Microsoft

Microsoft SQL Server 2017 on Linux

Technical white paper

Published: September 2017

Applies to: Microsoft SQL Server 2017 on Linux and Docker containers

Copyright

Contents

Summary

The IT landscape is constantly changing and increasingly diversifying—particularly operating systems, including Linux. SQL Server 2017 brings Microsoft's industry-leading relational database engine to the enterprise Linux ecosystem. Microsoft is committed to enabling customers to choose the best platform for their data and applications. This includes providing interoperability with open source solutions. For example, Linux distributions like Red Hat Enterprise Linux (RHEL), Ubuntu, and SUSE are becoming more popular for relational and nonrelational workloads. With SQL Server 2017, application developers can build their applications using any language, including Node.JS, .NET, Java, and PHP, and deploy their solutions on platforms such as Windows, Linux, and Docker containers—all in a private cloud (on-premises), Microsoft Azure, or third-party clouds.

Industry landscape and trends

The IT landscape is constantly changing, and increasingly diversifying—particularly in terms of heterogeneous operating systems (Linux, Windows, etc.). Additionally, IT organizations must contend with multiple data types, different development languages, and a mix of on-premises/cloud/hybrid environments. Maintaining increasingly complex environments is a daunting task, requiring people and processes that can keep the pace and somehow simultaneously reduce operational costs.

Microsoft is committed to enabling customers to choose the best platform for their data and applications. This includes providing interoperability with open source solutions. One out of three virtual machines on Azure is running Linux. Microsoft platforms such as HDInsight and R Server can be deployed on both Windows and Linux. This commitment extends into application development, including .NET core as an open source project and the newly announced Visual Studio Code extension for SQL Server. Similarly, SQL Server drivers and connectivity tools and APIs are available for nearly any environment, enabling any application to integrate with SQL Server, regardless of programming language or environment.

SQL Server 2017: The power of SQL Server now on Linux and Containers

SQL Server 2017 builds on the industry-leading[1] capabilities of SQL Server 2016, holding benchmarks in such areas as:

- **Performance**—SQL Server owns the top TPC-E performance benchmarks for transaction processing, the top TPC-H performance benchmarks for data warehousing, and the top performance benchmarks with leading business applications[2]. In April 2017, Hewlett Packard Enterprise published a new TPC-H 1TB world record with SQL Server 2017 running on Linux.[3]
- **Security**—According to the National Institute of Standards and Technology (NIST) public security board, SQL Server has the lowest number of reported security vulnerabilities across the major database vendors (NIST, 2016).
- **Total cost of ownership**—SQL Server has a significantly lower total cost of ownership (TCO) than similar enterprise data solutions. In some cases, the TCO for SQL Server 2016 was found to be as low as 1/12th the cost of comparable products/features.

SQL Server 2017 on LInux continues the evolution of SQL Server, bringing new capabilities to the modern data ecosystem to better support and enhance data management and data-driven applications. The following scenarios represent potential uses SQL Server 2017 on Linux, as well as the latest features.

SQL Server 2017 on Linux and Containers

SQL Server 2017 brings the industry-leading Microsoft relational database engine to the enterprise Linux ecosystem. This includes SQL Server Agent, Active Directory authentication, best-in-class high availability / disaster recovery, and unparalleled data security features. It's important to note that SQL Server on Linux is not a port or rewrite. This is the world-class Microsoft RDBMS now available on more operating systems—like Red Hat Enterprise Linux, SUSE Linux Enterprise Server, and Ubuntu—and more cloud and container platforms like Docker.

---

[1] Gartner has rated Microsoft as a leader with the most complete vision and highest ability to execute of any operational database management system for two consecutive years. *Gartner does not endorse any vendor, product or service depicted in its research publications, and does not advise technology users to select only those vendors with the highest ratings or other designation. Gartner research publications consist of the opinions of Gartner's research organization and should not be construed as statements of fact. Gartner disclaims all warranties, expressed or implied, with respect to this research, including any warranties of merchantability or fitness for a particular purpose*.

[2] Lenovo has announced the TPC-H 10 TB benchmark world record (Lenovo Server Performance Lab, 2016) using SQL Server 2016. In May 2016, Lenovo also published a new TPC-H 30 TB world record (Lenovo Server Performance Lab, 2016).

[3] www.tpc.org, "TPC-H Result Highlights HPE Proliant DL380 Gen9", March 2017 (link)

SQL Server 2017 offers the best performance and security features, and they're now available on all supported platforms, including Windows, Linux, and containers. High-performance features bring record-breaking speed to data-driven applications. These features include columnstore (which provides column-based data storage and processing to achieve up to 10 times the query performance and data compression of row-based storage) and in-memory OLTP (which brings transaction processing to memory-optimized tables at more than 2.5 times the speed of disk-based tables). Security features like Auditing, Transparent Data Encryption, Row-Level Security, Always Encrypted and Dynamic Data Masking provide server-side security measures that vastly simplify keeping data safe from unauthorized access, without the need to modify existing client applications. Auditing enables teams to monitor access and track potentially suspicious activity. Transparent Data Encryption protects data at rest at a file level, while Always Encrypted secures data in motion and at rest. With these capabilities available on all SQL Server editions, organizations can choose their deployment environment by operational need, as opposed to desired features.

SQL Server 2017 on Linux is not a rewrite or a port; SQL Server on Windows and on Linux share a common code base that access low-level operating system functions through a platform abstraction layer. While not all the scenarios and features covered by SQL Server on Windows are supported on Linux yet, SQL Server 2017 on Linux is ready to support transactional and data-warehousing workloads, as well as participating in Availability Groups. SQL Server Integration Services for extracting, transforming and loading (ETL) data from various data sources to a SQL Server data warehouse is now supported on Linux as well.. The majority of Database Engine workloads can be moved from Windows to Linux without modification. For more information about SQL Server features not currently supported on Linux, see the SQL Server on Linux release notes.

Native Linux experience with support for existing SQL Server tools

Microsoft has focused on providing a Linux-native user experience for SQL Server, starting with the installation process. SQL Server 2017 uses the standard package-based installation method for Linux using yum for Fedora-based distributions and apt-get for Debian-based distributions. Administrators can update SQL Server 2017 instances on Linux by using their existing package update/upgrade processes.

The SQL Server service runs natively using systemd, and performance can be monitored through the file system as expected. Linux file paths are supported in T-SQL statements and scripts to do things like defining/changing the location of data files or database backup files. High availability clustering can be managed with popular Linux high availability solutions like Pacemaker and Corosync.

Full-Text Search is now available for Linux. This feature enables you to run full-text queries against character-based data in SQL Server tables. Full-text queries perform linguistic searches against text data in full-text indexes by operating on words and phrases based on the rules of a language, such as English or Japanese. Full-text queries can include simple words and phrases or multiple forms of a word or phrase. A full-text query returns any documents that contain at least one match (also known as a *hit*). A match occurs when a target document contains all the terms specified in the full-text query and meets any other search conditions, like the distance between the matching terms. For more information on this feature, see SQL Server Full-Text Search on Linux.

Microsoft offers tools such as Data Migration Assistant to assist with moving existing workloads to SQL Server 2017.

Platform abstraction with containers

Containers are software-defined spaces with some similarities to virtual machines. Containers don't use hardware virtualization; instead, the host operating system is abstracted from the space, allowing a container to hold only the dependencies required for the application and the application itself. A Linux-based container can be deployed to any Linux machine (physical or virtual) running Docker and can be expected to run without changes to the host operating system.

With support for Windows and Linux containers, SQL Server can run in container orchestration solutions, such as Docker Swarm, Red Hat Open Shift, Mesosphere DC/OS, and Kubernetes. With the Management Pack for SQL Server on Linux, administrators can use System Center Operations Manager to monitor everything from the hardware up to the database engine instances and individual databases.

Similarly, Docker CE for Mac enables developers to run Linux containers on macOS. SQL Server also supports installation on Windows containers. One of the key benefits of using containers in the development process is the ability to work in various environments. With containers, development teams can work in dev/test environments that are functionally identical to production environments. In addition, container orchestration solutions can manage the deployment of containers automatically, aiding the automation of testing and deployment.

Enabling DevOps practices with containers

DevOps is about bringing great applications to customers through people, processes, and tools. Taking a lean approach to product development (for example, splitting work into small batches and implementing customer feedback) predicts higher IT performance and less deployment pain.[4] Compared to the challenges of a traditional development approach, or even an Agile approach, DevOps is gaining ground as a best practice for delivering high-performance solutions to market. Traditional hardware and VM installations of SQL Server have had a difficult time fitting into this framework. With SQL Server now available on containers, several DevOps practices are accessible to data-driven application development including the ability to build a container image that can be used in any environment.

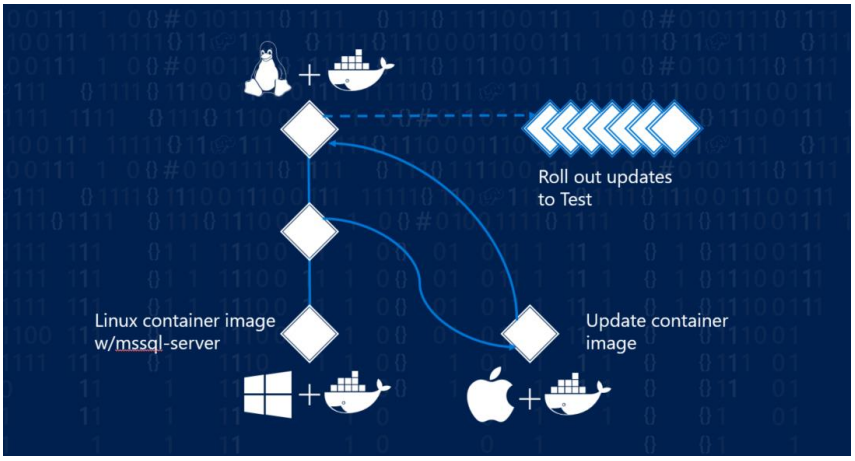
Dev/Test

Dev/Test is based on the principle of working in identical environments for development, test, and production. This minimizes the risk of a configuration or dependency-based issue manifesting after deployment to production. With containers, application dependencies are packaged within the container. Containers can operate on any host OS that is capable of hosting containers, such as Red Hat Enterprise Linux or Windows Server 2016. This also means that for container-based solutions, the host OS does not need to be the same as the production host OS.

For example, a development team may be working on a development environment, which could be Linux or Mac with Docker running the container locally, including the SQL Server database engine and the databases to support the application. The test environment could be a cluster of Linux hosts hosting hundreds of instances running test automation. The production environment would be effectively identical with containers orchestrated across the private/public/hybrid cloud infrastructure. The SQL Server database configuration, database schema, and pre-

[4] Puppet and Dora, 2016 State of DevOps Report (link).

populated data, as a part of the container image, are replicated with each container instance, eliminating configuration differences from the list of potential causes for a bug.



Continuous integration, continuous delivery

Continuous integration and continuous delivery (CI/CD) are DevOps practices that, by continually integrating updates and fixes, and releasing on the order of minutes instead of days, bring products to market faster, with better quality, reliability, efficiency, and customer satisfaction. With SQL Server 2017 on containers, container orchestrators such as Docker Swarm or Red Hat Open Shift can quickly propagate container updates to test automation environments, and eventually to production.

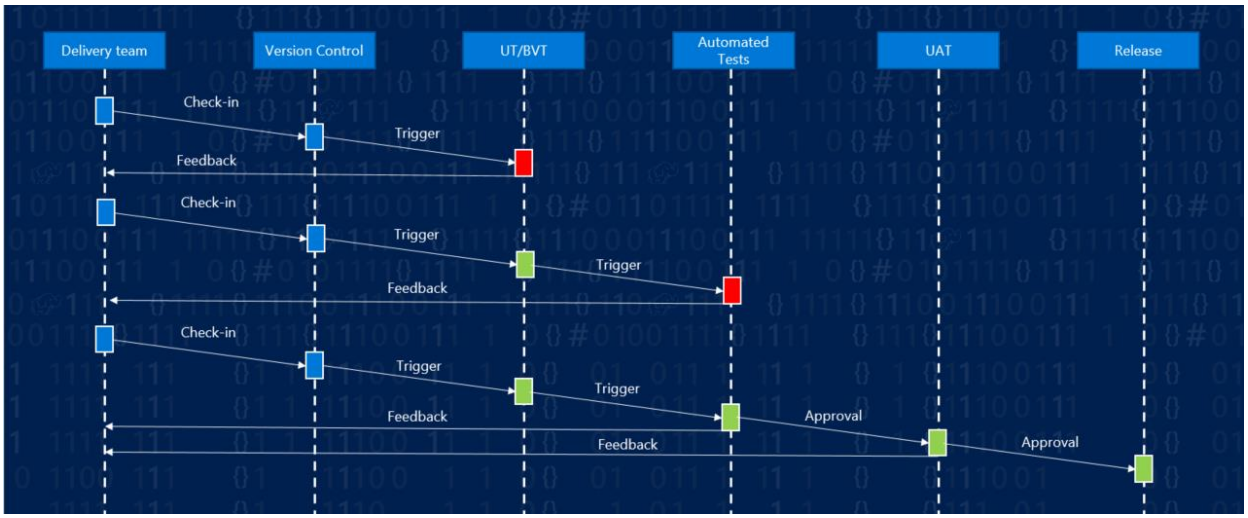


Figure: CI/CD sequence diagram

CI/CD is a departure from traditional software releases, enabling organizations to release daily or even multiple times a day. This requires a high degree of automation, triggering deployments to environments when changes are committed, executing tests, and moving on to the next level of deployment. The benefits outweigh the costs, enabling teams to deliver higher-quality products more reliably—with far greater efficiency and speed to market—leading to greater customer satisfaction as new content and features roll out more frequently.

SQL Server 2017 on containers aligns well with DevOps principles and enables several key practices that play a

pivotal role in bringing mission-critical, intelligent apps to production quickly, with high cost efficiency, high quality, and high customer satisfaction. From environment parity with Dev/Test to high-velocity deployments through CI/CD, SQL Server 2017 on Linux and containers make a capable addition to any organization's DevOps toolbox.

New SQL Server 2017 Features available on Linux

SQL Server 2017, with its support for Linux and Docker, is the release that truly makes SQL Server a platform with choice: choice of development languages, data types, on-premises or cloud, choice of operating systems, all with support for big data and advanced analytics. For a full list of new features and enhancements in specific feature areas, see What's new in SQL Server 2017.

Adaptive Query Processing feature family

New in SQL Server 2017, Adaptive Query Processing introduces new capabilities that allow the SQL Server query processor to modify plan choices based on runtime characteristics.

During query processing and optimization, the cardinality estimation (CE) process is responsible for estimating the number of rows processed at each step in an execution plan. Inaccurate estimates can cause slow query response time, excessive resource utilization (CPU, Memory, IO), and reduced throughput and concurrency. To improve CE techniques, SQL Server 2017 introduces a new feature family: adaptive query processing (AQP). AQP improves the handling of the more intractable CE issues. Features included in the AQP feature family are:

Interleaved Execution: Materializes problematic estimates of multi-statement table valued functions (MSTVF) that propagate to downstream operations, correcting the inaccurate estimates and enabling the Query Optimizer to revise plan choices based on accurate estimates. The first version of Interleaved Execution addresses cardinality estimates for MSTVFs.

Batch mode adaptive join: Enables the choice of a hash join or nested loop join method against a columnstore table to be deferred until after the first join input has been scanned. Adaptive join evaluates the input and executes the most efficient of the two join algorithms at execution time.

Batch mode memory grant feedback: Tracks the actual memory required for a query, and when an identical query statement is called, enables a more accurate memory grant size. This avoids excessive memory grants, which can reduce concurrency, as well as under-estimated memory grants, which can cause expensive spills to disk.

Automatic plan correction

New features in SQL Server 2017 detect plan choice regressions and give recommendations on how to fix the problem. These automatic plan correction features help to maintain the performance of data queries, even when application changes occur.

Automatic tuning: This database feature provides insight into potential query performance problems, recommends solutions, and automatically fixes identified problems.

Forcing last good plan: To prevent unexpected performance issues, users must periodically monitor the system and look for queries that regressed. If a plan has regressed, it's a good idea to find a previous good plan and force it, instead of using the current one. With this feature, you can monitor the performance of an executed query using the forced plan and verify that the plan works as expected.

Automatic regression detection: The Database Engine detects potential plan choice regressions and shows recommended actions to be applied in the sys.dm_db_tuning_recommendations view. This view shows information about the problem, the importance of the issue, and details such as the identified query, the ID of the regressed plan, the ID of the plan used as the baseline for comparison, and the Transact-SQL statement that can be executed to fix the problem.

Automatic plan tuning: The Database Engine can automatically switch to the last known good plan whenever a regression is detected.

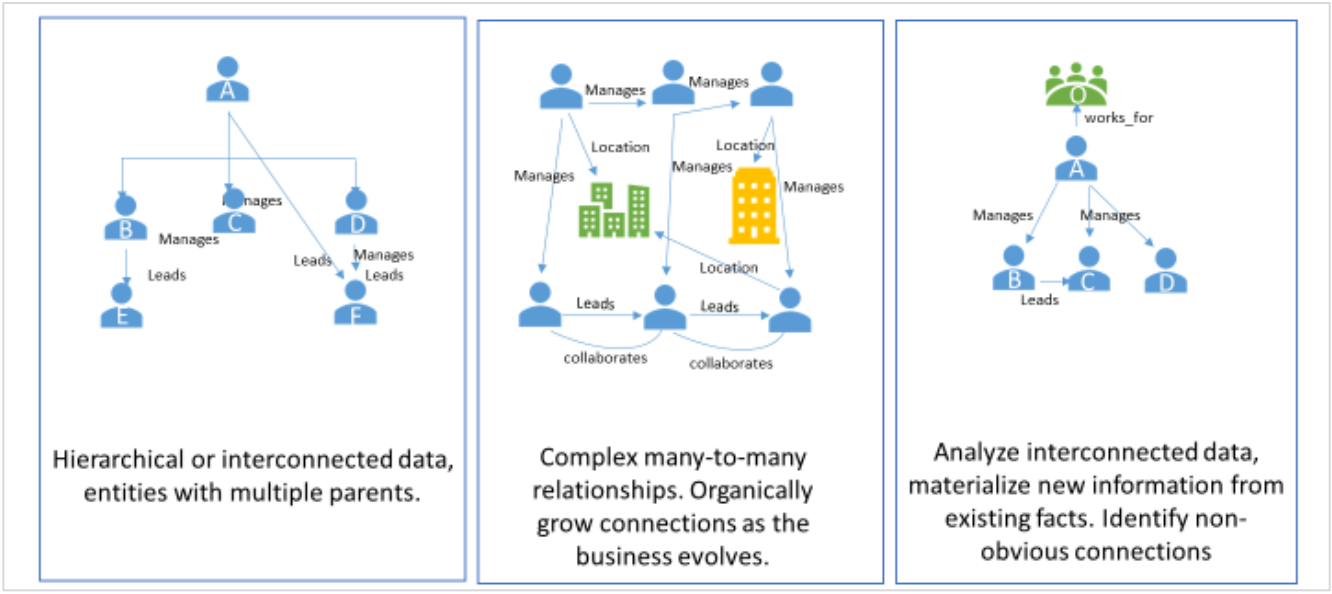
Resumable Online Index Rebuild

Planning, maintaining, and managing a large index online can be challenging. The bigger indexes are, the more difficult index maintenance becomes. Reorganizing and rebuilding indexes can be especially cumbersome. Resumable Online Index Rebuild in SQL Server 2017 enables continuation of an online index rebuild operation after a failure such as a failover to a replica or insufficient disk space. Resumable Online Index Rebuild also enables pausing and later resuming an online index rebuild operation. For example, you might need to temporarily free up system resources to execute a high-priority task, or complete the index rebuild at another time if the available maintenance window is too short for a large table. Finally, Resumable Online Index Rebuild does not require significant log space, which allows you to perform log truncation while the resumable rebuild operation is running.

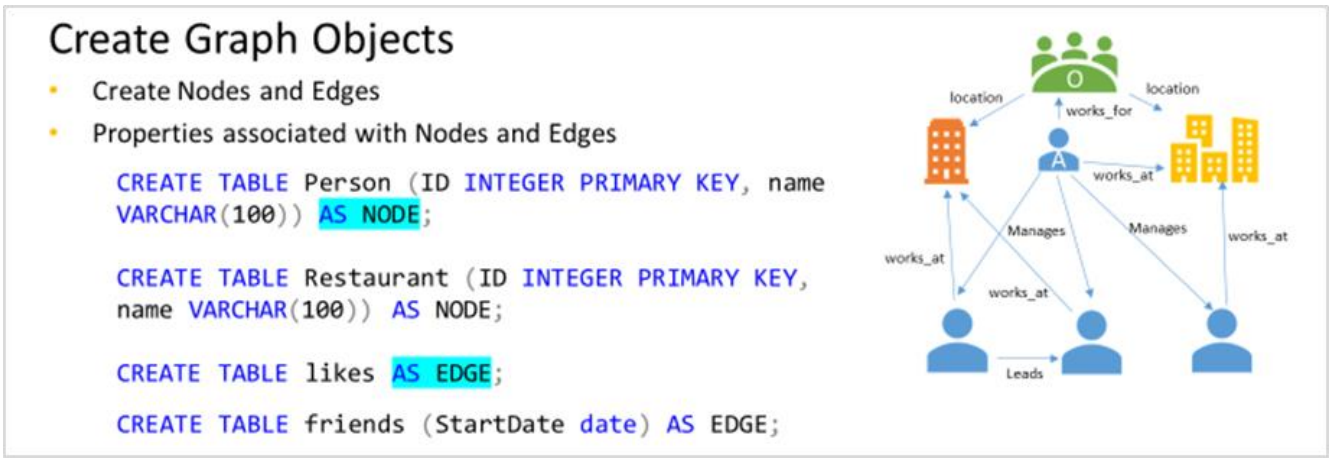
Non-resumable online index maintenance operations have been a feature of SQL Server Enterprise Edition since SQL Server 2005.

Graph processing with SQL Server 2017

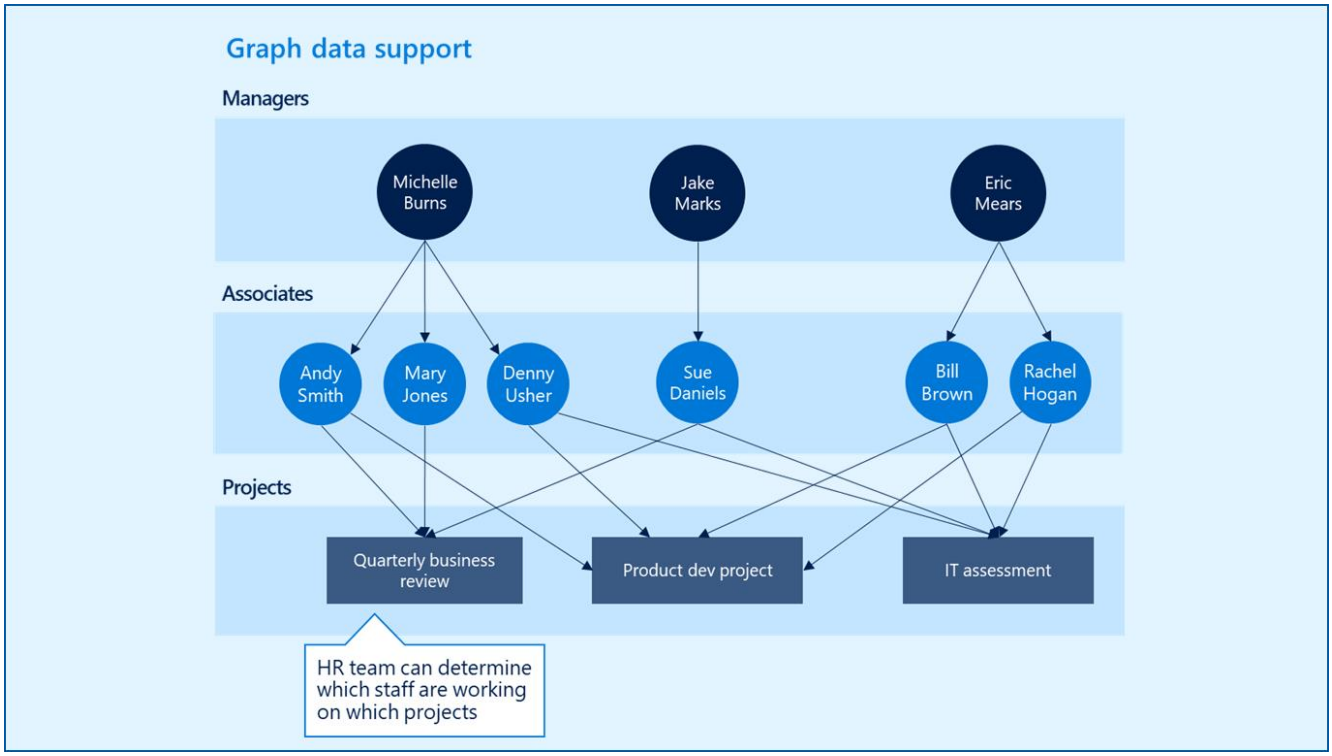
Customers need to do more than just manage large volumes of data. They also need to analyze their existing data more effectively to understand its relationships and patterns. Querying data from a relational schema by using traditional SQL queries can be a complex task. SQL Server 2017 introduces SQL Graph to make modelling and analyzing relationships easier by allowing users to handle the relationships in a more flexible and agile way.



A graph database is a set of nodes (or vertexes) and edges (or relationships). A graph database is useful for representing data that includes many—often complex—relationships. SQL Graph in SQL Server 2017 brings graph processing capabilities to SQL Server, enabling users to link different pieces of connected data to help them gather powerful insights and increase operational agility. This is well suited for applications in which relationships are important, such as fraud detection, risk management, social networks, recommendation engines, predictive analysis, and IoT suites.



CRUD operations with SQL Graph effectively create nodes to represent various entities and create edges to represent relationships between any two nodes. Both nodes and edges can have properties associated with them. In addition, SQL Graph can perform multi-hop navigation in a graph using pattern matching (no joins). SQL language extensions for graph support enable join-free, pattern-matching queries for multi-hop navigation.



Mission-critical high availability on Linux

SQL Server high availability solutions provide mission-critical uptime, fast failover, improved manageability, and better use of hardware resources.

Instance-level high availability with Always On Failover Cluster Instances

An Always On Failover Cluster Instance (FCI) provides instance-level redundancy, allowing a SQL Server instance to remain available during planned and unplanned operating system outages caused by hardware failure, software failure, or system maintenance. FCIs are supported on Windows and Linux.

An FCI comprises two or more cluster nodes with access to cluster shared storage (either SAN or direct-attached storage). Only one node is active at a time—secondary node(s) are available but passive, ready to assume the role of active node during failover. An FCI requires a cluster manager to marshal cluster resources. On Linux, the supported cluster manager is Pacemaker. For more information about the capabilities of this feature, see Always On Failover Cluster Instances (SQL Server).

Database-level high availability with Always On Availability Groups

An availability group supports a replicated environment for a discrete set of user databases, known as availability databases. You can create an availability group for high availability (HA) or for read-scale. An HA availability group

is a group of databases that fail over together. A read-scale availability group is a group of databases that are copied to other instances of SQL Server for read-only workload. An availability group supports one set of primary databases and one to eight sets of corresponding secondary databases. Always On Availability Groups offers the same level of high availability and disaster recover as Oracle Real Application Clusters but does so on fewer servers, and is included in the core SQL Server license cost.

SQL Server 2017 introduces the following enhanced features focused on ensuring high availability while running mission-critical workloads.

Flexibility for HA architectures

SQL Server 2017 supports two different architectures for availability groups: Always On and Read-Scale.

Always On Availability Groups provide high availability, disaster recovery, and read-scale balancing. These availability groups require Pacemaker for Linux clusters.

Read-Scale Availability Groups provide read-only workload replicas but not high availability. With this architecture, there is no need for a cluster manager. The benefit of this is the ability to have secondary replicas in mixed-OS environments. Read-scale availability groups are a new feature in SQL Server 2017.

Always On Availability Groups for Linux

Always On Availability Groups have been added to the Linux edition, enabling customers to test the database software's hardiness while running critical workloads. This feature is now available on all Linux OS distributions supported by SQL Server 2017—Red Hat Enterprise Linux, Ubuntu, and SUSE Linux Enterprise Server. All the capabilities that make availability groups a flexible, integrated, and efficient HADR solution are available on Linux, including multi-database failover, multiple synchronous and asynchronous secondaries, manual or automatic failover, active secondaries for read and backup workloads, and more. For more information about the capabilities of this feature, see Always On Availability Groups for SQL Server on Linux.

Member instances of a single Distributed Always On Availability Group are permitted to run on Windows, Linux, or a mixture of both operating systems. This enables organizations planning to migrate their SQL Servers to Linux to easily test workloads and applications before switch-over.

Load balancing of readable secondary replicas

Secondary replicas support read-only access to all the secondary databases. Typically, these replicas are in sync with the primary replica, including full-text indexes and durable memory-optimized tables. This means that secondary replicas can provide services somewhat like a data mart, giving read-only access to production data with little latency. Routing read-only requests can also be load balanced on the availability group listener, giving organizations control over how to route read-only workloads to their secondary replicas. For more information, see Configure Read-Only Routing for an Availability Group.

Performance and security features available on SQL Server 2017 on Linux

SQL Server 2017 provides a consistent programmability surface area for developers and organizations across SQL Server editions. This enables customers and partners to build advanced applications that scale across editions and the cloud as they grow. Developers and application partners can build to a single programming surface area when creating or upgrading intelligent applications and use the edition that scales to each application's needs. This includes features like in-memory OLTP, in-memory Columnstore, Real-Time Operational Analytics, compression, and partitioning. Security features like Always Encrypted, Row-Level Security, Dynamic Data Masking, Auditing, and Transparent Data Encryption are available on Linux as well. The following features are just a subset of all the features and capabilities available on the Linux version of SQL Server 2017. SQL Server Technical Documentation is a great resource for learning more about all the features, capabilities and services available on SQL Server.

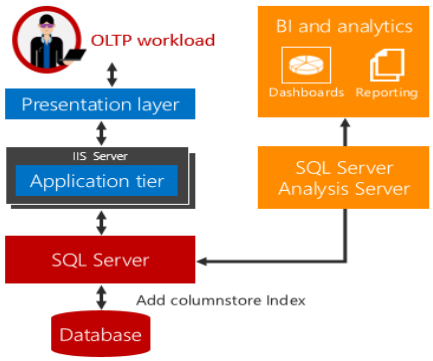
In-Memory OLTP

In-Memory OLTP is a memory-optimized database engine integrated into the SQL Server engine, optimized for transaction processing. In-Memory OLTP can significantly improve OLTP database application performance. It improves throughput and reduces latency for transaction processing, and can help improve performance of transient data scenarios such as temp tables and ETL.

For more information, see In-Memory OLTP (In-Memory Optimization).

Columnstore

SQL Server 2016 introduced real-time operational analytics—the ability to run both analytics and OLTP workloads on the same database tables at the same time. Besides running analytics in real time, you can also eliminate the need for ETL and a data warehouse for operational workloads.



Traditionally, businesses have had separate systems for operational and analytics workloads. For such systems, Extract, Transform, and Load (ETL) jobs regularly move the data from the operational store to an analytics store. The analytics data is usually stored in a data warehouse or data mart dedicated to running analytics queries. While this solution has been the standard, it has these three key challenges:

Complexity. Implementing ETL can require considerable coding, especially to load only the modified rows. It can be complex to identify which rows have been modified.

Cost. Implementing ETL requires purchasing additional hardware and software licenses.

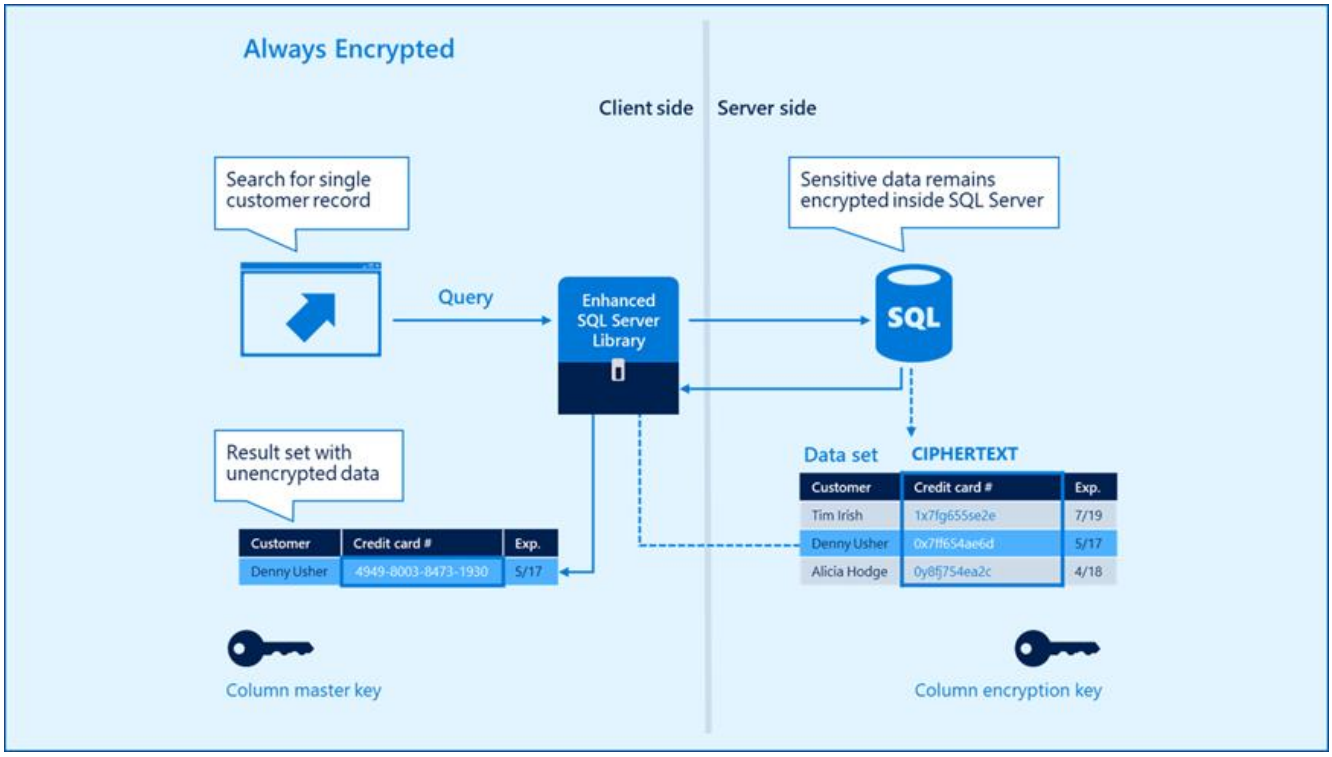
Data Latency. Implementing ETL adds a time delay for running analytics. For example, if the ETL job runs at the at end of each business day, the analytics queries will run on data that is at least a day old. For many businesses, this delay is unacceptable because the business depends on analyzing data in real time. For example, fraud detection requires real-time analytics on operational data.

Real-time operational analytics offers a solution to these challenges. There is no time delay when analytics and OLTP workloads run on the same underlying table. For scenarios that can use real-time analytics, the costs and complexity are greatly reduced by eliminating the need for ETL and the need to purchase and maintain a separate data warehouse.

For more information, see Columnstore Indexes Guide.

Always Encrypted

Always Encrypted is a feature designed to protect sensitive data, such as credit card numbers or national identification numbers (like US social security numbers), stored in Azure SQL Database or SQL Server databases. Always Encrypted allows customers to encrypt sensitive data inside their applications and never reveal the encryption keys to the Database Engine, whether SQL Database or SQL Server. As a result, Always Encrypted provides separation between those who own the data and those who can view it, and those who manage the data but should have no access. By ensuring that on-premises database administrators, cloud database operators, and other highly privileged but unauthorized users can't access the encrypted data, Always Encrypted enables customers to confidently store sensitive data outside of their direct control. This allows organizations to encrypt data at rest and in use for storage in Azure, to enable delegation of on-premises database administration to third parties, and to reduce security clearance requirements for their own DBA staff. For more information, see Always Encrypted.



Transparent Data Encryption

Transparent Data Encryption (TDE) encrypts SQL Server, Azure SQL Database, and Azure SQL Data Warehouse data files. This is known as encrypting data at rest. Organizations can take several precautions to help secure the database, such as designing a secure system, encrypting confidential assets, and building a firewall around the database servers. However, in a scenario where the physical media such as drives or backup tapes is stolen, a malicious party can restore or attach the database and browse the data. One solution is to encrypt the sensitive data in the database and protect the keys that are used to encrypt the data with a certificate. This prevents anyone without the keys from using the data, but this protection must be planned in advance.

TDE performs real-time I/O encryption and decryption of the data and log files. The encryption uses a database encryption key (DEK), which is stored in the database boot record for availability during recovery. The DEK is a symmetric key secured by using a certificate stored in the master database of the server or an asymmetric key protected by an Encrytion Key Management module. For more information, see Transparent Data Encryption (TDE).

Row-Level Security

Row-Level Security (RLS) enables customers to control access to rows in a database table based on the characteristics of the user executing a query (for example, group membership or execution context).

Row-Level Security simplifies the design and coding of security in an application. Row-Level Security also enables organizations to implement restrictions on data row access. For example, an organization can ensure that employees can access only those data rows that are pertinent to their department, or restrict a customer's data access to only the data relevant to their company.

The access-restriction logic is located in the database tier rather than separate from the data in another application tier. The database system applies the access restrictions every time data access is attempted from any tier. This makes the security system more reliable and robust by reducing its surface area. For more information, see Row-Level Security.

Dynamic data masking

Dynamic data masking (DDM) limits sensitive data exposure by masking it to nonprivileged users. It can be used to greatly simplify the design and coding of security in an application.

Dynamic data masking helps prevent unauthorized access to sensitive data by enabling organizations to designate how much of the sensitive data to reveal, and it has minimal impact on the application layer. Dynamic data masking can be configured on the database to hide sensitive data in the result sets of queries over designated database fields without changing the data in the database. Dynamic data masking is easy to use with existing applications because masking rules are applied in the query results. In many applications, sensitive data can be masked without changes to existing queries. For more information, see Dynamic Data Masking.

Auditing

SQL Server audit allows customers to track and log events that take place on an instance of the Database Engine, or on an individual database. Server audits can contain server audit specifications for server level events, and database audits can contain specifications for database level events. Audited events can be written to the event logs or to audit files.

There are several levels of auditing for SQL Server, depending on government or standards requirements for your installation. SQL Server Audit provides the tools and processes you must have to enable, store, and view audits on various server and database objects. For more information, see SQL Server Audit.

Getting Started with SQL Server 2017 on Linux

SQL Server 2017 is supported on Red Hat Enterprise Linux (RHEL), SUSE Linux Enterprise Server (SLES), and Ubuntu. It is also available as a Docker image, which can run on Docker Engine on Linux or Docker for Windows/Mac. The following example is for Red Hat Enterprise Linux. For full instructions on how to install SQL Server 2017 and the tools on supported Linux distributions and containers, see Installation guidance for SQL Server on Linux.

Installation of SQL Server 2017 is quick and easy—SQL Server is installed using the package manager on your OS. To get started, you will need to download the repository configuration file. You also need to have a user who has **sudo** access.

```
sudo curl -o /etc/yum.repos.d/mssql-server.repo
https://packages.microsoft.com/config/rhel/7/mssql-server.repo
```

Run the following commands to install SQL Server:

```
sudo yum update
sudo yum install -y mssql-server
```

After the package installation finishes, run **mssql-conf setup** and follow the prompts to set the system administrator (SA) password and choose your edition.

```
sudo /opt/mssql/bin/mssql-conf setup
```

Conclusion

SQL Server 2017 is a breakthrough for SQL Server, bringing the industry-leading performance and security capabilities of SQL Server to Linux and Docker containers. New features like Adaptive Query Processing, and Automatic Plan Correction enable organizations to further optimize their data processing capabilities, while SQL Graph brings the ability to map and query relationships in a graph structure rather than using a traditional relational schema. With SQL Server 2017, organizations have a full range of options for building or extending their data ecosystems across operating systems and programming languages.

Calls to action

Get more information about SQL Server: https://docs.microsoft.com/en-us/sql/sql-hub-menu

Download a trial at https://www.microsoft.com/en-us/sql-server/sql-server-downloads

Feedback

Did this paper help you? Please give us your feedback by telling us on a scale of 1 (poor) to 5 (excellent) how you rate this paper and why you have given it this rating. More specifically:

- Are you rating it highly because of relevant examples, helpful screen shots, clear writing, or another reason?
- Are you rating it poorly because of examples that don't apply to your concerns, fuzzy screen shots, or unclear writing?

This feedback will help us improve the quality of white papers we release.

Please send your feedback to: sqlsrvwpfeedback@microsoft.com