# 

SQL Server Analysis Services to Hive

A Klout Case Study

**Author:** Kay Unkroth

**Technical Reviewers:** David P Mariani (VP of Engineering, Klout), Denny Lee

**Published:** September 2012

**Applies to:** SQL Server 2012

**Summary:** Klout (www.klout.com) measures influence across the social web by analyzing social network user data. Klout uses the impact of opinions, links, and recommendations to identify influential individuals. Every day Klout scans 15 social networks, scores hundreds of millions of profiles, and processes over 12 billion data points. The Klout data warehouse, which relies on Apache Hadoop-based technology, exceeds 800 terabytes of data. But Klout doesn’t just crunch large data volumes; Klout takes advantage of Microsoft SQL Server 2012 Analysis Services to deliver reliable scores and actionable insights at the speed of thought. Microsoft and Klout collaborated to build this Big Data Analytics solution. The goal for this solution was to find a cost-effective way to combine the power of Hadoop with the power of Analysis Services. The result is a solution that connects Analysis Services to Hadoop/Hive via the relational SQL Server engine, enabling Klout to reduce data latencies, eliminate maintenance overhead and costs, move aggregation processing to Hadoop, and shorten development cycles dramatically. Organizations in any industry and business sector can adopt the solution presented in this technical case study to exploit the benefits of Hadoop while preserving existing investments in SQL Server technology. This case study discusses the necessary integration techniques and lessons learned.

Copyright

This document is provided “as-is”. Information and views expressed in this document, including URL and other Internet Web site references, may change without notice. You bear the risk of using it.

This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes.

© 2012 Microsoft. All rights reserved.

Contents

[Introduction 4](#_Toc336245928)

[The Klout Big Data solution 5](#_Toc336245929)

[Big Data Analytics based on Analysis Services 6](#_Toc336245930)

[Hadoop/Hive and Analysis Services integration 7](#_Toc336245931)

[Limitations of direct connectivity 8](#_Toc336245932)

[Pass-through queries to linked servers 10](#_Toc336245933)

[Best practices and lessons learned 11](#_Toc336245934)

[Conclusion 13](#_Toc336245935)

# Introduction

Big Data enables tremendous opportunity and competitive advantage for organizations. International Data Corporation (IDC) predicts an explosive growth of more than 60 percent per year for the Hadoop-based Big Data market until 2016, and Gartner states that enterprises should not delay implementation in order to seize early adopter competitive advantages. However, there are no out-of-the-box business intelligence (BI) solutions that organizations can readily deploy on top of Hadoop. There is no one-size-fits-all recipe for integrating Hadoop with traditional data management, analytics, and reporting systems. There are no established guidelines or best practices yet on how to implement Hadoop in the enterprise while preserving existing investments in IT.

Klout, a leading social network analytics company, is an early Hadoop adopter. Not surprisingly, building a system that recognizes millions of people for the power of their voices on social media requires a Big Data Analytics solution. The challenge is to serve up an 800-terabyte data warehouse with over 1 trillion rows of data for ad-hoc analysis. Although Hive supports ad-hoc queries for Hadoop through HiveQL, query performance is often prohibitive for even the most common BI scenarios. A better solution is to bring relevant Hadoop data into SQL Server Analysis Services by using HiveQL. Analysis Services can then serve up the data for ad-hoc analysis and reporting. In this way, Klout can achieve an average query response time of less than 10 seconds on 1 trillion rows of data.

Unfortunately, there is no direct way to connect an Analysis Services multidimensional online analytical processing (MOLAP) database to a Hive data source. A common workaround, and the initial solution for Klout, is to load the data into a MySql staging database by using Sqoop. Analysis Services can then import the data through dotConnect for MySQL or another appropriate data provider. Yet, this approach is not without disadvantages. It introduces overhead in terms of data latencies and complexity, which in turn impacts systems manageability and total cost of ownership.

The Microsoft ODBC driver for Hive enabled Klout to break through the limitations in the initial design, eliminate the MySQL staging database, and connect Analysis Services quasi-directly to Hive. Analysis Services does not support the ODBC driver for Hive, but the relational SQL Server engine can use it through OLE DB for ODBC. SQL Server can serve as an intermediary without a full staging database. In collaboration with Microsoft, Klout devised a comprehensive solution that delivers cost effectiveness at Hadoop’s scale with a single Analysis Services server with direct attached storage (DAS). The new solution rolled out to production in May 2012.

The revised Big Data Analytics design enables Klout to capture, retain, and support queries against all detail data in Hadoop. For example, within this design, Klout can track users and events for A/B testing. It also supports interactive queries, common analytics and reporting tools, and custom BI applications through Analysis Services. The Klout solution leverages the best tool for each specific function without reinventing the wheel. This approach provides the basis to preserve existing IT investments, skill sets, and IT experience, improve systems manageability, and lower total cost of ownership.

# The Klout Big Data solution

Klout started in 2008 with a vision to recognize people for the power of their voices on social media by measuring their influence on others. CEO and Cofounder Joe Fernandez prototyped his idea in Microsoft Excel and developed the solution’s basic principles and core algorithms. The Klout score (Kscore) is based on a variety of factors, including among other things the number of friends and followers, likes and dislikes, the frequency of updates, and retweets.

In April of 2011, Klout officially launched the new *klout.com* site. Initially, the Klout solution captured data from only a single social network: Twitter. Twitter was a natural choice because this social networking service offers flexible interfaces to the global stream of Tweet data. Based on the Tweet data stream, the Klout pipeline calculated Kscores, tracked topics, and plotted the social graph. The Klout solution was initially deployed on a Hadoop cluster of ten nodes and data was served up to clients through HBase and MySql.

The Klout solution, depicted in Figure 1, grew quickly to a system with over 1 petabyte of storage, distributed over more than 150 data nodes in a Hadoop cluster. At the time of this writing, each data node runs on a quad-core SuperMicro server with 32 gigabytes of memory and 10 terabytes of DAS disk capacity. The pipeline includes a multitude of signal collectors that scan 15 social networks every day. Every day, the system scores hundreds of millions of profiles and processes over 12 billion data points. An Apache Pig and Hive-based data enhancement engine performs the necessary data transformations as part of the data flow into an 800-terabyte Hive-based data warehouse.

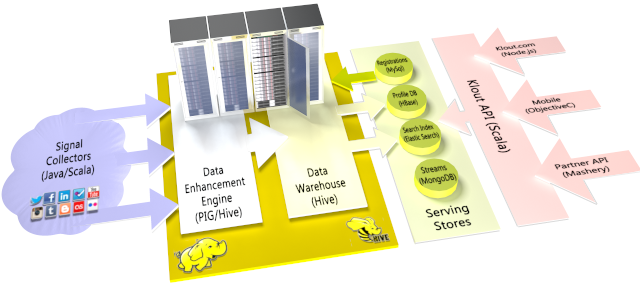


Figure 1   The Klout Big Data solution

# Big Data Analytics based on Analysis Services

The first deployment of a SQL Server Analysis Services (SSAS) cube took place at Klout in June of 2011. Before the cube was deployed, Klout had no easy way of accessing the data across all users. The Klout.com site served data for each user, but Klout lacked a consolidated data warehouse for viewing its entire user population, making it very difficult to drive the models that the Klout scientists needed to further improve the Kscoring algorithms. The lack of visibility made it especially hard to monitor population score changes and track data quality. It was also challenging to launch new service offerings for Klout users.

In August 2011, Klout deployed Hive to optimize the data pipeline and migrate all the individual Hadoop data extracts into a single, consistent Hive data warehouse. Hive enabled Klout to query its large population of user data by using an SQL-like syntax. Klout was also able to retire a myriad of custom data mashups and provide data consistency across its features and product lines, accelerating feature development and drastically improving data quality for customers.

While Hive is an excellent and scalable Hadoop-based data warehouse framework, it isn’t the best choice for serving ad-hoc queries at scale. Klout still needed an interactive query environment that would complement its investment in a Hive data warehouse. Figure 2 shows this interactive query environment based on Analysis Services in the Klout infrastructure.

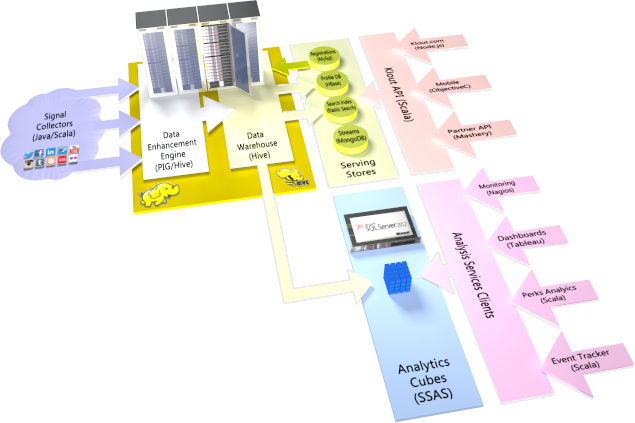


Figure 2 Analysis Services in the Klout Big Data Analytics solution

With the deployment of Analysis Services, Klout user data became accessible in one centralized data store that supported ad-hoc queries across the entire user population. The cube provided Klout with a number of benefits beyond population-level visibility. It helped to answer data support queries, provided insights into the Klout algorithms, and served as a source for quality alerts and service-level agreements (SLA) reports.

*“When it comes to business intelligence, Microsoft SQL Server 2012 demonstrates that the platform has continued to advance and keep up with the innovations that are happening in Big Data.”*

*David Mariani, Vice President of Engineering, Klout*

Integrating Analysis Services with Hadoop/Hive offers Klout the following advantages:

* Delivers cost-efficient OLAP and data mining functionality for a wide assortment of query tools and BI applications.
* Enables Klout to leverage existing knowledge and expertise in building very large cubes with hundreds of billions of rows of data.
* Supports ad-hoc data support queries in Microsoft Excel and dashboards to provide insights into the Klout algorithms. It also serves as a source for quality alerts and SLA reports.
* Offers exceptional query performance over vast amounts of data. At Klout, the average query response time is less than 10 seconds on 1 trillion rows of data.
* Provides a true business view of data to end users in the form of a cube with measures and dimensions. The cube eliminates the need to navigate the complexities of SQL and delivers a rich semantic layer on top of raw, unstructured Hadoop data.
* Offers optimized cube processing performance that enables Klout to add new data to the Analysis Services cubes on a daily basis.

# Hadoop/Hive and Analysis Services integration

Analysis Services provided Klout with a central view of its user data, which made it possible to build dashboards and scorecards for managing business operations. Yet, the process for feeding the raw data to the Analysis Services cube was laborious, time consuming, and fragile. There was no direct way to connect Analysis Services to Hadoop. Klout had to deploy a MySql staging database and write custom MapReduce jobs to create its own data mashups. After the data was loaded into the staging database by using Sqoop, it was imported into the cube through the dotConnect for MySQL data provider.

When the Microsoft ODBC Driver for Hive was introduced at the end of 2011, Klout saw an opportunity to load the data from the Hive data warehouse more directly into the cube without the need for a MySql staging database in-between. The new driver offers several advantages over the MySQL-based approach, such as lower data latencies, a simplified operational environment, and shortened development cycles (because aggregation processing is moved to Hadoop). However, SQL Server Analysis Services does not support ODBC drivers natively. It requires an OLE DB client library or a Microsoft .NET data provider.

## Limitations of direct connectivity

A common way to bring an ODBC driver into an OLE DB scenario is to use the Microsoft OLE DB Provider for ODBC (MSDASQL). For example, MSDASQL can wrap the ODBC Driver for Hive so that Hive appears as an OLE DB-based data source to Analysis Services. This works for tabular Analysis Services databases and for PowerPivot scenarios in Excel, as Figure 3 illustrates, but the Klout Big Data solution relies on a multidimensional cube. In multidimensional scenarios, directly connecting Analysis Services to Hive via MSDASQL does not work because the logic for loading data into a multidimensional model is more complex than the logic for loading data into a PowerPivot database or a tabular database.

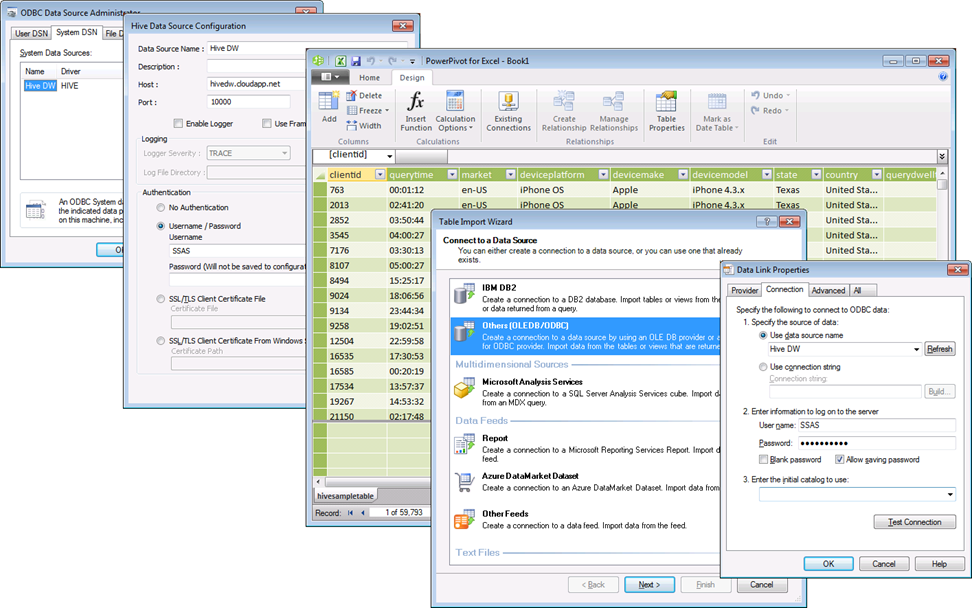


Figure 3   Connecting to a Hive data source in PowerPivot

The first obstacle for Klout to use MSDASQL over ODBC for Hive is that Analysis Services does not ship with a cartridge for HiveQL. A cartridge is an Extensible Stylesheet Language (XSL) file that transforms the abstract Extensible Markup Language (XML) queries that Analysis Services uses internally into an SQL dialect that the data source can understand. Cartridges are located in the installation folder of each Analysis Services instance under *\OLAP\Bin\Cartridges* and identified based on the provider name and version. Because there is no HiveQL cartridge, Analysis Services falls back to the Sql2000.xsl cartridge, which generates a query in a Transact-SQL dialect.

In Tabular Mode cases, the Sql2000.xsl cartridge is applicable to Hive data sources because there are many similarities between Transact-SQL and HiveQL. Queries are less complex because tabular data bindings reference relational tables directly but multidimensional models rely on one more layer of abstraction between the data bindings and the data sources—the data source view (DSV). The DSV defines the logical layout of the data sources and the bindings define which tables, rows, and columns Analysis Services should use to populate the cubes. Analysis Services uses the DSV together with the bindings to generate the internal XML queries, which in turn generate the SQL queries against the data source. The issue is that Business Intelligence Development Studio and SQL Server Data Tools are unable to generate an appropriate DSV for ODBC data sources through MSDASQL. These tools use the .NET Framework Data Provider for OLE DB (System.Data.OleDb), which does not support the OLE DB Provider for ODBC. For this reason, the Connection Manager in Business Intelligence Development Studio and SQL Server Data Tools does not offer the OLE DB Provider for ODBC as an option, as depicted in Figure 4. Hence, the Klout Big Data solution requires a different integration approach.

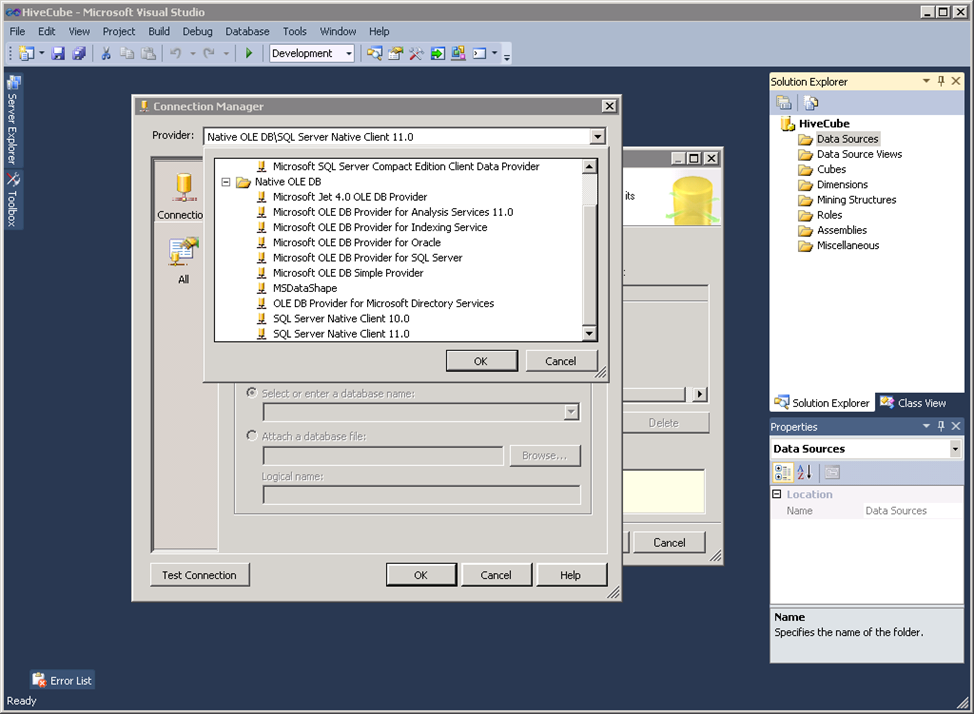


Figure 4   OLE DB Provider choices for multidimensional models

## Pass-through queries to linked servers

The fact that multidimensional Analysis Services models cannot directly use ODBC for Hive via MSDASQL doesn’t imply that Klout cannot connect Analysis Services quasi-directly to its Hadoop/Hive data warehouse. The key is to use a relational SQL Server database engine instance as a relay or proxy to execute commands against external OLE DB data sources, including MSDASQL and therefore ODBC for Hive. Figure 5 illustrates how Klout established Analysis Services-Hive connectivity.

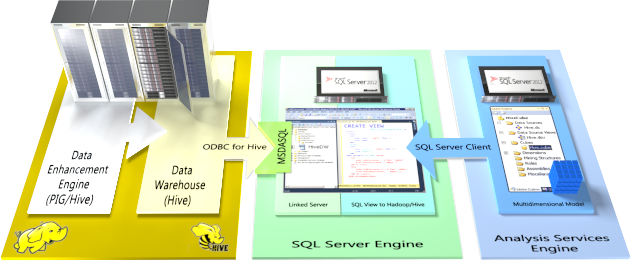


Figure 5   Integrating Analysis Services via a linked server and an SQL view

At the outset, Klout configured the following components to establish connectivity between a relational SQL Server instance and the Hadoop/Hive data warehouse:

* A system data source name (DSN) for the Hive ODBC connection. The DSN points to the host that is running Hadoop/Hive and specifies important connection parameters. System DSNs can be created by using the Data Sources (ODBC) administrative tool.
* A linked server object. The following Transact-SQL script illustrates how to create a linked server that points to a Hive data source via MSDASQL. The system DSN in this example is called “Hive DW”.

EXEC master.dbo.sp\_addlinkedserver

@server = N'HiveDW', @srvproduct=N'HIVE',

@provider=N'MSDASQL', @datasrc=N'Hive DW',

@provstr=N'Provider=MSDASQL.1;Persist Security Info=True;User ID=SSAS;

Password=P@assw0rd;

* An SQL view that is based on an OpenQuery Transact-SQL command. The OpenQuery command connects to the data source, runs the query on the target system, and returns the ResultSet to SQL Server. The following Transact-SQL script illustrates how to create such an SQL view to retrieve rows from a sample table in Hive.

CREATE VIEW vw\_tbl\_HiveSample AS

SELECT \* FROM OpenQuery(HiveDW, 'SELECT \* FROM HiveSampleTable;')

Because these components were created on the computer running SQL Server, it is straightforward to connect Analysis Services to Hive in Business Intelligence Development Studio or SQL Server Data Tools:

1. Launch the Data Source Wizard and use SQL Server Native Client to connect to the SQL Server database that hosts the SQL view to Hadoop/Hive.
2. Launch the Data Source View Wizard, select the SQL Server database as the relational data source, and then include the SQL view to Hadoop/Hive in the DSV.
3. Launch the Cube Wizard and create a cube on top of the DSV.

# Best practices and lessons learned

*“Big Data Analytics solutions are inherently complex. The sheer data volume can be overwhelming, but you can succeed if you rigorously streamline the design and use the best tool for each purpose. Use what works and keep it simple.”*

*Denny Lee, Principal Program Manager Lead, Microsoft*

With its Big Data cube deployment complete, Klout has gained in-depth knowledge on how to integrate Analysis Services efficiently with Hadoop/Hive. This paper shares that knowledge with the Big Data community. Klout reviewed the lessons learned and then, together with Microsoft, developed the following best practices to help customers reduce planning, deployment, and management overhead while achieving a solid design and ensuring smooth daily operations.

Best practices to implement and maintain an Analysis-Services-based Big Data solution:

* **Avoid using a traditional database management system for staging purposes.**   Rather, write the data once on Hadoop and then use Hive in combination with linked servers and views on SQL Server to make the data accessible to Analysis Services. This approach reduces latencies and eliminates data duplication in the warehouse. It offers the advantage to optimize Hadoop for storage and write operations without compromising SQL-based access to the data.
* **Use the SQL Server OpenQuery interface for heterogeneous joins.**   The OpenQuery command executes a pass-through query on a specified linked server provides in this way the basis to expose HiveQL queries as pseudo-tables in SQL Server. It is possible to join pseudo-tables from different data sources and create a view of the joined data in SQL Server.
* **Leverage Hive user-defined functions (UDFs) to transform complex data types, such as JSON, into rows and columns that Transact-SQL can understand.**    By using Hive UDFs, it is possible to expose just about any type of unstructured or structured data to HiveQL and in this way to OpenQuery and in turn Analysis Services.
* **Make sure Hive UDFs are permanent and visible to the ODBC provider.**   Hive UDFs are typically registered by using *Create Temporary Function* statements at the beginning of a Hive session. This means that UDFs are not available over ODBC for Hive connections, unless the UDFs are defined as permanent functions. Common options to define permanent UDFs include:
  + Adding the Hive commands to the *.*hiverc file to execute them automatically when Hive starts. This typically works only for the Hive command-line interface (CLI).
  + Converting the UDF to a built-in function and recompiling the Hive code. This definitely works, but it requires detailed knowledge of the Hive source code.
  + Updating the Hive function registry to add the UDFs to the built-in function list. If you choose this option, you need to modify the *FunctionRegistry* class to register the UDFs that are defined in an hql file (in HDFS) that includes all *Create Temporary Functions* and then add all the dependent jars in *hive.aux.jars.path* properties.
* **Pad zero-length string data.**    The ODBC Provider for Hive might not correctly handle zero-length string data returned from Hive. Common error messages for linked servers to Hive via MSDASQL include “*Cannot fetch a row from OLE DB provider "MSDASQL" for linked server…*”, “*Failed to get data for column.*”, “*Column index out of bounds.*”, and “*Option value changed*.” To avoid these issues, avoid returning empty strings from Hive. The following script illustrates this approach for querying Hive through a linked server connection.

SELECT   
 State =   
 CASE  
 When state = 'empty' Then Null   
 Else state  
 END,  
 Country =   
 CASE  
 When country = 'empty' Then Null  
 Else country  
 END  
 FROM OpenQuery(HiveDW,'SELECT   
 CASE  
 WHEN LENGTH(state) = 0 then ''empty''   
 ELSE COALESCE(state, ''empty'')   
 END AS state,  
 CASE  
 WHEN LENGTH(country) = 0 then ''empty''   
 ELSE COALESCE(country, ''empty'')   
 END AS country  
 FROM HiveSampleTable')

The OpenQuery function executes a HiveQL statement that determines whether any of the specified string columns (in this case *state* and *country*) have zero-length or Null values. If so, the statement inserts or coalesces the text “*empty*” into the string to ensure that it is not empty. When the results are returned to SQL Server, the outer SELECT statement reverts to the Null values. This step is performed for every string column within Hive to ensure that the ODBC Provider for Hive can return the result set.

* **Manage large dimensions by using Hive views.**To save storage space and processing time, use only the attributes that are necessary for analysis. This is a best practice in Analysis Services. For example, if a user is no longer using a service, that user can be excluded from a corresponding Users dimension for this service. An efficient way to limit the dimensions is to create Hive views that link to the facts in the fact tables. Such views can also help to prune fact table partitions more efficiently.
* **Keep Hive objects in the default schema.**   Microsoft’s Big Data solution includes a Hive add-in for Microsoft Excel, which is released as part of Hadoop on Windows Azure. This add-in provides the ability for Excel to connect to the Hive data warehouse framework in a Hadoop cluster. However, the current version of the Excel Hive add-in is designed to query only the default schema. It is therefore a good idea to place all Hive tables and other objects in the default namespace. For links to additional information about the Excel Hive add-in, see “For more information” at the end of this paper.

# Conclusion

By connecting Analysis Services to Hadoop/Hive via a linked server and an OpenQuery-based SQL view, Klout was able to reduce data latencies, eliminate maintenance overhead and costs, move aggregation processing to Hadoop, and shorten development cycles dramatically. In comparison to the previous architecture based on a MySql staging environment, the data now moves six times faster across the pipeline, which enables Klout to load far more data into the Analysis Services cubes on a daily basis.

Analysis Services is a vital piece in the Klout Big Data Analytics solution architecture because it provides a true business view of the data to end users in the form of a cube with measures and dimensions, hiding the complexities of SQL and delivering a rich semantic layer on top of raw, unstructured Hadoop data. The new design helps to expose all of the data that was previously tucked away in Hadoop. Analysis Services supports a wide range of query tools and BI applications and enables data scientists to run ad-hoc data support queries in Microsoft Excel. Dashboards provide insights into the Klout algorithms. The cubes also serve as a source for quality alerts and SLA reports.

At Klout, there is no doubt that Analysis Services is the best tool for the job at hands. The Analysis Services engine offers exceptional query performance over vast amounts of data. Klout enjoys average query response times of less than 10 seconds on 1 trillion rows of data. All queries are served by a single server running Windows Server 2008 R2 with 24 cores, 128 GB of RAM, and 10 terabytes of DAS.

Klout will continue to push the limits of data analysis to maintain its competitive advantage and manage social presence. Thanks to Analysis Services, the Klout solution architecture is flexible enough to provide the required capacity for future growth without exploding costs or complexities. Klout engineers like David P Mariani have the experience to build very large cubes at big-data scale, and they keep sharing their knowledge and expertise so that others can succeed with Big Data Analytics as well.

**For more information:**

<http://klout.com/understand/score>: Understanding Klout's mission and how Klout measures your online influence.

<http://corp.klout.com/blog/>: The official Klout blog.

<http://go.microsoft.com/fwlink/?LinkId=54833>: Big Data Analytics with Microsoft SQL Server

<http://www.microsoft.com/sqlserver/>: SQL Server Web site

[http://technet.microsoft.com/sqlserver/](http://technet.microsoft.com/en-us/sqlserver/): SQL Server TechCenter

<http://msdn.microsoft.com/sqlserver/>: SQL Server DevCenter

<http://go.microsoft.com/fwlink/?LinkId=264857>: Information on how to connect Microsoft Excel to Hadoop on Windows Azure via the Microsoft ODBC Provider for Hive.

Did this paper help you? Please give us your feedback. Tell us on a scale of 1 (poor) to 5 (excellent), how would you rate this paper and why have you given it this rating? For example:

* Are you rating it high due to having good examples, excellent screen shots, clear writing, or another reason?
* Are you rating it low due to poor examples, fuzzy screen shots, or unclear writing?

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

[Send feedback](mailto:sqlfback@microsoft.com?subject=White%20Paper%20Feedback:%20SQL%20Server%20Analysis%20Services%20MOLAP%20to%20Hive).