Ovum
TMT intelligence | informa

Microsoft SQL Server 2017: Multiplatform becomes reality

Linux support and data science headline the new release

Publication Date: 28 Feb 2018 | Product code: INT002-000080

Tony Baer

Summary

Catalyst

The October release of Microsoft SQL Server 2017 is the latest proof positive that this is not your father's Windows company anymore. The announcement, which included a native Linux version, was hardly a secret, as Microsoft disclosed its plans nearly two years ago. Beneath the headlines, the fact that with the new release, the code bases of SQL Server and its cloud counterparts (Azure SQL Database and Azure SQL Data Warehouse) have merged means that new features and refreshes of SQL Server will occur at a much faster pace in the future. The new release also takes the next step with in-database machine learning and advanced analytics by putting Python on the same footing as R.

Ovum view

The headlines of a Linux version of SQL Server mark a milestone for the product. It places SQL Server in the database mainstream, as Linux is where most of the competing open source and enterprise platforms lie. Now the decision over adopting SQL Server is no longer Windows-dependent. But beyond the headlines are several other developments that make this release significant. First and foremost is convergence of the code base with Azure SQL Database; while the two platforms will never become identical (as one is specifically engineered for cloud), it means more common features, a cloud-first roadmap for introducing new capabilities, and more frequent product refreshes of SQL Server. Code base convergence provides Microsoft a strategic competitive advantage vs. its pure cloud counterparts, Amazon Aurora and Google Cloud SQL, thanks to a stronger hybrid and the opportunity to support a "lift and shift plus" strategy that provides an easier migration path to the flexibility, economics, and scale of cloud database deployment. Other enhancements – including more data science support and a new graph table feature – place SQL Server in the mainstream of databases going multimodel.

Key messages

- SQL Server 2017 delivers on the promise of multiplatform support with full native support of Linux.
- Merging of the SQL Server code base with that of Azure SQL Database promises more frequent product refreshes.
- New machine-learning in-database processing capabilities centering on Python and T-SQL features follow in the footsteps of in-database R processing delivered with the previous 2016 release.
- While Microsoft is not blazing new ground with graph extensions to SQL Server 2017, its table-oriented *database* approach is far more ambitious than many of its rivals that are supporting graph *engines.*

Recommendations

Recommendations for enterprises

Addition of Linux support is no surprise, as Microsoft disclosed long-term plans just months *before* releasing the predecessor, SQL Server 2016. But what is notable is that the Linux release is not just a virtual machine or container but a fully native implementation that supports the core features of the platform (although there are some delays with several features). So, a hurdle to SQL Server adoption (the requirements for Windows) is eliminated; SQL Server now runs on the same platforms as the bulk of its enterprise rivals (e.g. Oracle, IBM, open source databases, and cloud-native databases) – OS is no longer a decision factor.

But there's another development with this release that in the long run may be even more significant, especially if your organization is considering cloud deployment. With this release, Microsoft has merged the code bases of SQL Server and Microsoft's cloud-native counterpart, Azure SQL Database. While there is not yet 100% feature parity, that is just a matter of time (excluding capabilities that are cloud-specific, such as database automation or multi-data center high availability). The merging of code bases means that Microsoft will adopt a strategy that Ovum expects will become customary for enterprise databases – to become a "cloud-first" provider where new features debut in the cloud before they are packaged for release for on-premises editions (Oracle has already adopted this policy with Oracle database).

For enterprises, converging SQL Server and Azure SQL Database means that there is the opportunity for what we term a "lift and shift plus" scenario where you can migrate transparently to the cloud and then gradually leverage the transformational features of cloud deployment as your organization is ready, without migrating databases. Code convergence has allowed Microsoft to introduce Azure SQL Database Managed Instance, which provides a bare metal node (not multitenanted) service that emulates SQL Server (the primary difference being that the underlying file system is different).

Going forward, competition for enterprise databases will center on supporting a variety of deployment and use case scenarios, from on premise and cloud to small and globally scaled deployment; database automation; and support for advanced analytics. SQL Server 2017 builds on advanced analytic support from the previous edition (where R language processing was brought inside the database) with adding Python in-database support and new PREDICT functions for T-SQL (which means that some forms of predictive analytics come within reach of the core SQL skills base). SQL Server 2017 also adds to the multimodel story begun with JSON support in its predecessor with graph database computing, meaning you won't have to buy a specialized database or third-party engine to tackle the growing array of graph analysis use cases. While Microsoft is not the only database provider blazing this path, SQL Server is certainly in the mainstream.

Recommendations for vendors

Anticipated for nearly 18 months before its September 2017 release, Microsoft has transformed SQL Server into a multiplatform database. In the past, this would have been interpreted as a play for stealing share (and accounts) from Oracle. But the universe has significantly widened thanks to the emergence of open source databases and cloud-native platforms (e.g. Amazon Aurora, Google Cloud SQL). There is a common thread here: most open source databases and cloud-native databases run on Linux. And so the extension of SQL Server to Linux must be viewed in conjunction with another

development with this release that may be potentially even more significant in the long run: the merging of the SQL Server code base with Azure SQL Database, Microsoft's cloud-native relational database offering. As enterprises increasingly place business-critical systems in the cloud, providers that have a good hybrid "lift and shift plus" story (common platforms that run in the data center *and* natively in the cloud, which can smooth the path to transformation) will have competitive advantage vs. Amazon and Google. Among enterprise databases, Oracle is the first to make that claim; with the merging of the code bases, Microsoft has a chance to seize the opportunity as well.

Scope

Starting with Access, and then SQL Server, Microsoft has considerably broadened its portfolio of data platforms over the years. On the relational side, the flagship product SQL Server has been joined with a massively parallel processing (MPP) appliance branded the Analytics Platform System (APS). Likewise, in the cloud, there are several relational database SKUs including Azure SQL Database and Azure SQL Data Warehouse. While this report spotlights SQL Server 2017, we also are comparing notes with the latest release of Azure SQL Database, because in the current generation the code bases for both platforms have merged; comparing features provides useful insight on near-future roadmaps. Later in 2018, Ovum will provide a deep dive on all the Azure data platforms.

The big picture

Table 1. Evolution of SQL Server

Category	Feature	When Introduced
Platform	Multiplatform: Linux & Docker support	2017
	Temporal tables	2016
	JSON support	2016
	Graph database support	2017
Performance	In-memory OLTP	2014
	In-memory column store	2012
	Buffer pool extension to SSD	2012
	Adaptive Query Processing	2017
BI & Analytics	SQL Server Reporting Services (SSRS)	2004 (as enhancement to SQL Server 2000)
	SQL Server Integration Services (SSIS)	2012
	PolyBase (federated T-SQL across Hadoop) integrated into database	2016
	Tabular BI semantic model	2008
	Master Data Services	2008
	Data Quality Services	2012
	In-database advanced analytics	2016
	End-to-end mobile BI on any device	2016
Cloud readiness	Backup to Azure	2012
	Disaster recovery to Azure	2014
	Optimized VM (virtual machine) images in Azure gallery	2012
	Stretch Database	2016
Availability	Always-on	2012

	Basic availability groups	2016
Security	Transparent data encryption (TDE)	2008
	Backup encryption support	2014
	Encryption at rest and in motion	2016
	Dynamic data masking & row-level security	2016
	Separation of duties	2012

Source: Microsoft

SQL Server's evolution

Tracing the evolution of SQL Server over the past decade, several patterns emerge. Early on, the focus was on incorporating capabilities into the database that would otherwise require third-party tools, such as the master and data quality services that were introduced in the 2008 and 2012 releases, respectively. SQL Server 2012 introduced support for advanced storage with in-memory column store and SSD buffers; it also introduced always-on high availability and failover features, plus backup to the Azure cloud. The 2014 release built on the 2012 enhancements by extending in-memory processing to OLTP, and supporting Azure in disaster recovery mode. SQL Server 2016 introduced heterogeneity in data types (JSON support) and deepened support for analytics by supporting in-database R processing and integrating PolyBase federated query (to Hadoop) into the core platform. SQL Server 2016 also introduced Stretch Database, which provides a way to extend read-only access for selective tables from on-premises to the Azure cloud; while it provided a form of data tiering solution that could take advantage of less expensive cloud storage, we believe that Microsoft's next step will be to refine the idea further (or replace it) to take advantage of elastic compute.

Multiplatform is the headline, but cloud convergence is the sleeper

For SQL Server 2017, the obvious headline is multiplatform support. But behind the news is a related development that in the long run may even be more significant: the merging of the code base with Azure SQL Database. In the long run, this will bring the following scenarios:

- Packaging – SQL Server and Azure SQL Database will be different editions, with varied mixes of features, based on a common engine. The differences will be deployment related, as some cloud-oriented features such as global, multi-data center scaling, automated backup, automated patching, and some auto-tuning features may not be as practical in on-premises data centers that Microsoft does not control.
- Release cycles – SQL Server and Azure SQL Database will have aligned roadmaps, but the release cadence will differ. We expect Microsoft (like Oracle) will go cloud-first, with most new features initially surfacing in Azure SQL Database. While Azure SQL Database releases will

be more frequent, having the common code base means that new features will also make it into SQL Server more quickly. Going forward, we expect SQL Server release cycles will go on 3- or 6-month schedules.

- Differentiation – A “lift and shift plus” hybrid story that differentiates from cloud-native databases Amazon Aurora and Google Cloud SQL. Microsoft (along with Oracle) promotes transparency of the ability to run the same database in your data center and in the cloud, with the difference that the cloud version is, in effect, a super set that allows you to take advantage of cloud-oriented capabilities (e.g. automation, replication, scaling, elasticity). Azure SQL Database Managed Instance (discussed below), just announced, will be part of this story. Conversely, cloud-only platforms differentiate with a “revolutionary” architecture theme. For instance, Amazon Aurora promotes its unique approaches to ACID (using reliance on change logs rather than database pages), performance optimization (via multi-threading), automated replication, and distributed architecture (using sharding).

Convergence of general-purpose relational databases does not do away with Microsoft’s specialized analytic SQL platforms, including Advanced Parallel Server and Azure SQL Data Warehouse. While those platforms remain in the portfolio (we expect similar cloud/on-premises convergence there as well), the intent for SQL Server and Azure SQL Database is an 80/20 approach, where most analytics use cases should be accommodated on the flagship platform.

In reviewing new features, we note which ones are available in SQL Server 2017, Azure SQL Database, or both platforms.

Multimodel extensibility and advanced analytics

Other key enhancements center around extensibility (with graph database support), broader support for data science and machine learning (with in-database Python processing and new T-SQL PREDICT functions), along with machine-learning enhanced operational enhancements to database tuning (e.g. query and index optimizations).

SQL Server goes multiplatform

At long last, Linux

The headline of the SQL Server 2017 release is full support for Linux. There was little surprise, as Microsoft tipped its cards nearly two years ago, and a few months prior to release, announced the makeup of the release candidate.

As we noted in our report from the 2017 Microsoft Ignite user conference, the Linux edition of SQL Server is a fully native implementation, rather than a Windows virtual machine image that runs in Linux. And while this might sound backward to Windows developers, SQL Server on Linux supports the command line interface that Linux geeks prefer. It runs with what Microsoft terms the Platform Abstraction Layer (PAL), which abstracts the core database engine from the operating system. While it requires a few Windows binaries at startup, after that the boot up is fully in Linux.

Significantly, PAL was not suddenly created for the 2017 release; as far back as SQL Server 2005, Microsoft created SQL OS, an internal project that was the forerunner of PAL, so that the database would not have to rely on Windows to manage memory.

Besides Windows, SQL Server 2017 also runs on all the major commercial versions of Linux: Red Hat Enterprise Linux (RHEL), SUSE, and Ubuntu. The finished product looks and feels like Linux. While the existing Windows-based SQL Server tools portfolio can connect to both the Windows and Linux editions, the Linux version also supports the command line interface to which Linux developers are accustomed.

Feature comparison between Linux and Windows editions

The game plan is that the Linux edition is to be a full-fledged offering. The initial release has almost all the *core* features of SQL Server database, but as shown in Table 2, there are still some blanks to fill.

Table 2. Sampling of current and roadmap features for SQL Server 2017, Linux edition

Supported in initial release	On the roadmap
JSON and XML support	SQL Server Reporting Services (SSRS) and SQL Server Analysis Services (SSAS)
Power BI	SQL Server R Services
Common language runtime (CLR)	SQL Server Python Services
Always-on availability groups	SQL Server Master Data Services
Columnar compression	SQL Server Data Quality Services
Partitioning	PolyBase
Graph database	Database alerts and database mail
Package-based installs	
Failover clustering	
Replication	
Backup and restore	
Change data capture (CDC)	
SQL Server Management Studio (SSMS)	
SQL Server Integration Services (SSIS)	
Transparent data encryption (TDE)	
Containers	

Source: Microsoft

There are a handful of Windows-specific features that won't be supported on the Linux side as they are Windows-specific; examples include support of Windows NTFS file system, Windows Cluster Services (instead, the Linux Pacemaker utility is supported), and Stretch Database. There are other existing SQL Server features on the roadmap for Linux, including SQL Server Reporting Services (SSRS) and SQL Server Analysis Services (SSAS). These features did not make it into the initial 2017 Linux release because they still have some dependencies on Windows IIS Server, which must be abstracted.

Don't forget Docker

Another facet of SQL Server 2017's multiplatform support extends to Docker containers. The operable notion about containerized images of the database is simplifying deployment for test/dev and/or continuous integration/continuous deployment scenarios where database images have finite lifespans. Running Docker containers is simpler to set up compared to configuring SQL Server images in virtual machines (VMs). The new Docker feature puts SQL Server on par with other databases such as PostgreSQL and Oracle (which began making Oracle database images available through the Docker Store last year).

Convergence with Azure SQL Database

Rejoining twins separated at birth

SQL Server and Azure SQL database are like long-lost siblings who have come back together at a family reunion. Azure SQL Database, the cloud-native Microsoft relational database of Azure, was based on SQL Server. Given the limitations of earlier versions of SQL Server, the Azure product team developed on a separate fork to support the storage management and multi-data center/multi-region replication capabilities associated with cloud deployment. Furthermore, the ability for cloud platform providers to control their platform offerings means that the pace of new feature introductions typically occurs more frequently compared to on-premises offerings. And so, not surprisingly, there has been some divergence in features.

With the Fall 2017 release, SQL Server and Azure SQL Database have come full circle. For instance, some of the management features emerging in SQL Server 2017, such as automatic tuning and adaptive query processing, had their origins a year earlier in Azure SQL Database. Meanwhile, SQL Server began adding data science-oriented capabilities beginning with the 2016 release that are only now starting to make it into Azure SQL Database.

As off Fall 2017, the respective code bases of SQL Server and Azure SQL Database have merged. That has made it possible for both platforms to concurrently debut features including SQL Graph database and native scoring. And in the future, SQL Server updates will be introduced more frequently now that it is on a common code base with the cloud offering. Microsoft will follow what is becoming the customary cloud-first release model for established database providers who make their offerings available on-premises and in the cloud.

Note: Over the past five years, Microsoft has picked up the pace with SQL Server releases. Before SQL Server 2012, release cycles averaged 3–4 years. That accelerated to two years for SQL Server 2014 and 2016, and 16 months for the current 2017 release.

While the code bases have merged, there are still some feature sets that are not (yet) completely in sync; some of them (e.g. cross-data center/cross-region replication, automated threat detection) are specific to cloud automation. This scenario is not unique to Microsoft; for instance, Oracle is only offering “autonomous database” capabilities with the 18c database managed service that runs in the Oracle Public Cloud.

Otherwise, over time the core “features” (the capabilities that are user facing) will harmonize. They will include the R and Python services, which for now are not as far along in Azure SQL Database. R Services (which debuted in SQL Server 2016) are only in preview on Azure SQL Database, while Python Services have yet to be introduced on the Azure SQL platform. There are some differences with advanced indexing features: automatic index management (which uses machine learning to identify indexes that should be added or removed) are for now only available in Azure SQL Database, while resumable index rebuilds are only available in SQL Server 2017. Table 3 lists some of the new features introduced in Azure SQL Database in the latest release.

Table 3. Azure SQL Database Fall 2017 new features

Category	Feature
Analytics	SQL Graph
	Native scoring (new T-SQL PREDICT function)
	R Services (in preview)
Performance	Automatic tuning (includes index optimization and reverting to last “good” query plan)
	OMS (Operations Mgmt. Suite) Performance Insight (manage multiple database instances in one logical view)
	Adaptive query processing
Monitoring	Automated threat detection
	Centralized OMS dashboard
High availability/cloud integration	Active replicas across multiple regions (up to four supported)
	Managed SQL Server Integration Services (SSIS) (in preview)
	Azure Data Sync integration

Source: Microsoft

Azure SQL Database Managed Instance introduces middle ground for SQL Server users

SQL Server customers wishing to move to the cloud have the option to run the database on Azure as Infrastructure-as-a-Service (IaaS) in an Azure VM (that is, you manage all provisioning, patches, and

backups yourself). Now, Microsoft is about to add a second option: Azure SQL Database Managed Instance, which is currently in private preview and expected to become generally available sometime in 2018. It will be available with two service tiers including General Purpose and Business Critical (which adds faster local SSD storage). It is aimed at SQL Server customers who want to "lift and shift" workloads to the cloud but also want to take advantage of a managed service that automates the housekeeping. In effect, it is a middle ground, providing the equivalent of their private, multi-database environment of SQL Server on-premises instances with the automation of the Azure SQL Database cloud environment. As with any SQL Server to Azure SQL Database migration, it would use Microsoft Azure's Database Migration Service (DMS) service.

As this will be an initial release, there are a number of SQL Server features still on the to-do list for Managed Instance, including filestream, filetable, cross-instance distributed transactions, Master Data services (MDS), Data Quality Services (DQS), and stretch Database.

It starts with VNET isolation that provides a virtual private cloud, without the resource sharing of multitenant environments. Because it supports multi-database instances, you can continue to run common global temp tables, cross-database queries, and SQL agents. Nonetheless, this is not a carbon copy of SQL Server; there are many SQL Server features that are superseded with new alternatives with the cloud managed service, as shown in Table 4.

Table 4. SQL Server feature differences in Azure SQL Managed Database

SQL Server 2017	Azure SQL Database Managed Instance
Always-on Availability Groups	Built-in HA and geo replication
Backup/restore (differential, copy-only log, etc.)	Automated backup, point-in-time restore; copy-only backup will be available in the future
Windows Authentication	Azure Active Directory
Managed Data Warehouse	Azure Operations Management Suite (OMS)
Policy-Based Management	Most tasks are built into Azure SQL Database
SQL Server Analytic Services (SSAS)	Azure Analysis Services
SQL Server Integration Services (SSIS)	Azure Data Factory
SQL Server Reporting Services (SSRS)	Power BI
Database mirroring	Built-in HA/geo-replication
Extended stored procedures	CLR

Source: Microsoft

Analytics

Graph table support

SQL Server 2017 and Azure SQL Database are the latest enterprise databases to add graph support. Specifically, it allows you to create graph tables by supporting new node and edge tables, and MATCH functions for pattern matching. It supports use of most T-SQL commands, so DBAs do not have to learn specialized languages like SPARQL or Gremlin. In that sense, Microsoft's approach is similar to Teradata Aster's, as both treat graph functions as extensions of the SQL language; but in contrast to Teradata, Microsoft's implementation is representing graphs as physical tables, rather than as virtualized data views (an approach often described as "graph engine").

This is not Microsoft's first foray with graph; it also provides a graph API for its multimodel Cosmos DB NoSQL database. And as noted, Microsoft is hardly alone here; others that have already gotten their feet wet with graph include:

- IBM (Compose database for JanusGraph)
- SAP (graph API for in-memory HANA database)
- Oracle (Oracle Spatial and Graph database)
- Teradata (via Teradata Aster SQL-GR graph API)
- DataStax (DSE graph using Cassandra as the storage engine)
- Amazon (which introduced Neptune, supporting property and RDF semantic graphs).

Why all this sudden support for graph? The answer is that graph is uniquely suited for representing many-to-many relationships that, in a conventional relational database, would require endless table joins. Such relationships are surprisingly common in the real world and have proven useful for use cases as varied as customer 360, supply chain optimization, and cybersecurity applications. Until recently, graph was among the best-kept secrets of the database world because of its uniqueness and lack of widely used standards.

SQL Server 2017 is not meant to replace specialized graph databases that can hold multiple graphs (it is limited to a single graph per database instance). Instead, it takes the 80/20 approach in viewing graph as an extension of a SQL database. As this is an initial release, there are still some gaps with data validation for delete functions, conditional range updates (where you can update a specific range of columns), and the capability to designate temporal tables as nodes or edges.

Making Python a first-class citizen

Microsoft cracked open the door to SQL Server for data scientists with the previous release, which added support for processing R inside the database. Now it's taken the next logical step with Python. The same extensibility framework (SQL Server Machine Learning Services) has added support of the popular Anaconda distribution of Python; as with R, Python runs in a separate process so as not to disrupt SQL operations.

Note: As mentioned below, Azure SQL Database is not quite as far advanced with R and Python support.

Not surprisingly, in-database R and Python analytics is associated with analytic databases, such as Teradata (which provides support through user-defined functions), Oracle R Enterprise (which makes

in-database R [not Python] processing available through the Advanced Analytics Option), and IBM (Advanced Analytics System embeds Spark processing). For Microsoft, the differentiator is that in-database R and Python processing are not limited to specialized analytic platforms or options.

Scoring predictive analytics models in T-SQL

SQL Server 2017 and Azure SQL Database now enable practitioners with SQL skills (the sweet spot of the database talent pool) to run predictive analytics models without having to know Python, R, or other specialized languages. This doesn't turn SQL developers into predictive modelers, but it lets database practitioners run scoring queries using them.

This comes via a new T-SQL PREDICT command that lets SQL developers make scoring queries to existing R or Python models that have been loaded into the database. There are some restrictions, however. The models must be written using specific Microsoft libraries for R or Python (RevoScaleR and RevoScalePy, respectively); presumably, that would be the job of data scientists proficient in those languages. T-SQL developers (who could be DBAs or power users) would then specify the data source and choose the model to get predictive scoring results. These capabilities are win-wins for SQL developers, data scientists, and the organizations to which they belong. It provides opportunities for:

- SQL practitioners to enrich their skills by learning how to use predictive scoring while staying within their technical comfort zone (and for the more ambitious, provides them a foothold into data science that they could extend by learning new statistical and programming skills)
- data scientists to operationalize their predictive models
- enterprises to get better return on their data science investments.

A related capability is an additional way to load R packages into the database, expanding on the R *install.packages* function that SQL Server 2016 supported. For SQL Server 2017, a simplified CREATE EXTERNAL LIBRARY command provides some additional control on access to R packages. If the DBA creates it (and takes ownership), the library can be shared by all authorized database users; conversely, if an individual user creates and claims ownership of the library, it is treated as a personal library available only to that user. While this function for now only supports R packages, we expect Python support to appear in an upcoming release.

Performance enhancements

Adaptive query processing

A step toward database automation is a new feature that can be switched on through a rather geeky low-level command ("set compatibility_level = 140") where the database can readjust query plans on the fly for more efficient processing or resource utilization (we're surprised that this isn't a higher-level menu or dashboard option). This picks up where SQL Server's (or Azure SQL Database's) existing query optimizer (designed to automatically select the best query plan) leaves off: where conditions change or queries encounter unexpected conditions during operation. Adaptive query processing can readjust query plan parameters, such as memory utilization, join approaches, and table scans.

Automatic tuning and query plan correction

Another new automatic tuning feature is automatic query plan correction. Just like adaptive query processing, this feature is designed to respond to query plans that prove suboptimal in practice. Previously, you could have gone back and changed the plan manually to an earlier version (a process that was often tedious). SQL Server 2017 (and Azure SQL Database) automates this step with a statement that turns on automatic tuning, which will force the move back to earlier plans when performance falls below a specific threshold set by the DBA. Azure SQL Database adds another enhancement with automatic index management (identifying indexes that should be dropped or added to optimize performance), as noted earlier.

Resumable index rebuilds

Building indexes can be time-consuming and resource-intensive. A new feature that for now only appears in SQL Server (not Azure SQL Database) is the capability to pause and resume index building. This feature is useful for scenarios where indexes cannot be fully built within the time for the maintenance window, or because of glitches attributable to disk space or transaction log issues. Rather than start again, this feature allows indexing operations to pick up where they left off.

Enhancements to SQL Server ancillary services

There are a number of incremental enhancements to the analytic, reporting, and integration services that are bundled with SQL Server. SQL Server Analysis Services (SSAS) adds new dynamic management views that provide more granular analysis in dependencies between partitions, expressions, and data sources that can help troubleshoot issues in analytics reporting. It also adds a new tabular mode that is aligned with several related offerings including the Excel data model, Power Pivot, and Power BI desktop (the desktop extension of this cloud feature). SQL Server Reporting Services (SSRS) adds a comments capability that will improve communication and collaboration in developing reports. Meanwhile, SQL Server Integration Services (SSIS) has added support for Linux in line with release of a Linux version of SQL Server. As a first release on Linux, there are some features that are on the roadmap, such as SSIS Catalog Database, Scheduled package execution by SQL Agent, change data capture (CDC), Hadoop/HDFS support, scale-out, Windows authentication, and Microsoft connector for SAP BW.

Appendix

Methodology

This report was compiled based on extensive discussions with Microsoft, Microsoft customers, attendance at the Microsoft 2017 Ignite conference, and consultation with online documentation.

Further reading

SWOT Assessment: Microsoft Azure, IT0022-000942 (May 2017)

“Multiplatform support highlights Microsoft Ignite data announcements,” IT0014-003348 (October 2017)

“Microsoft Cosmos DB: The new flagship internet database of Azure,” IT0014-003285 (June 2017)

Microsoft's AI Strategy: Capabilities and Impact on Consumer Services, ME0002-000746 (April 2017)

SWOT Assessment: Microsoft OneDrive for Business, IT0021-000244 (March 2017)

Microsoft SQL Server 2016: An Initial Assessment, IT0014-003125 (June 2016)

“Microsoft extends the path with enterprise R,” IT0014-003105 (February 2016)

"Microsoft Azure Data Lake takes big step in taming big data," IT0014-003078 (November 2015)

Author

Tony Baer, Principal Analyst, Information Management

tony.baer@ovum.com

Ovum Consulting

We hope that this analysis will help you make informed and imaginative business decisions. If you have further requirements, Ovum’s consulting team may be able to help you. For more information about Ovum’s consulting capabilities, please contact us directly at consulting@ovum.com.

Copyright notice and disclaimer

Ovum
TMT intelligence | informa

CONTACT US

ovum.informa.com
askananalyst@ovum.com

INTERNATIONAL OFFICES

Beijing
Dubai
Hong Kong
Hyderabad
Johannesburg
London
Melbourne
New York
San Francisco
Sao Paulo
Tokyo