The Multi-Model Data Management Platform

Flexibility for Cloud Applications in the Right-Now Economy
# CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTRODUCTION</td>
<td>3</td>
</tr>
<tr>
<td>MULTI-MODEL: AN EVOLUTIONARY TALE</td>
<td>4</td>
</tr>
<tr>
<td>FROM RDBMS TO NOSQL TO MULTI-MODEL</td>
<td>5</td>
</tr>
<tr>
<td>DATASTAX AND MULTI-MODEL</td>
<td>6</td>
</tr>
<tr>
<td>DECIDING WHICH DATA MODELS TO USE FOR A CLOUD APPLICATION</td>
<td>6</td>
</tr>
<tr>
<td>Key Value and Tabular</td>
<td>6</td>
</tr>
<tr>
<td>JSON / Document</td>
<td>7</td>
</tr>
<tr>
<td>Graph</td>
<td>8</td>
</tr>
<tr>
<td>CONCLUSION</td>
<td>11</td>
</tr>
<tr>
<td>ABOUT DATASTAX</td>
<td>12</td>
</tr>
</tbody>
</table>
INTRODUCTION

Cloud applications have been a disruptive force in every industry and have changed the way most companies do business. Enterprises that have embraced the cloud application business model are outperforming their competition, while those that haven’t are struggling and trying desperately to catch up.

A cloud application is one with many endpoints including browsers, mobile devices, and/or machines that are geographically distributed, intensely transactional, continuously available, as well as instantaneously and intelligently responsive no matter the number of users or machines using the application. Properly satisfying a cloud application’s data management requirements is no easy task and necessitates a different data management platform than the ones used for the past three decades.

Part of the “new normal” where data and cloud applications are concerned is the ability of the underlying data layer to have the flexibility to handle multiple types of data models that exist in the application and persist each of those data models in a single datastore. This adaptive data management capability is sometimes called a “multi-model” database.

This white paper explores the multi-model concept, its rationale, and how DataStax Enterprise (DSE) realizes the vision of a multi-model database that supports today’s cloud applications.
MULTI-MODEL: AN EVOLUTIONARY TALE

Understanding the why’s and how’s of a multi-model database requires us to go down the memory lane and take a brief look at the various phases in the evolution of data management that have brought things to where they are today.

The first phase of data management evolution saw companies moving from very centralized database deployments to ones that are semi-decentralized. Today, that evolution has continued to deployments that are radically distributed, which also is a must-have requirement of today’s cloud applications (e.g. write, read, distribute, stay-active everywhere).

The next two evolution phases dealt with the concept of polyglot persistence, which refers to the practice of using multiple data storage technologies to support either a single or multiple applications.

Initially, the idea of polyglot persistence came about via the need to separate data management workloads such as transactional, analytics, search, and other workloads, so there is no competition for data or compute resources between systems devoted to these workloads. It started with having distinct applications targeting separate datastores, which worked fine for older applications. However, the emergence of cloud applications required the data management solutions to evolve to be able to handle mixed workloads with a single platform and therefore support mixed workloads of transactions and analytics. Industry analysts cite this with various terms, for example as Hybrid Transaction Analytical Processing (HTAP) by Gartner and as “Translytical” by Forrester.

The third phase of data management evolution resembled the second phase, but instead of mixed workloads it involved support for different data models, another must-have characteristic of a cloud application. The progression began predominantly with the RDBMS (Relational DataBase Management System) data model moving to a mixture of different technologies and data models to, now, a unified platform capable of supporting multiple data models against a single integrated backend.
FROM RDBMS TO NOSQL TO MULTI-MODEL

The RDBMS ecosystem exhibits a few key characteristics such as: a vendor agnostic standard language for the developers (SQL); well-defined separation between logical (developer) and physical (DBA) aspects of the DBMS; and a cohesive set of mechanisms in a variety of languages (drivers) for the applications can interact with the databases. These characteristics allow businesses to adopt data infrastructure technologies without worrying about vendor lock-in.

However, while the above characteristics of RDBMS serve centralized applications well, there are impediments to its adoption for cloud applications:

- A master-slave architecture mandating concessions on uptime and resiliency
- Scale and write-and-distribute-anywhere constraints for cloud application workloads
- Strict adherence to logical data layer constructs such as third normal form (3NF), which values storage efficiency more than application agility
- A rigid data model that make the use of semi and unstructured data extremely cost prohibitive at scale
- The sharding architectural “Band-Aid” that increases the operational expenses exponentially

Some NoSQL technologies like Apache Cassandra™ can address the above mentioned challenges of the RDBMS ecosystem. However, the adoption of NoSQL technology could create a fragmented and expensive data management system for businesses due to the two issues outlined below:

First, polyglot persistence implied that customers would either use a limited set of one of the data models (Key Value, Tabular, JSON, Graph) or perform extract-transform-load operations across data stores. Enterprise use cases such as master data management (MDM), customer-360-view and others, mandated the latter, which resulted in a higher total cost of ownership (TCO).

Second, each NoSQL vendor's mechanism for interacting with the data store was different, both with respect to its dialect and where they lay on the logical/physical divide. This forced application developers to write abstraction layers if they needed more than a single data model in their application. Further, these abstraction layers had to work at very different level across the physical/logical spectrum to keep application development aligned.

Multi-model data management solutions represent the next phase of maturity of the NoSQL industry that aims to eliminate the hurdles around its adoption, especially as it relates to mainstream developers and administrators within the enterprise. This is accomplished by supporting multiple data models against a single, integrated backend. Such a multi-model data management platform should:

- Provide support for more than one post relational data model (e.g. tabular, JSON, graph) at a logical layer for ease of development for application developers
- Ensure all models are exposed via cohesive mechanisms thereby avoiding cognitive context switching for developers
- Provide a unified persistence layer that delivers geo redundant, continuously available characteristics and a common framework for operational aspects such as security, provisioning, disaster recovery, etc.
- Empower a variety of use cases across OLTP and OLAP workloads for lines of businesses within an enterprise to innovate with agility
- Deliver lower TCO to enable wider adoption within centralized IT teams of an organization
DATASTAX AND MULTI-MODEL

DataStax offers DataStax Enterprise (DSE) and DataStax Managed Cloud to meet the multi-model needs of today's cloud applications. DSE, built on the best distribution of Cassandra, is an always-on data management platform with search, analytics, tooling, administration, and security all integrated into a unified platform. DataStax Managed Cloud is DSE delivered as a fully managed, secure, white-glove service, so you can simply focus on innovation that matters most to you and your customers and offload all your operations to DataStax.

Regardless of the data model used, each data model in DSE and DataStax Managed Cloud enjoys the benefits of Cassandra's always-on, distribute-anywhere, active-anywhere, linear-scale architecture. Further, each data model inherits all of DSE's and DataStax Managed Cloud's enterprise capabilities including advanced security, built-in analytics, enterprise search, and visual administration and monitoring.

DECIDING WHICH DATA MODELS TO USE FOR A CLOUD APPLICATION

A cloud application is intensely multi-faceted. For example, a modern retail cloud application includes various modules such as product catalogs, user profile management, fraud detection, recommendation engine, shopping cart, clickstream/log analysis, and others.

Each module of a cloud application may have distinct data model support requirements, but deciding which data model to use for which component can be confusing. One approach is to step back and ask what level of data complexity and connectedness is involved. Modules that have little to no complexity or connectedness (i.e. data relationships) involved can be served by simpler data models like tabular while modules that have greater complexity and require looking at relationships are best handled by a graph data model.

![Data Model Continuum](image)

Figure 3 – The data model continuum represented by complexity and data connectedness.

DataStax Enterprise currently supports four different data models.

**Key Value and Tabular**

Both DSE and DataStax Managed Cloud support key-value and tabular data models. However, the support for Materialized Views along with the widespread adoption of the CQL dialect, presents DSE and DataStax Managed Cloud as more of a tabular store to the application developer.

The specific use cases handled by this data model include almost all time-series data scenarios (e.g. IoT, user activity management, financial streaming) and other similar write-and-read heavy situations. As compared to a relational design, tables in DSE and DataStax Managed Cloud are typically modeled in a denormalized fashion around the queries needed to satisfy requests from the application.
JSON / Document

The JSON/Document model support in DSE and DataStax Managed Cloud has the flexibility to store data with complex nested schemas and is able to easily move data to and from application tiers – both of which are the primary use cases of popular Document-oriented data management platforms. The major difference between DSE, DataStax Managed Cloud and some other data platforms that support JSON is that DSE and DataStax Managed Cloud require the JSON data to adhere to a database schema.

In some schema-less document records, one record might vary significantly from the next, with no expectation that the database will enforce any record structure. DSE, however, retains the requirement that a CQL table holding JSON data be defined with a schema that implements columns and column types in a row. It should be noted that some document data platform vendors are now recognizing the need for schema enforcement for JSON data and have introduced mechanisms to validate data that is loaded into the database.

In DSE and DataStax Managed Cloud, CQL supports collection types (maps, sets and lists) and user-defined types that all map into JSON list and map types. This allows DSE and DataStax Managed Cloud to store records that have structural complexity similar to document-style records without being completely form free. Sparse storage allocation in the underlying storage engine also means that column values can be left null with no penalty, further adding to the flexible structure of the record.

It is possible to define columns that might only be used with a handful of rows in the table, which matches the ability for a JSON document to leave values out of a record. It is also possible to add columns to a table at any time without negatively impacting the performance.

If the requirement is to deal with an incoming data feed that might change as time progresses, this can be handled with a no-cost change to the table schema. In this respect, DSE and DataStax Managed Cloud have many of the Document storage capabilities of pure Document databases except for the ability to handle truly schema-less data without prior knowledge of the structure of incoming records.

There are a variety of use cases that lend themselves to the use of JSON inside DSE and DataStax Managed Cloud. For example, DSE is often used as the “state store” for cloud applications written with JavaScript UI frameworks that are designed to exchange JSON between the browser and the server. With DSE’s support for JSON, the server-side code required to do that is dramatically simplified. Without JSON support, the application developer would have to convert CQL results to JSON before sending query results back to the client. Conversely, incoming data would have to be transformed from JSON into CQL statements. With JSON/Document data model support in DSE and and DataStax Managed Cloud, none of that coding is required.

Lastly, DataStax also provides JSON support through DataStax DevCenter, a visual development tool for writing CQL and JSON queries against DSE. DevCenter includes various JSON editors including syntax highlighting, code completion, code correction, and more, which makes developing with JSON against DSE simple and straightforward.
Graph data model support is realized in DSE through DSE Graph, which is a graph database built for cloud applications that manage highly complex and connected data. DSE Graph delivers continuous uptime along with predictable performance and scale for modern systems dealing with complex and constantly changing data, while remaining operationally simple to manage. Support for graph is present in the DSE Server, the DSE OpsCenter management tool, DSE Studio – a visual developer tool for graph – and lastly, DataStax drivers.

A graph model should be considered for any use case involving complex data scenarios that consist of intense and numerous relationships among the data elements. Since an RDBMS and a graph data platform are similar in that they involve data that contains connections or relationships between data elements, why not just use an RDBMS versus a graph data model like the one found in DSE?
Foundationally an RDBMS and graph data platform differ in the underlying engine each uses to store and access data. There are also some data modeling features that RDBMS’s and graphs don’t share. For example, a graph data platform allows characteristics (i.e. properties) to be assigned to edges, whereas an RDBMS relationships have no such ability.

Another key difference between a graph data platform and an RDBMS is how relationships between entities/vertexes are prioritized and managed. While an RDBMS uses mechanisms like foreign keys to connect entities in a secondary fashion, edges (the relationships) in a graph data platform are of first order importance. In other words, relationships are explicitly embedded in a graph data model.

The following comparisons can be used to help in the decision making process of whether to use an RDBMS or a graph data platform like DSE Graph for a particular use case:

<table>
<thead>
<tr>
<th>RDBMS</th>
<th>DSE GRAPH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple to moderate data complexity</td>
<td>Heavy data complexity</td>
</tr>
<tr>
<td>Hundreds of potential relationships</td>
<td>Hundreds of thousands to millions or billions of potential relationships</td>
</tr>
<tr>
<td>Moderate JOIN operations with good performance</td>
<td>Heavy to extreme JOIN operations required</td>
</tr>
<tr>
<td>Infrequent to no data model changes</td>
<td>Constantly changing and evolving data model</td>
</tr>
<tr>
<td>Static to semi-static data changes</td>
<td>Dynamic and constantly changing data</td>
</tr>
<tr>
<td>Primarily structured data</td>
<td>Structured and unstructured data</td>
</tr>
<tr>
<td>Nested or complex transactions</td>
<td>Simple transactions</td>
</tr>
<tr>
<td>Always strongly consistent</td>
<td>Tunable consistency (eventual to strong)</td>
</tr>
<tr>
<td>Moderate incoming data velocity</td>
<td>High incoming data velocity (e.g. sensors)</td>
</tr>
<tr>
<td>High availability (handled with failover)</td>
<td>Continuous availability (no downtime)</td>
</tr>
<tr>
<td>Centralized application that is location dependent (e.g. single location), especially for write operations and not just read</td>
<td>Distributed application that is location independent (multiple locations involving multiple data centers and/or clouds) for write and read operations</td>
</tr>
<tr>
<td>Scale up for increased performance</td>
<td>Scale out for increased performance</td>
</tr>
</tbody>
</table>
A graph data platform like DSE Graph is a better choice than an RDBMS when it comes to identifying commonalities and anomalies in large, complex, and highly connected datasets. While DSE Graph can be used for a variety of application use cases, the following are some of the most common use cases:

- **Master Data Management**: A graph is the best model for critical business data and their relationships that are consolidated across business units, which is then queried and maintained by various transactional and business intelligence (BI) business applications. Other examples include product catalogs, which often have complex hierarchical structures and are overlaid by taxonomies to capture composition or other relationships. In those cases, the data complexity is high enough to warrant a graph data platform. It is also typical for search functionality to be necessary for master data management use case, which can be handled with DSE Search.

- **Recommendation/Personalization**: Oftentimes, relevant recommendations can be best identified from a large graph of users and entity interactions. A graph is well suited to help recommend products, next actions, or advertising based on a user’s information, past behavior, and interactions.

- **Security Management and Fraud Detection**: In a complex and highly interrelated network of users, entities, transactions, events, and interactions, a graph data platform can help determine which entity, transaction or interaction is fraudulent, poses a security risk, or is a compliance concern. In short, a graph data platform like DSE Graph assists in finding the bad needle in a haystack of relationships and events.

- **IoT, Network Asset Management and Monitoring**: A graph is a good model for managing network assets (with their properties or configurations) and how they relate to each other over time, allowing you to manage and monitor the network, optimize resource allocation, detect and fix problems, etc. This can also include IoT use case where assets are devices or machines that generate time-series data (e.g. status records, event data). One way to approach the data management with DSE is to have the network asset information stored in DSE Graph and the IoT time-series data in Cassandra.
CONCLUSION

Today’s cloud applications involve numerous components that usually differ in their data model support requirements. So, a data management platform that provides adaptive data management (or multi-model) capabilities will deliver a simpler and more agile solution to quickly bring the cloud applications to market. DataStax Enterprise and DataStax Managed Cloud have built-in multi-model capabilities and provide support for key-value, tabular, JSON / document, and graph data models.

Since, DSE and DataStax Managed Cloud are built on top of Cassandra and all the data (regardless of the model) is stored in a single backend, each data model inherits the following benefits of Cassandra:

- Continuous availability
- Easy geographical data distribution
- Operational low latency
- Linear scalability
- Operational maturity
- Simplified development (e.g. drivers that support all data models in each connector)

In addition, each data model benefits from the enterprise capabilities found only in DSE and DataStax Managed Cloud:

- Enterprise-grade security that protects sensitive data
- Functional cohesiveness that includes built-in analytics for analyzing operational data, integrated enterprise search that satisfies modern application search requirements, and an in-memory option for fast read operations
- Simplified visual management via DSE OpsCenter and command line tools (applicable only to DSE)
- Expert 24x7x365 support
- Certified software updates, hot fixes, and formal end-of-life policies

For more information about DataStax Enterprise and DataStax Managed Cloud and to download the DataStax software, visit http://www.datastax.com.
ABOUT DATASTAX

It starts with a human desire, and when a universe of technology, devices and data aligns, it ends in a moment of fulfillment and insight. Billions of these moments occur each second around the globe. They are moments that can define an era, launch an innovation, and forever alter for the better how we relate to our environment. DataStax is the power behind the moment. Built on the unique architecture of Apache Cassandra™, DataStax Enterprise is the always-on data platform and has been battle-tested for the world’s most innovative, global applications.

With more than 500 customers in over 50 countries, DataStax provide data management to the world’s most innovative companies, such as Netflix, Safeway, ING, Adobe, Intuit and eBay. Based in Santa Clara, Calif., DataStax is backed by industry-leading investors including Comcast Ventures, Crosslink Capital, Lightspeed Venture Partners, Kleiner Perkins Caufield & Byers, Meritech Capital, Premji Invest and Scale Venture Partners. For more information, visit DataStax.com or follow us on @DataStax.