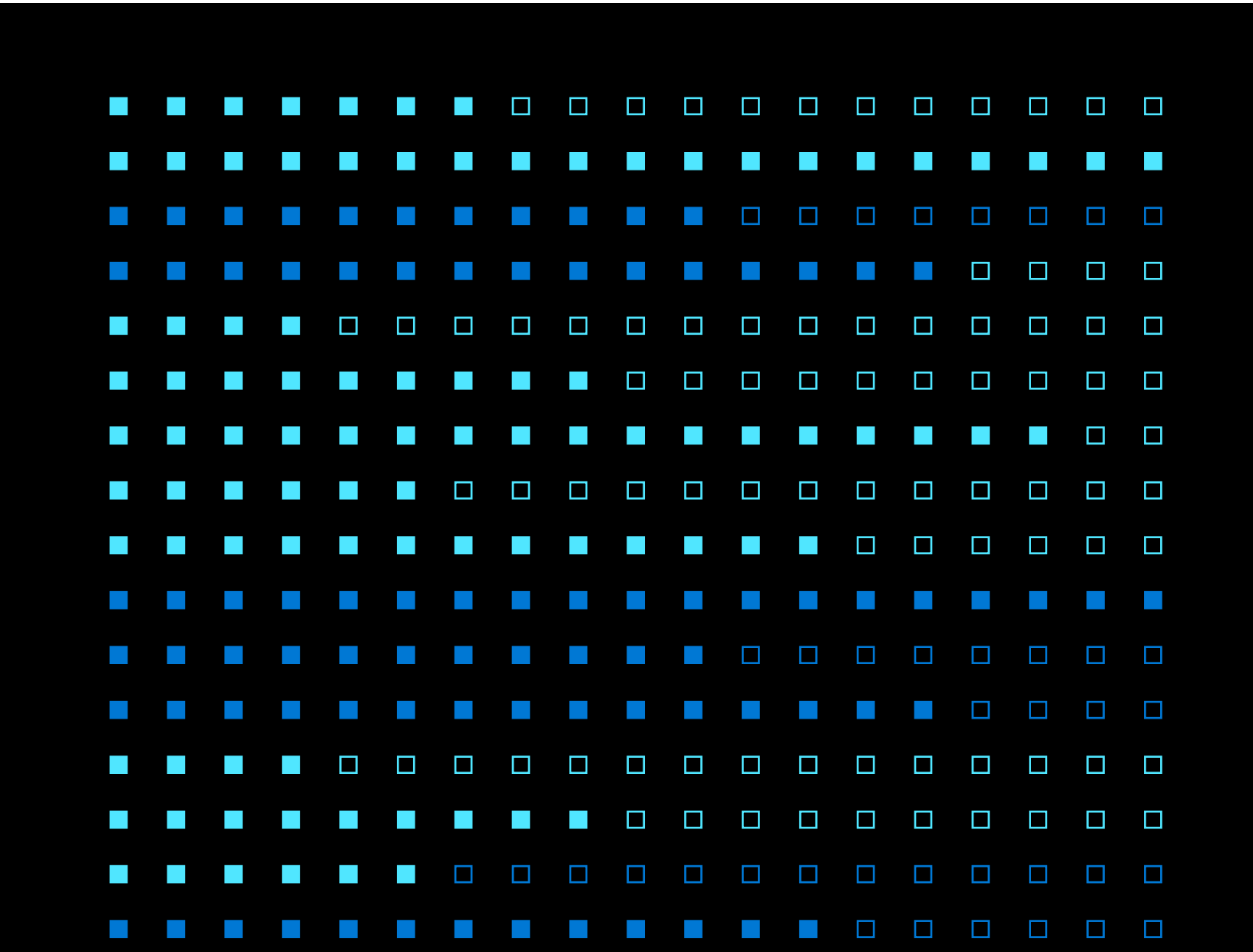
Microsoft SQL Server 2019 Big Data Clusters

Technical white paper

Published: November 2019
Applies to: Microsoft SQL Server 2019



Copyright

Microsoft

Contents

Introduction

Getting value from relational and big data doesn't have to be complicated

Today, connected devices and people are generating volumes of data that exceed the storage capacity of any traditional relational database system. By some estimates, 1.7MB of data is created every second for every person on earth.[1]

Many organizations need to use this data for AI efforts that can deliver better experiences for customers, develop new revenue streams, reduce costs, and drive business growth. At the same time, they need to combine it with traditional relational data such as orders, customers, and transactions to get an accurate picture of the business and its future.

Currently, this isn't easy. Big data is stored in one place and enterprise data in another. These different storage types use different technology and systems for management, security, structure, and queries. To be able to take advantage of big and relational data, complex solutions have developed over the years that include moving data from one place to another and combining it for analysis or using special technologies to run open-source programming languages and code on both data types to build analytics dashboards and machine learning models.

Even with recent advancements in distributable, scalable compute for petabytes and exabytes of data, joining big and relational data is time-consuming and resource intensive. It often delays critical insights from reaching business decision makers. Although SQL Server has kept pace, it was not designed to be a database engine for analytics on the scale of petabytes or exabytes, nor for storing and analyzing data in unstructured formats, such as media files.

The introduction of Big Data Clusters in SQL Server 2019 solves the compute and storage challenges of scaling to petabytes and exabytes of data while offering more flexible processing through Spark (a unified analytics engine for large-scale data processing). Big data clusters bring together multiple instances of SQL Server with Spark and HDFS, making it much easier to unite relational and big data and use them in reports, predictive models, applications, and AI.

SQL Server Big Data Clusters make it easy to unite high-value relational data with high-volume big data.

SQL Server Big Data Clusters unite relational and big data

Big data clusters make it easy to integrate, manage, and analyze, big data and relational data using tools preferred by data scientists and data engineers to drive applications and reporting.

Using either Transact-SQL (T-SQL) or Spark, you can easily combine and analyze relational data and big data. HDFS integration creates an elastic storage layer that can scale to petabytes of data storage. The Spark engine processes and analyzes high-volume data in a scalable, distributed, in-memory compute layer.

[1] DOMO, Data Never Sleeps 6, https://www.domo.com/solution/data-never-sleeps-6

The release of SQL Server 2019 allows for the integration of Spark and HDFS to create big data clusters. Now you can integrate, manage, and analyze across all your data.

With SQL Server Big Data Clusters, you have the flexibility to:

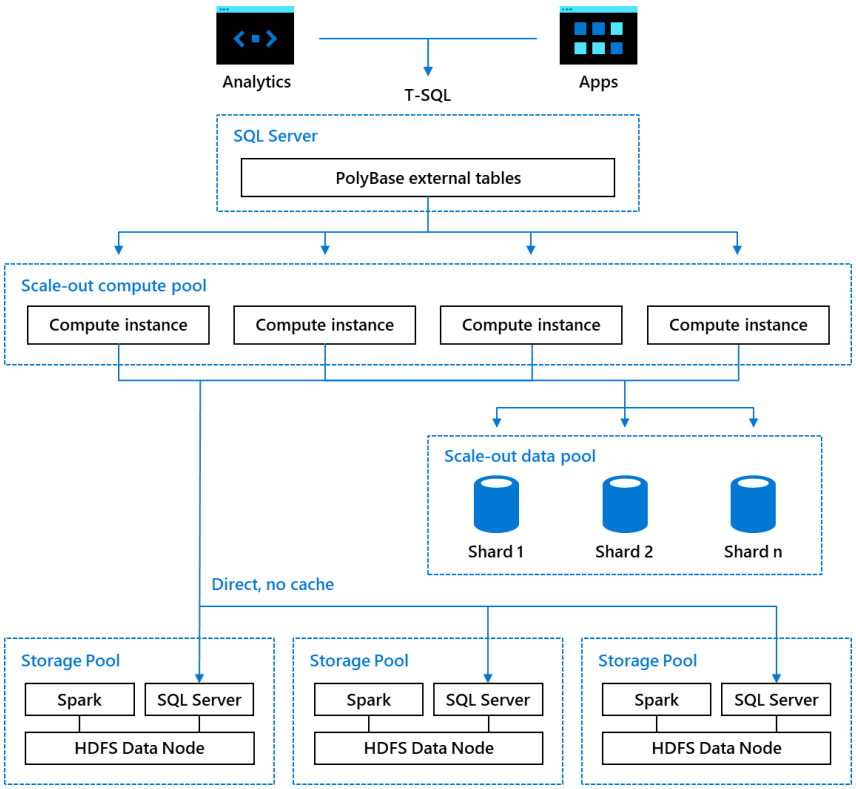
- Integrate relational and big data without having to copy or move it by ingesting your data via Spark Streaming or traditional SQL inserts and storing it in HDFS, relational tables, graph, or JSON/XML.
- Leverage data virtualization, and connect to data sources outside of the cluster using out-of-the-box connectors for Oracle, MongoDB, Teradata, and more.
- Manage data using built-in cluster management tools and services, like log analytics, monitoring, and high availability that provide a consistent experience wherever your clusters are deployed.
- Leverage AI using Spark jobs or T-SQL queries to prepare data and feed it into machine learning model training routines. This can be done through Spark or the SQL Server master instance using a variety of programming languages—Java, Python, R, and Scala. The models can be operationalized via batch scoring jobs in Spark, T-SQL stored procedures for real-time scoring, or in containerized applications exposed through REST APIs.

With SQL Server Big Data Clusters, existing SQL Server customers can do more with their enterprise data lakes. At the same time, data scientists can continue to use tools from the Hadoop ecosystem while gaining real-time access to data in SQL Server. The big data cluster architecture makes these new scenarios—and others—possible.

Big data cluster architecture

Like other relational databases, SQL Server has historically been a scale-*up* system. To handle more data or get better performance, you need a bigger or faster server. Although the computing power and storage capacity of today's servers is staggering, the size of any single server eventually reaches its physical limits. In SQL Server Big Data Clusters, big data technology is integrated into the SQL Server engine, so it can *scale-out* compute and storage independently.

When SQL Server 2017 added support to run on Linux, it opened the possibility of deeply integrating SQL Server with Spark, HDFS, and other big data components that are primarily Linux-based. Big data clusters in SQL Server 2019 take that to the next step by fully embracing the modern architecture of deploying applications—even stateful ones like a database—as containers on Kubernetes.



Big data cluster architecture in SQL Server 2019

A big data cluster is a compute cluster of containers running SQL Server and big data services, marshalled by a SQL Server master instance. Each cluster contains SQL Server, Spark, and HDFS containers running on Kubernetes and supported by Microsoft.

Kubernetes is an open-source orchestration platform that makes it easier to manage the deployment of logical groups of containers called pods. A pod runs on a node, which is either a virtual or physical worker machine in Kubernetes. Each node is managed by the master. A node can have multiple pods, and the Kubernetes master automatically assigns pods to nodes in the cluster. The master's automatic scheduling takes into account the available resources on each node.

Deploying a big data cluster on Kubernetes ensures a predictable, fast, and elastically scalable deployment, regardless of the location of the deployment.

Pools enable independent, customizable compute and storage scale-out

Pods are grouped into different types of pools. Each pool can be independently scaled up and down to adjust the number of pods to meet changes in demand:

- A **compute pool** is made up of one or more SQL Server compute instances running in Kubernetes. Each instance contains a set of base services and an instance of the SQL Server database engine.
- A **storage pool** is a group of pods containing a SQL Server engine, HDFS data node, and Spark containers. This provides the scalable storage tier along with the collocated compute for SQL Server and Spark right next to the data.

- A **data pool** is a group of SQL Server instances that is used either to stage data from an external source or to store an incoming stream of append-only data. In either case, the data is partitioned and distributed across all the SQL Server instances in the pool.
- The **master pool** is a special, singleton pool of SQL Server pods that can be either a singleton SQL Server or a SQL Server deployed as multiple instances in an Always On Availability Group for high availability and read scale out. This SQL Server instance is where read-write OLTP or dimensional data is stored in a big data cluster.
- A **Spark pool** enables Spark compute separate from HDFS. A Spark pool can separate Spark compute from the storage layer for better resource management and to optimize the type of hardware the two pools require.

The benefits of the big data cluster architecture

Easy data integration

Through the SQL Server Big Data Clusters architecture, Microsoft has created a way for you to integrate, manage, and analyze all your data using the preferred skills and tools of your data engineers, data analysts, and savvy business users. The value of the big data greatly increases when it is combined with the high-value data stored in SQL Server through reports, dashboards, and applications. SQL Server 2019 Big Data Clusters makes all your data available in one all one integrated system and accessible through big data tools or SQL Server tools.

Optimized and enhanced query performance

In some data query scenarios, data virtualization across multiple data sources can be inherently slower than reading the data from a single system. To alleviate the performance impact of virtualization, PolyBase and SQL Server Big Data Clusters employ a variety of technology and techniques that parallelize and scale-out compute and cache data.

PolyBase optimizes performance by using push-down computation. Operations such as projections, predicates, aggregates, limits, and homogeneous joins are pushed to the source system to take advantage of the query optimizer in each of the source systems. Only the filtered results of these operations are returned to SQL Server, which improves performance by reducing the amount of data to transfer.

In big data clusters, the SQL Server engine has gained the ability to natively read HDFS files, such as CSV and parquet files, by using SQL Server instances collocated on each of the HDFS data nodes to filter and aggregate data locally in parallel across all of the HDFS data nodes.

Performance of PolyBase queries in big data clusters can be boosted by distributing the cross-partition aggregation and shuffling the filtered query results to "compute pools" comprised of multiple SQL Server instances that work together. Compute pools are similar to scale-out groups with PolyBase in SQL Server on Windows. Unlike the scale-out groups of PolyBase in a SQL Server 2017 instance, a big data cluster can have a compute pool with up to eight SQL Server instances. Scale-out nodes in a SQL Server 2017 PolyBase scale-out group must be individually installed and configured, but compute pools with many SQL Server instances can be provisioned as containers on a Kubernetes cluster with a single command or API call in seconds.

Throughout the rest of this white paper, we'll explore the key scenarios for using big data clusters and take a closer look at the new capabilities and enhancements in SQL Server 2019 that enable big data.

Scenarios enabled by SQL Server Big Data Clusters

Data virtualization

Modern enterprises store data in a mixture of relational and non-relational data stores—often from several different vendors. This presents a challenge for developers, data scientists, and business analysts who want to combine these disparate sources for analysis. In the past, the only way to reliably do this was to move copies of the data to a single platform via extract-transform-load (ETL) processes. However, ETL isn't right for every scenario, especially when a data warehouse isn't in the long-term plan because:

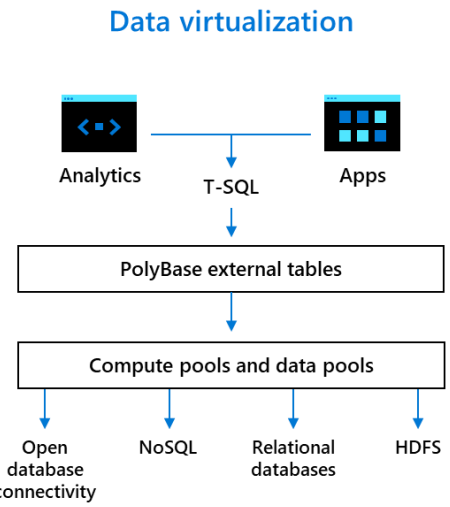
- ETL is expensive to develop, maintain, and support.
- Data from an ETL pipeline lags behind business.
- Running multiple copies of data poses security threats and requires expensive disk space.

Data virtualization offers a fast, cost-effective alternative to ETL.

Based on PolyBase technology, data virtualization works by integrating data from disparate sources, locations, and formats, without replicating or moving the data. It creates a single "virtual" data layer that delivers unified data services to support multiple applications and users. The virtual data layer allows users to query data from many sources through a consistent interface. User access to sensitive data sets can be controlled from a single location, and the processes that cause delays using ETL are not applicable with virtualization, meaning queried data sets are up to date.

Big data clusters integrate structured and unstructured data

With big data clusters, you can combine data from external data sources without moving or replicating it by leveraging enhancements to SQL Server 2019 PolyBase.



PolyBase integrates data from disparate sources and locations.

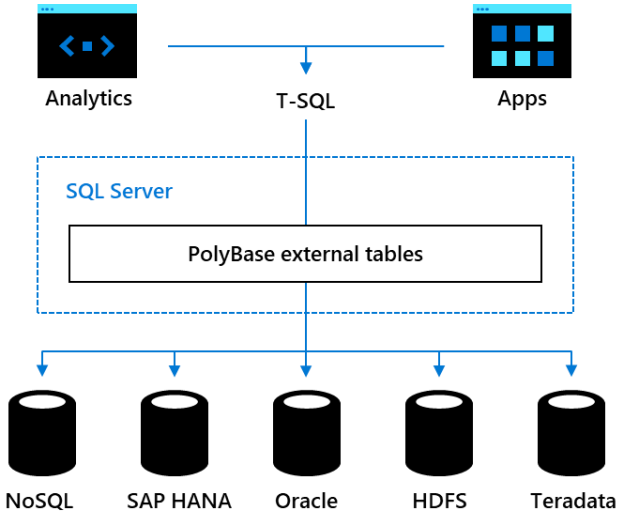
First added to the SQL Server database engine in SQL Server 2016, PolyBase enables applications and users to query big data stored in HDFS-compatible Hadoop distributions and file systems such as HortonWorks, Cloudera, and Azure Blob Storage by using T-SQL to define an external table to represent HDFS data in SQL Server. Users or applications can run T-SQL queries that reference the external table as if it were a normal SQL Server table. When

the query is executed, data from the external data source is retrieved and returned to the user, but it is not stored in persistent storage in SQL Server.

Using this approach of querying data from the source ensures that there are no data latencies, duplicated storage costs, or data-quality issues which may be introduced using ETL pipelines. Once you have created external tables in SQL Server, you can control access to data sources by granting access to external tables to Active Directory users and groups, thus centralizing the data access policies to a single location.

SQL Server 2019 extends the capabilities of PolyBase with new connectors to create external tables that link to a variety of data stores. Given these enhancements to PolyBase, a big data cluster can act as a data hub by integrating structured and unstructured data from across the entire data estate– SQL Server, Azure SQL DB, Azure SQL DW, Oracle, Teradata, MongoDB, Azure Cosmos DB, HDFS, and more–using familiar programming frameworks and data analysis tools.

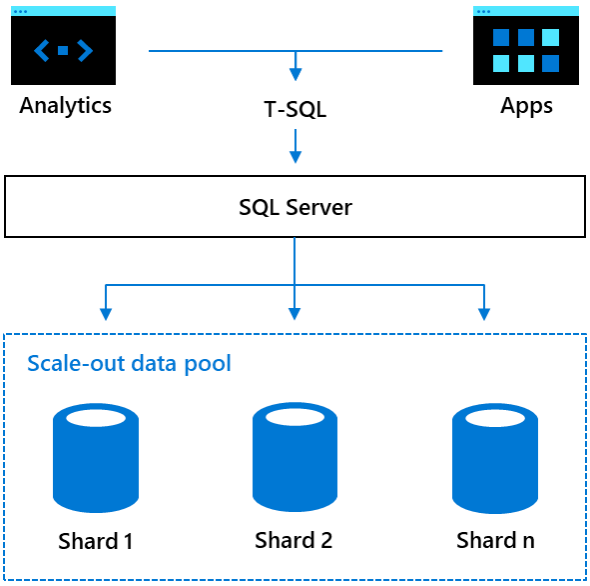
> With the latest enhancements to PolyBase, SQL Server Big Data Clusters can act as a data hub to integrate structured and unstructured data from across the entire data estate using familiar programming frameworks and data analysis tools.



Data sources that can be integrated by PolyBase in SQL Server 2019

Scale-out data marts

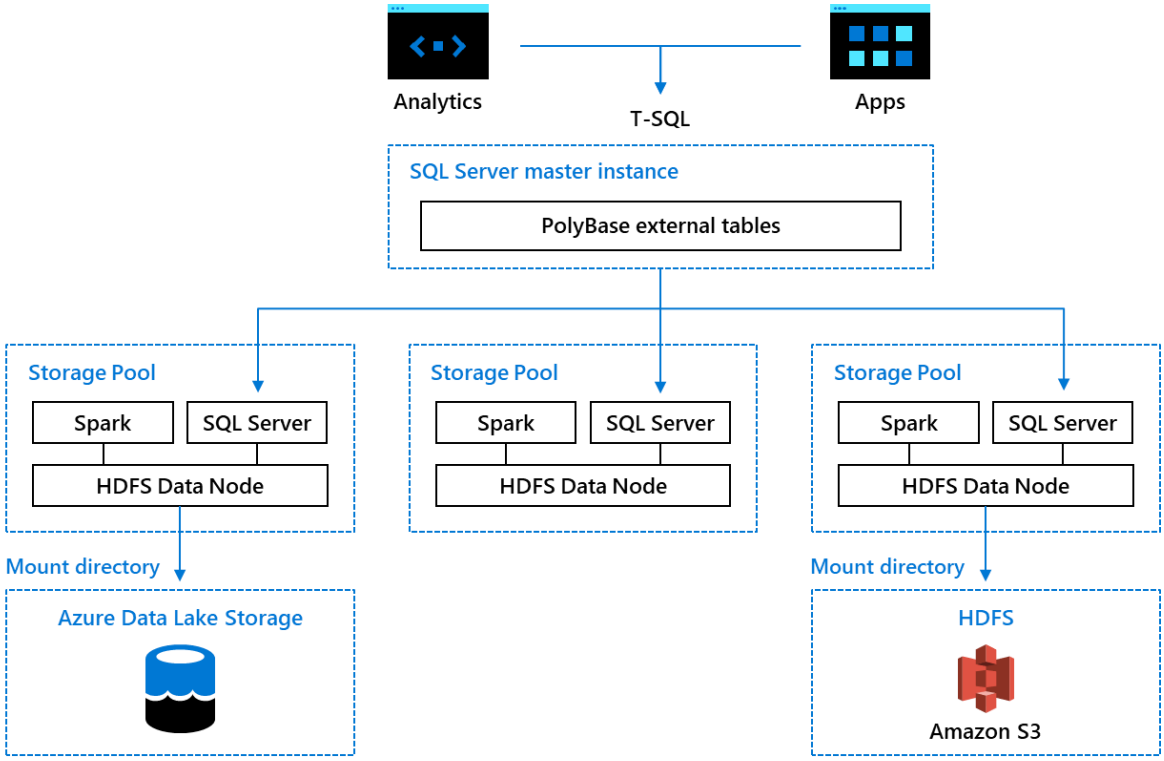
With big data cluster data pools, data from external data sources can be partitioned and cached across all the SQL Server instances in a data pool, creating a scale-out data mart. There can be more than one scale-out data mart in any given data pool. A data mart makes it easy to integrate and cache combined data sets from multiple external sources.



Using a scale-out data pool to cache data from external data sources for better performance

A shared data lake

For persistence in a big data cluster, SQL Server instances in the storage pool can read from and write data to parquet and CSV files in HDFS. Applications and analytics that query the data through the SQL Server master instance and Spark jobs can enable different analytics scenarios, using tools, languages, and skills of choice. Both SQL Server and Spark can read and write these files, creating a shared data lake accessible by many different types of users, tools, and systems. Using HDFS tiering, you can cache data from the source to big data clusters to expand existing data lakes.



SQL Server and Spark are deployed together with HDFS creating a shared data lake

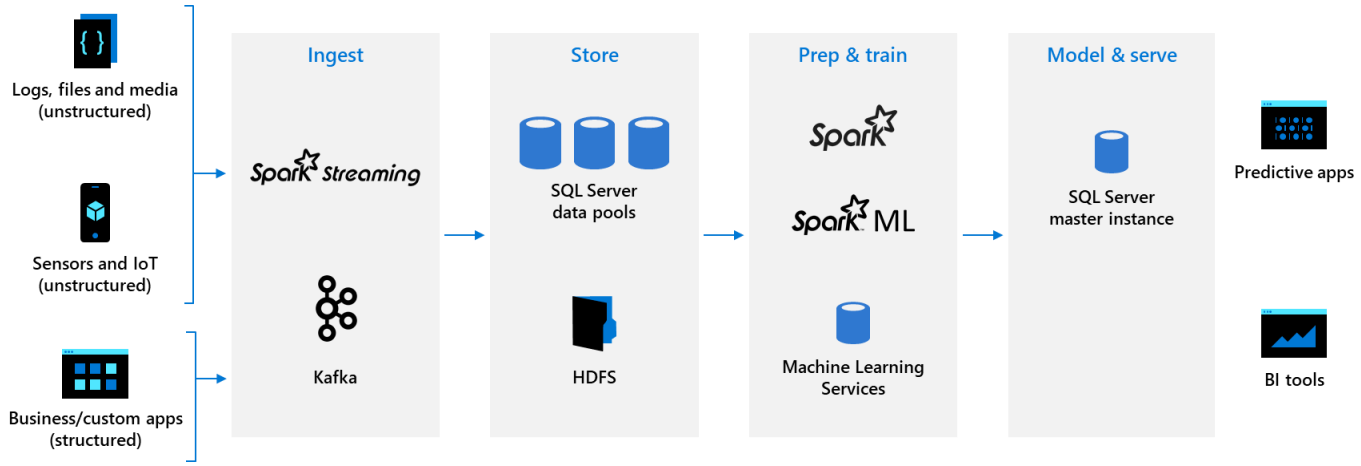
AI and machine learning

Artificial intelligence and machine learning are frequent headliners in the news today. Organizations in every industry are rapidly hiring data scientists and setting up systems to collect data and use it to train models. The outcomes from predictive analytics is a function of the four Vs of big data.

Big data clusters make machine learning easier and more accurate by improving the four Vs of big data.

	The impact of the Vs on analytics	How a big data cluster helps
Volume	The greater the volume of data processed by a machine learning algorithm, the more accurate the predictions will be	Big data clusters increase the data volume available for AI by capturing data in scalable, inexpensive big data storage in HDFS and by integrating data from multiple sources using PolyBase connectors
Variety	The greater the variety of different sources of data, the more accurate the predictions will be	Big data clusters increase the number of varieties of data available for AI by integrating multiple data sources through the PolyBase connectors
Veracity	Accurate machine learning depends on the quality of the data going into the model training	Big data clusters increase the veracity of data available for AI by sharing data without copying or moving data, which introduces data latency and data quality issues SQL Server and Spark can both read and write into the same data files in HDFS
Velocity	Real-time predictions depend on up-to-date data flowing quickly through the data processing pipelines	Big data clusters increase the velocity of data to enable AI by using elastic compute and caching to speed up queries

Big data clusters provide all the tools and systems to ingest, store, and prepare data for analysis as well as to train the machine learning models, store the models, and operationalize them.



A complete AI platform: SQL Server 2019 Big Data Clusters

- **Ingest:** Data can be ingested using Spark Streaming by inserting data directly to HDFS through the HDFS API or by inserting data into SQL Server through standard T-SQL insert queries.
- **Store:** The data can be stored in files in HDFS, partitioned and stored in data pools, or stored in the SQL Server master instance in tables, graph, or JSON/XML.

- **Prep and train:** Either T-SQL or Spark can be used to prepare data by running batch jobs to transform the data, aggregate it, or perform other data wrangling tasks. Data scientists can choose to use either SQL Server Machine Learning Services in the master instance to run R, Python, or Java model training scripts or to use Spark. In either case, the full library of open-source machine learning libraries, such as TensorFlow or Caffe, can be used to train models.
- **Model and serve:** Once the models are trained, they can be operationalized in the SQL Server master instance using real-time, native scoring via the PREDICT function in a stored procedure; or you can use batch scoring over the data in HDFS with Spark. Alternatively, using tools provided with the big data cluster, data engineers can easily wrap the model in a REST API and provision the API + model as a container on the big data cluster as a scoring microservice for easy integration into any application.

This entire pipeline takes place in the context of a big data cluster. The data never leaves the security and compliance boundary to go to an external analytics server or a data scientist's laptop. The full power of the hardware underlying the big data cluster is available to process the data, and the compute resources can be elastically scaled up and down as needed.

New tools and services complement the AI/ML platform

A unified analytics experience provided by Azure Data Studio

Big data clusters leverage Azure Data Studio to provide unified analytics. Azure Data Studio is an open-source, multi-purpose data management and analytics tool for DBAs, data scientists, and data engineers. When connected to a big data cluster endpoint, it provides an enterprise view of the data estate with the high-value data in the relational databases and the high-volume data stored in the HDFS instances within the cluster. The new HDFS browser lets analysts, data scientists, and data engineers easily view the HDFS files and directories in the big data cluster, upload/download files, open them, and delete them if needed.

Data scientists can quickly analyze the HDFS files using the Notebook experience included in Azure Data Studio to ingest, store and prep, and train the data to produce models. The experience is built on Jupyter, enabling data scientists and engineers to write Python, R, or Scala code with Intellisense and syntax highlighting before submitting the code as Spark jobs and viewing the results inline.

Included in the Notebooks is PROSE SDK from Microsoft Research, which makes it easy to normalize the chaos in your data by analyzing raw data and producing the code that needs to be executed on it to convert it to a normalized and cleaned-up version of the data. For example, the PROSE-generated code can be used in Spark jobs to prep data for ingestion into SQL Server or to feed into model training or scoring.

Spark developers are also able to submit jobs as .jar files in the cluster and Spark applications from their common development environments in VSCode as well as IntelliJ. The new Spark-SQL connector is based on SQL Bulk Copy APIs, which enable you to write batch and streaming data from Spark to SQL Server and the Data pools.

Machine learning model building with Notebooks

Data scientists can download and install a set of packages in Notebooks, which can make it easier for users to perform complex tasks without rewriting many lines of code. With SQL Server Big Data Clusters, you can install any of the PyPI packages locally to build your own machine learning models. To submit the job against the cluster, you can install the packages through the Spark context or inside SQL Server using sqlmlutils, which is designed to

help users interact with SQL databases (SQL Server and Azure SQL Database) and execute R or Python code in SQL from an R/Python client.

Native support for Jupyter Books provides a collection of Jupyter Notebooks and Markdown files. Supported Jupyter books can help you accomplish tasks and solve data science problems focused on industry verticals, such as retail, finance, healthcare, public sector, manufacturing, and agriculture.

Model lifecycle management with MLFlow and SQL Server

Analytical models must be trained, compared, and monitored before deploying into production. Once data scientists, data engineers, business analysts, and machine learning engineers have successfully built their machine learning models, they need model management—a system that manages and orchestrates the entire lifecycle of machine learning models to accelerate operationalization.

MLFlow, an open source platform, manages the machine lifecycle in SQL Server 2019, including experimentation, reproducibility, and deployment. One of the features Microsoft has introduced into MLFlow is the ability to store models into backend SQL Server as the model artifact store.

Models are stored as serialized varbinary objects in SQL Server, just like data is stored. SQL Server keeps the models close to data, so all the capabilities of a Management System for Data can be transferred to machine learning models. This can help simplify the process of managing models tremendously which results in faster delivery and more accurate business insights.

> The SQL Server engine treats models just like data, which means many of the mission-critical features of data management already built into SQL Server can be used for machine learning models too.

Support for machine learning models

SQL Server Big Data Clusters provides capabilities that support the performance, security and compliance, and availability and scalability of your machine learning models.

Performance

- **Fast model training and scoring using operational analytics:** Use in-memory OLTP and/or in-memory columnstore.
- **Monitoring and optimizing model performance via Query Store and DMVs:** Query Store is like a "black box" recorder on an airplane, recording how queries have executed and simplifying performance troubleshooting. You will be able to quickly see database usage patterns and find performance differences caused by changes in query plans from an automatically captured history of queries, plans, and runtime statistics that separates data by time windows.
- **Hierarchical model metadata:** Expanded support for unstructured JSON data inside SQL Server enables you to store properties of your models in the JSON format. Process JSON data just like any other data inside SQL. Organize collections of your model properties and establish relationships between them. Combine strongly typed scalar columns stored in tables with flexible key/value pairs stored in JSON columns, and query both scalar and JSON values in one or multiple tables using full Transact-SQL. Store JSON in In-memory or Temporal tables, apply Row-Level Security predicates on JSON text, and so on.

- **Temporal support for models:** Use temporal tables to keep track of the state of models at any specific point in time, so you can understand model usage trends over time, monitor track model changes over time, audit all changes to models, and recover from accidental model changes and application errors.

Security and compliance

- **Sensitive model encryption via Always Encrypted:** Protect models at rest and in motion by requiring the use of an Always Encrypted driver when client applications communicate with the database and transfer data in an encrypted state.
- **Transparent Data Encryption (TDE):** Encrypt an entire database that stores machine learning models. Backups for databases that use TDE, the primary SQL Server encryption option, are also encrypted. TDE protects the data at rest and is completely transparent to the application and requires no coding changes to implement.
- **Row-level security:** Set permissions for users to only see the models (rows) relevant to them.
- **Dynamic model (data) masking:** obfuscates a portion of the model data to unauthorized viewers e.g., credit card numbers).
- **Change model capture:** Record, insert, update, and delete activity applied to models stored in tables in SQL Server and make the details of the changes available in an easily consumed relational format. The change tables used by change data capture contain columns that mirror the column structure of a tracked source table, along with the metadata needed to understand the changes that have occurred.
- **Enhanced model auditing:** Implement user-defined audit, audit filtering, and audit resilience.

Availability and scalability

- **Always On availability groups:** An availability group in SQL Server serves as a failover environment, supporting a set of primary databases and one to eight sets of corresponding secondary databases. Secondary databases are not backups. In addition, you can have automatic failover based on database health. Availability groups with readable secondaries enable a "champion-challenger" model setup, whereby the champion model runs on a primary and challenger models are scoring and being monitored on the secondaries for accuracy (without having any impact on the performance of the transactional database). Whenever a new champion model emerges, it's easy to enable it in the primary.
- **Enhanced model caching:** Facilitate model scalability and high performance. SQL Server enables caching with automatic, multiple TempDB files per instance in multi-core environments.

Built-in management services and deployment options

Controller service helps you manage and secure your clusters

The big data cluster controller service coordinates the behavior of the cluster. It orchestrates the provisioning and deprovisioning of pods and pools through Kubernetes APIs and manages the allocation of resources. The controller service is responsible for many aspects of the cluster lifecycle, including:

- Initializing the compute, data, and storage pools when the cluster is first installed
- Managing scaling operations of the cluster
- Configuring high availability through SQL Server availability groups
- Managing software updates
- Configuration management
- Monitoring and troubleshooting

The controller service is also responsible for security, including cluster authentication, cluster authorization, and rotation of the certificates used to allow nodes to communicate securely within the cluster.

On-premises and cloud-based deployment options

Because big data clusters are composed of containers orchestrated by Kubernetes, administrators a choice about where to deploy: on-premises on a Kubernetes cluster deployed via a kubeadm, on a hosted cloud in VMs with Kubernetes in them, or on a managed Kubernetes service, such as Azure Kubernetes Service (AKS).

Automated, customizable, and flexible management and deployment

Big data clusters are considered self-managed because of their tight integration with Kubernetes and components that ensure automatic monitoring, security and more. Administrators can reduce the operational overhead of a traditional big data cluster by relying on the automated services for provisioning, high availability, and monitoring that are built into big data clusters. A REST API and the azdata command-line utility open options for automation of management tasks.

Your choice of management, monitoring, and security tools

Management

SQL Server Big Data Clusters include management and monitoring out of the box, using open-source components. Choose from a combination of command line tools (including azdata, a command-line utility written in Python), APIs, and dynamic management views to accommodate your existing skillsets.

Monitoring

SQL Server Big Data Clusters have built-in monitoring and troubleshooting components that provide automatic metrics and log collection. These are exposed through corresponding dashboards that users can leverage to query metrics and logs. Active Direct Services also has big data cluster dashboards that expose the health of the cluster and guide you to troubleshoot any issues using built-in notebooks.

Job Management UI, a big data administration tool, renders detailed views into the health and performance of Spark and the jobs running in the big data cluster. SQL Server 2019 also includes some unique extensions to the Job Management UI that graphically visualize the job steps, including detailed resource consumption and timing metrics as well as a capture of the files that were accessed by a given job.

You can also use built-in Azure Data Studio (ADS) services to perform a variety of common management tasks on the big data cluster, including browsing HDFS, uploading files, previewing files, and creating directories, along with creating, opening, and running Jupyter-compatible Notebooks. A data virtualization wizard simplifies the creation of external data sources.

All the nodes in the cluster run collectd for OS and application performance data collection and fluentd for log collection. Logs are collected to ElasticSearch and monitoring data to InfluxDB. Data collected into ElasticSearch is exposed through Kibana for powerful log analytics and the monitoring data collected into InfluxDB is rendered in the provided dashboards in Grafana.

Security

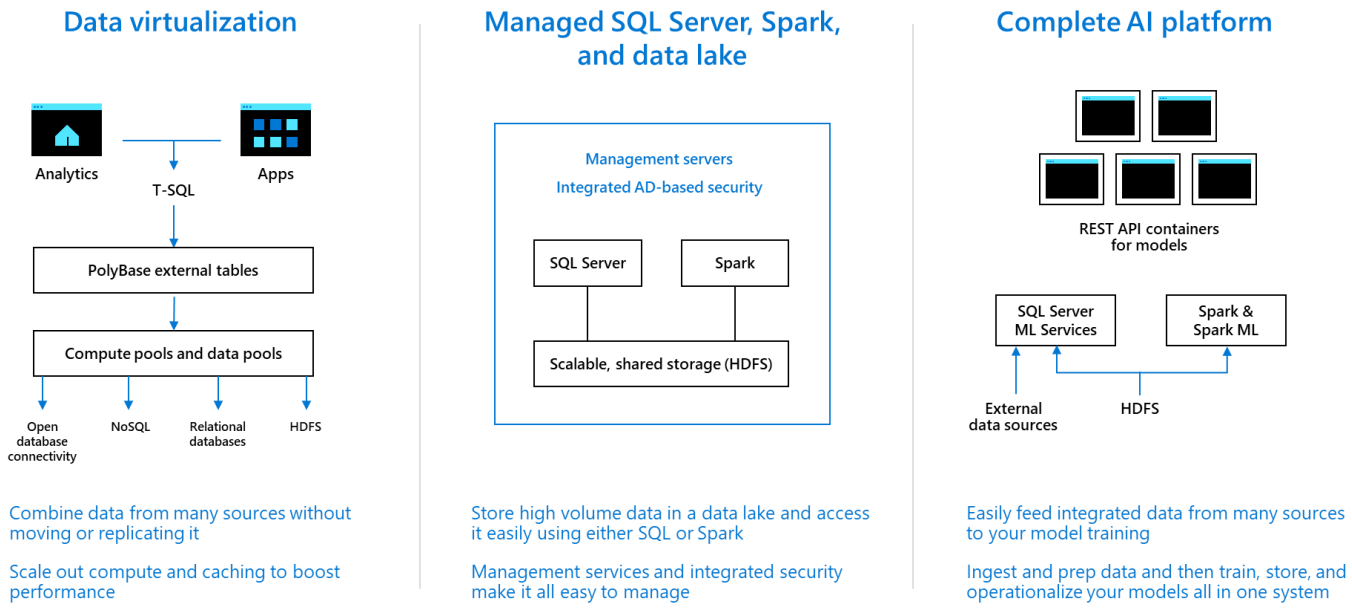
For big data cluster security, coherent and consistent authorization and authentication are the centerpieces. Big data clusters can be integrated with Active Directory through a fully automated deployment. Once a big data cluster is configured with Active Directory, you can leverage existing identities and user groups for unified access across all endpoints. In addition, once you have created external tables in SQL Server, you can control access to data sources using Active Directory users and groups to implement policies from a single location.

Throughout the cluster, integrated security between different components allow the original user's identity to be passed through when issuing queries from Spark and SQL Server, all the way to HDFS. The various external cluster endpoints support AD authentication. There are two levels of authorization checks in the cluster for managing data access. Authorization in the context of big data is done in SQL Server, using the traditional SQL Server permissions on objects and in HDFS with control lists (ACLs), which associate user identities with specific permissions. HDFS supports authorization by limiting access to service APIs, HDFS files, and job execution.

Tools like Azure Data studio offer a single sign-on experience. Encryption of communication between components is secured both internally and externally with TLS/SSL, using certificates. In addition to Active Directory, big data cluster security options also include authentication using basic username and passwords.

Conclusion

SQL Server Big Data Clusters offer a compelling new way to utilize SQL Server to bring high-value relational data and high-volume big data together on a unified, scalable data platform. Enterprises can leverage the power of PolyBase to virtualize their data stores, create data lakes, and build scalable data marts in a unified, secure environment without needing to implement slow, costly ETL pipelines. This makes data-driven applications and analysis more responsive and productive. SQL Server Big Data Clusters provide a complete AI platform to deliver the intelligent applications that help make any organization more successful.



Summary of SQL Server 2019 Big Data Clusters

Want to learn more?

For more information about SQL Server 2019, please visit https://aka.ms/ss19

For more information about Azure Data Studio, please visit https://aka.ms/azuredatastudio

For SQL Server 2019 Big Data Clusters documentation, please visit https://aka.ms/sqlbigdatacluster