ENZO UNIFIED SOLVES THE CHALLENGES OF OUT-OF-BAND SQL SERVER PROCESSING

Enzo Unified Extends SQL Server to Simplify Application Design and Reduce ETL Processing

CHALLENGES

- SQL Server does not scale out easily, creating CPU and Memory constraints on the servers.
- Sharing data across SQL Server databases is hard and increases maintenance costs.
- SQL Server cannot participate in messaging technologies, leading to costly workarounds.
- SQL Server cannot easily reach out to fetch data from disparate systems, leading to an increase in ETL development costs.

SOLUTION

Enzo Unified is an extensible data service platform that SQL Server can directly tap into, and offers co-processing and bridging capabilities.

BENEFITS

- Reduce complexity of data integration and reduce ETL jobs
- Leverage database developers to access disparate systems
- Allow SQL Server to participate in messaging and service bus communication
- Share tables geographically between SQL Server databases for simpler development

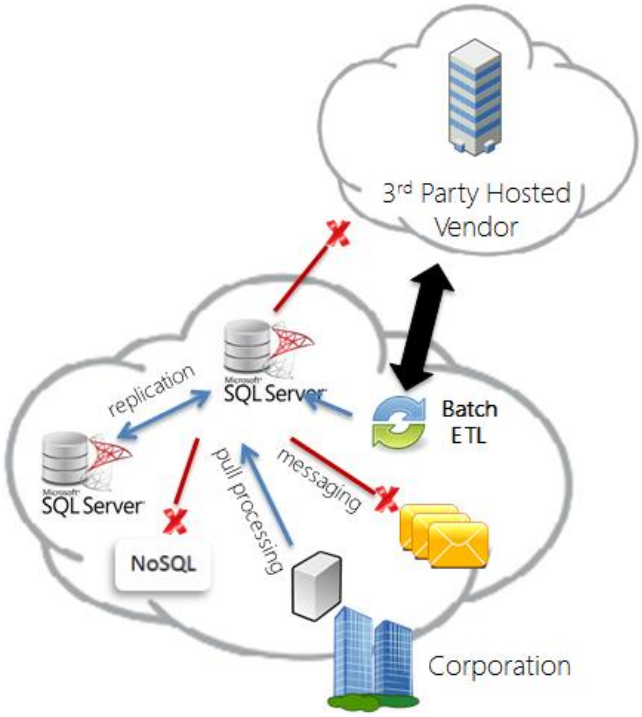
Organizations know that investing in a proven database technology is a fundamental business decision since it is responsible for storing digital assets. Because so much information is stored on database servers, they usually become the cornerstone of most data integration projects, data processing, and data replication. However, database servers such as SQL Server are not designed to be distributed in nature, do not understand NoSQL data formats, cannot communicate to disparate data sources easily, and are not capable of participating actively in messaging architectures. This places natural constraints on how IT organizations can leverage SQL Server and in turn leads to undesired complexities in systems architecture, increases the use of Extract Transform and Load (ETL) tools and batch jobs, and leads to the creation of many custom programs and services that bridge data access and transfer on behalf of SQL Server, leading to significant increase in development and maintenance costs. How can organizations streamline SQL Server integration with other systems and messaging technologies? How can IT departments further leverage their SQL knowledge? How can CPU and memory intensive workloads be offloaded from SQL Server easily? How can developers share data across multiple SQL Server databases transparently without complex replication topologies?

THE CHALLENGE

IT organizations dealing with SQL Server, and most other relational database platforms, know that database servers are confined systems in the sense that they cannot reach out easily to remote or disparate data sources. As a result, organizations spend a lot of time, and money, on building satellite custom programs and ETL jobs to move data in an out of SQL Server.

The inability to leverage SQL Server directly for data integration means that IT organizations depend on development skills that DBAs do not usually offer. As such, importing complex data sets from disparate data sources either requires the development of ETL jobs or custom services that perform the necessary data acquisition and service calls on behalf of SQL Server. While the implementation of these custom jobs and services is needed in some cases, it is usually a symptom of the inability of SQL Server to interact with external data sources, such as Internet services, legacy systems, or more modern NoSQL data stores. This increases the complexity of the technical solutions and raises development and maintenance costs.

In addition, SQL Server lacks the capabilities of NoSQL data stores that offer simpler distributed data storage, in-memory data access across multiple servers, and serialized document storage in JSON format. This means that SQL Server can only be used when an application needs a relational data store. However, many modern applications typically use both types of designs: relational and NoSQL. Applications need relational storage for key information that depends on data integrity rules provided by SQL Server, and NoSQL storage for highly distributed and in-memory data access for high availability and extreme scale. From a data replication standpoint, it can be difficult to setup replication with SQL Server, such as transactional replication, mirroring or log shipping. While these features of SQL Server are widely popular, they come with certain restrictions, both from an application development and maintenance standpoint.



For example, replication works better with SQL Server databases of similar versions, and some data replication features are only available with the Enterprise Edition of SQL Server. While replication, log shipping and mirroring offer extensive support for disaster recovery and high availability, their footprint and complexity can make them overkill for simpler data sharing scenarios that do not warrant such infrastructure. For example, multiple geographically distributed databases could share the same in-memory table for logging database events in the cloud, or store a list of historical purchases across two or more server nodes in a completely transparent way, without involving the overhead and complexities of replication or mirroring. The lack of distributed storage capability and simple data sharing means that IT organizations typically need to make design trade-offs or use advanced features that usually translate into more complex designs and higher implementation and maintenance costs.

As discussed previously, like most relational database servers, SQL Server can scale-up (more memory and more processors on a single machine) more easily than scale-out (distributed processing). This is in part because developers place significant data processing burdens on the database server itself. As a result, some organizations choose a pull processing approach that extracts the necessary data for processing on other machines, and save the result back on the database server at a later time. However, in many cases the work is better achieved closer to the data, and needs to be initiated by the database server itself (push processing), such as complex computations, compression, encryption and more. In SQL Server, Extended Stored Procedures can help with such push processing, providing the advantage of accessing most of the .NET framework for development. However, Extended Stored Procedures have serious drawbacks and as a result are rarely used, including security concerns and additional load on the database server itself. The limitations around out-of-band processing with SQL Server usually yields further complexity in application design and support.

Finally, for similar reasons explained previously, SQL Server is also unable to tap directly into an enterprise or cloud messaging architecture, and as a result further complicates loosely coupled application designs that depend on event subscription and consumption. For example SQL Server cannot tap directly into MSMQ or Azure Queues for messaging, which forces developers to build additional layers of application logic that plug into messaging services.

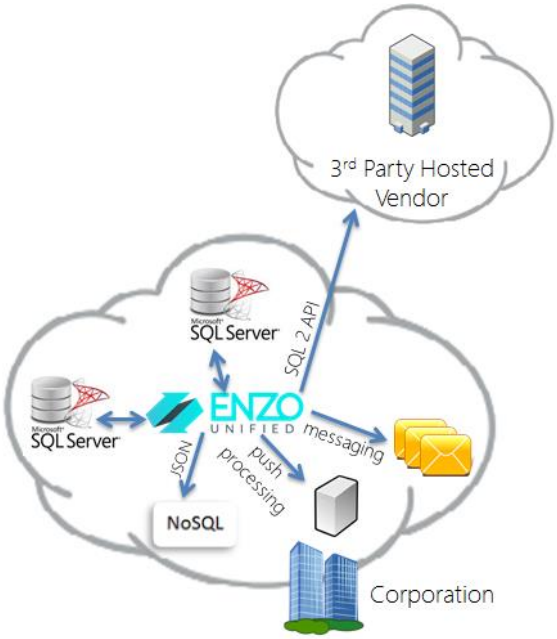
The Enzo Unified Solution

Enzo Unified is an extensible data service platform that SQL Server can directly tap into, and offers co-processing and bridging capabilities that drastically reduce the need for custom development and ETL tools. Enzo Unified exposes vendors' API and services as simple virtual databases that make it trivial to access directly from SQL Server, without ETL processing, batch jobs or temporary storage.

As a result, SQL Server can now tap into third party services directly, in real-time, such as Twitter, SharePoint services, Address Validation services, weather data, Bing translation and much more.

In addition, SQL Server can now also access other internal systems and Internet files natively, such as Excel documents, text files on FTP sites, internal Legacy systems and ERP products just to name a few. In essence, Enzo Unified provides a "SQL 2 API" bridging capability, allowing database developers to run queries against virtually any service endpoint using SQL, eliminating the need to build custom components to access disparate data sources and services, which in turn accelerates project development timelines and reduces implementation complexities.

SQL Server can now tap into NoSQL databases, such as Azure Tables and Couchbase for example, to leverage distributed and highly available document storage and in-memory tables. This capability provides a unique way for SQL Server to share virtual tables easily with other databases, including SQL Server located in remote locations, without using complex technologies like replication. Because this is accomplished using NoSQL databases, it becomes just as easy for SQL Server to tap into NoSQL data stores of existing applications.



For example, a website could be designed to read from a NoSQL database to access its product inventory for display on its main web page, and SQL Server could be used to update the number of remaining items from triggers within SQL Server for inventory changes that take place outside of the website. Alternatively, two or more SQL Server databases geographically dispersed could be using a NoSQL database to share common data, such as an Audit table or a list of products available for both read and write, without the need to replicate information.

This capability drastically simplifies development and provides a new set of design options when building applications that use both relational and NoSQL databases.

Enzo Unified can also serve as an external co-processor for SQL Server, so that CPU or memory intensive operations can take place, synchronously or asynchronously, on one or more servers. This allows SQL Server to offload expensive workloads on other machines using a push mechanism, which offers significant advantages over pull methods. Of importance, offloading workloads on other servers using a push mechanism means that the operation run on demand such as data compression, mathematical computations, data conversion and more. Enzo Unified includes a Software Development Kit (SDK) that allows developers to build extensions to SQL Server using the full .NET framework, allowing the extensions to run on remote servers and thus preserving the resources of the database server.

When it comes to messaging integration, SQL Server offers no option out of the box to integrate with existing message bus technologies, such as the Microsoft Service Bus, or popular messaging systems such as MSMQ. However with Enzo Unified, SQL Server can now tap directly into these popular messaging and bus technologies, and participate in reading and writing messages. This capability offers tremendous opportunities to streamline integration projects with SQL Server, and offers the ability to leverage cloud queues, such as Azure Queues, or even the Azure Message Bus to achieve cross data-center messaging between two or more SQL Server databases.

FEATURES AND BENEFITS

- Reduce complexity of data integration by reducing the need for ETL jobs
- Leverage SQL talent in the organization to access disparate systems and reduce custom code
- Allow SQL Server to participate in messaging and service bus communication
- Share NoSQL tables between any number of SQL Server databases for simpler development
- Scale out SQL Server by offloading data processing logic on other nodes

SUMMARY

While organizations leverage the traditional relational capabilities of SQL Server successfully, many still need to build custom services, ETL jobs, and accommodate application designs to build their solutions due to the limited extensibility of database servers. This can in turn create delays in development and integration projects, and increase maintenance costs. Furthermore, building custom services and ETL jobs often requires skilled developers and introduces project risks.

Enzo Unified solves these challenges by allowing organizations to extend SQL Server using external processing nodes, like a co-processor, so that data access, acquisition and transformation can be performed outside of the SQL Server database server. In addition, Enzo Unified provides a direct bridge to external services, including service bus technologies, messaging architectures and NoSQL data stores to offer radically new design options.

ABOUT ENZO UNIFIED

Enzo Unified is a data platform that helps companies reduce development and data integration project timelines, improve data quality, and increase operational efficiency by solving some of the most complex real-time data consumption challenges. For more information, contact info@enzounified.com, or visit www.enzounified.com.