Cisco Data Virtualization Technical Overview

This document is intended to provide clients with information about the basic concept, functions, and features of Cisco® Data Virtualization.
# Contents

1. **INTRODUCTION** ..................................................................................................................... 3

2. **CISCO DATA VIRTUALIZATION PLATFORM** ............................................................................ 3

   2.1 **CISCO INFORMATION SERVER** ..................................................................................... 4

      2.1.1 Data Access Components ......................................................................................... 4

      2.1.2 Query Processing Components ............................................................................... 6

      2.1.3 Metadata Management Components ....................................................................... 9

      2.1.4 Caching Components ............................................................................................... 9

      2.1.5 Security Components ............................................................................................... 11

      2.1.6 Data Delivery Components ..................................................................................... 13

      2.1.7 Transaction Processing Components ..................................................................... 13

      2.1.8 Triggers .................................................................................................................... 14

   2.2 **CIS DEVELOPMENT** ........................................................................................................... 14

      2.2.1 CIS Studio ................................................................................................................ 14

      2.2.2 Relational Views ..................................................................................................... 15

      2.2.3 Data Services ......................................................................................................... 16

      2.2.4 Team Development ................................................................................................ 17

3. **CISCO DATA VIRTUALIZATION PLATFORM OPTIONS** ...................................................... 17

   3.1 **BUSINESS DIRECTORY** ................................................................................................ 18

   3.2 **CIS DISCOVERY** .......................................................................................................... 19

   3.3 **CIS ACTIVE CLUSTER OPTION** .................................................................................. 20

      3.3.1 Distributing Queries Across a Cluster ....................................................................... 21

      3.3.2 Metadata Synchronization ...................................................................................... 21

      3.3.3 Caching within a Cluster ......................................................................................... 21

   3.4 **CIS ADAPTERS** .............................................................................................................. 22

   3.5 **DEPLOYMENT MANAGER** .......................................................................................... 22

   3.6 **CIS MONITOR** .............................................................................................................. 23

4. **CISCO DATA VIRTUALIZATION SYSTEMS MANAGEMENT** ........................................... 24

   4.1 **CISCO DATA VIRTUALIZATION PLATFORM SYSTEM MANAGEMENT TOOLS** .......... 25

5. **CONCLUSION** ......................................................................................................................... 26
1. Introduction

Information is power. But turning the growing amount of data available from big data, cloud applications, mainframes, relational databases, flat files, and other sources into instantly accessible information can be daunting.

New approaches are needed. Following the path of storage, server, and application virtualization, enterprises and government agencies are applying data virtualization to their critical information needs.

Data virtualization is an agile data integration method that simplifies information access. Unlike integration approaches such as data consolidation via data warehouses and extraction, transformation, and loading (ETL), or data replication via enterprise service bus (ESB) and FTP, data virtualization obtains data from diverse sources on demand without requiring extra copies. This streamlined approach lets businesses derive greater value from growing volumes of data faster and with far fewer resources.

Cisco Data Virtualization can help customers increase profitability, speed time-to-market, reduce costs and risk, and improve compliance and overall agility. This paper provides a technical overview of the Cisco Data Virtualization Platform and shows how it is uniquely suited to the requirements of today’s enterprise-scale data virtualization challenges.

2. Cisco Data Virtualization Platform

The Cisco Data Virtualization Platform is designed to solve the toughest data proliferation, data complexity, and data availability problems faced by the world’s most complex organizations. Leveraging Cisco Data Virtualization Platform’s query optimization, modeling techniques, and visualization helps organizations gain timely and comprehensive business insights.

The Cisco Data Virtualization Platform supports a complete data virtualization development lifecycle and high-performance operations, including:

- **Data Analysis** – CIS Discovery helps to locate and model key entities and relationships during the requirements development and high-level design phases.

- **Development** – Developers employ the easy-to-use CIS Studio development environment with automated code generators to create high-performance standards, compliant views and data services. Rich tools enable complex federation and transformation functions. Standard adapters simplify access and publication development activities. The Manager controls security, metadata, source code and more. Adapters, with pre-built objects for leading enterprise resource planning (ERP) suites and SQL to MDX translators further automate and accelerate critical view and data service development activities.

- **Business Directory** – Business users use the self-service tool to quickly and easily search, browse, and select business data from a user friendly directory of views. With their preferred analytic or BI tools, users can obtain the desired data in Business Directory to drive their business decisions and actions without the need to have a higher level of technical expertise to view the data using CIS Studio.

- **Run Time** – At run time, the Cisco Information Server’s query engine securely queries, accesses, federates, abstracts, and delivers data to consuming business solutions on demand. Multiple caching options provide additional speed and flexibility.
• Management – Enhancing the Cisco Information Server’s fully scalable architecture, CIS Monitor and CIS Active Cluster provide the monitoring, load balancing, high availability, and failover required for scale-up, scale-out performance. Automation tools, such as Deployment Manager, simplify migration and promotion of configurations and settings across large data virtualization environments, and helps minimize deployment risks.

Figure 1. Cisco Data Virtualization Platform Architecture

2.1 Cisco Information Server

The Cisco Information Server (CIS) is a pure Java, multi-threaded application that forms the core functions of the Cisco Data Virtualization Platform.

CIS provides 24-hour availability and does not depend on a third-party application server. It connects to multiple, complex data sources and streams arbitrarily large result sets from those sources to an arbitrarily large number of requesting clients. The server software also intelligently uses all available hardware resources and gracefully throttles back as resources become fully utilized.

The server comprises a number of major system components that work together to provide secure, high-performance data services.

2.1.1 Data Access Components

Data source access is the lowest layer in the CIS platform, sitting closest to the data as it naturally exists in the enterprise. It provides protocol-specific and vendor-specific access to all of the disparate data sources required for your data virtualization implementation. At design time, it examines (introspects) the data source to expose its structure and data types to CIS. At run-time, it normalizes and streams data in forms that can be efficiently manipulated by CIS.
Each data source type available in CIS uses an associated driver. When you provision a new data source for use in CIS, you set the driver’s parameters so that it can access a specific data source of that type (such as an Oracle database). Once connectivity is established, CIS introspects the data source structure and stores the resulting metadata in the CIS metadata repository. With the data source’s metadata now available, you can create views and data services using that data source.

One of the most important roles of the data source access layer is normalizing both the shape and type of the data. CIS normalizes data sets into one of three shapes: tabular (rows and columns of values), hierarchical (such as an XML document), or scalar (a single value). It normalizes each datum within a data set into a standard SQL or XML data type (all of which are listed in the CIS reference manual). Data normalized into these standard shapes and types can then easily be manipulated, transformed, and combined with other data, without regard for vendor-specific or protocol-specific details.

It’s worth noting that even though the shape and type of the data are normalized by the data access layer, the values remain unaltered as they pass through the data virtualization layer. This practice is important for both accuracy and efficiency, and CIS has been carefully engineered to keep data values as close to their native forms as possible.

CIS can integrate data from a long list of data source types. All of the compatible data source types are listed in the CIS data sheet. Each of data source type belongs to a shorter list of data source classes that are described below:

- **Relational Databases** – These tabular data sources include the familiar Oracle, DB2, SQL Server, Sybase ASE, Sybase IQ, Teradata, Greenplum, Netezza, and Vertica databases. All relational databases are accessed using a vendor-specific JDBC driver, around which CIS wraps its own driver framework in the data access layer. Any database for which a standard JDBC driver exists can be accessed by CIS.

- **Web Services** – More and more data is becoming available through standardized web service interfaces, which may include Simple Object Access Protocol (SOAP), XML/HTTP, and Representational State Transfer (REST). CIS integrates the most prevalent web service standards to access the XML data from these sources.

- **Multi-dimensional Sources** – CIS accesses data from SAP Business Warehouse (BW) by transforming SQL queries to MultiDimensional eXpressions (MDX) queries via adapters. In effect, this multi-dimensional data is transformed into slices of tabular data so the CIS can perform further federation, abstraction, and delivery functions. The adapter also allows CIS to access SAP Business Explorer (BEx) Queries, providing users with the enhanced insight and visibility needed to effectively run their businesses.

- **Packaged Applications** – Most industry standard applications like SAP, Siebel, Oracle E-Business Suite, and Salesforce.com provide their own vendor-specific APIs. CIS adds value to these APIs with a collection of Adapter products that expose these APIs and key data entities as common objects.

- **Big Data** – Big data analytics provides unprecedented analytic opportunities for business innovation, customer retention, and profit growth. By simplifying access to Hadoop data through Hive, Cloudera, and Hortonworks, CIS enables organizations to include big data sources as well as enterprise and cloud in their analytics.

- **Mainframes** – A significant amount of enterprise data still resides inside mainframes. Some mainframes can be accessed with modern database standards like Java Database Connectivity (JDBC), and CIS handles...
those as normal relational databases. Other mainframes are unique legacy systems that require specific expertise and specialized software to access efficiently.

- **Files** – CIS can access data directly from certain kinds of structured files, including delimited tabular files, XML files, and Excel spreadsheets.

- **Custom** – The universe of enterprise data sources is extremely large and varied. Some data sources may not offer any standards-based API for accessing data. CIS allows you to create custom data source drivers using Java. If there is a way to read data from a given data source, it is almost always possible to create a custom driver that allows CIS to use the data.

- **Data Source SDK** – Cisco believes in promoting vendor neutrality and openness so customers reap the greatest value from their data virtualization investment. The Data Source SDK (software development kit) opens up the CIS to a larger community of consultants and developers providing them a feature-rich toolkit for developing CIS adapters for non-standard data sources.

### 2.1.2 Query Processing Components

Central to the CIS architecture is a very-high-performance query processing engine. The engine uses advanced query-plan optimization and data-streaming technologies to very quickly return complex, federated results. Sophisticated rule- and cost-based query-optimization strategies examine the plan of each query, and then create a plan that optimizes processing and performance. The engine is designed to minimize overhead and aims to leverage efficient join methods that outperform hand-coded queries written by a typical developer. In so doing, the engine employs a number of techniques, including:

- **SQL Pushdown** – The Query Processing Engine tries to offload as much query processing as possible by pushing down select query operations such as string searches, comparisons, local joins, sorting, aggregating, and grouping into the underlying relational data sources. This allows the engine to take advantage of native data source capabilities and limit the amount of intermediate data returned from each data source.

- **Parallel Processing** – The engine tries to optimize query execution by employing parallel and asynchronous request processing. After building an optimized query plan, the engine executes data service calls asynchronously on separate threads, reducing idle time and data source response latency.

- **Distributed Joins** – The Query Processing Engine detects when a query being executed involves data consumed from different data sources and tries to employ distributed query optimization techniques to improve overall performance and minimize the amount of data moved over the network. The engine optimizes queries and takes advantage of sort-merge, semi, hash, and nested-loop joins, depending on the nature of the query and types of data sources.

- **Caching** – CIS can be configured to cache results for query, procedure and web service calls on a per-view and per-query basis. When enabled, the caching engine stores result sets either in a local file-based cache or in a database. The Query Engine will verify whether results of a query it is about to execute are already stored in the caching system and will use the cached data as appropriate. While caching can improve the overall execution time, its impact is most pronounced when used on frequently invoked queries that are executed against rarely changing data or high-latency sources such as web data services.
Complete Set of Join Algorithms – Select and employ the most efficient join strategy for a given situation (such as Hash Join, Sort-Merge Join, Distributed Semi-Join, Data-ship Join, and Nested-Loop Join) to ensure the most efficient data processing.

Single-Source Join Grouping – Execute data-reducing joins in the data source rather than bringing the data across the network.

Projection Pruning – Eliminate all unnecessary columns from fetch nodes in a query tree.

Constraint Propagation – Distribute filters to multiple branches of the query plan, allowing data reduction by a single filter to potentially occur in multiple data sources.

Scan Multiplexing – Re-use datasets that appear in multiple places in a single query plan.

Empty Scan Detection – Detect logical conditions that would produce empty data sets, and then eliminate those parts of the query plan prior to execution.

Redundant Operator Cropping – Eliminate redundant or extraneous operators within a complex multi-operator query.

Blocking Operator Pre-fetching – Proactively execute parts of the query plan that must complete before other parts of the query plan can continue, thereby increasing overall responsiveness of the query.

Result Streaming – Stream data to consuming applications as results are obtained and processed from the underlying sources.

The overall goal of these optimizations is to reduce the amount of data that must be moved across the network to satisfy the request. Network bandwidth is generally the scarcest resource in the processing pipeline, so reducing the amount of data that needs to be transferred has a significant impact on the latency and overall performance of a data virtualization implementation.

Query processing occurs in several phases:

Plan Generation and Optimization Phase – The plan generator and cost-based optimizer are responsible for producing one or more plans for each query. The optimal plan is automatically selected based on rules and dynamic cost functions. The cost function measures the time and computing effort required to execute each element of the query plan. Cost information comes from a service that pulls data from the repository about each data source in the query, as well as other environmental information during plan generation. For each data source, capabilities that are factored into plans include indexes, join support, and cardinality estimates.

The plan generator also compensates for the fact that each data source might have different capabilities that need to be normalized across the distributed and disparate data sources to successfully resolve the query. It’s important to note that the cost-based optimizer can work with data sources with limited metadata (such as a Web service) as well as systems that have a significant volume of metadata (such as Oracle). Also, during plan generation, the server employs non-blocking join algorithms that use both memory-based and memory-disk algorithms to manage large result sets, and that have underlying data source awareness for pushing joins and predicates to underlying sources for execution.

Query Execution Phase – Proprietary algorithms enable CIS to stream results from massively complex queries while the user query is still running. This lets applications begin consuming data from the server before the query completes execution, and results in substantial performance gains. Queries are
decomposed and executed in the target data sources, and the streaming results are assembled and pipelined directly to the user’s client software. The parallel multi-threaded architecture of the server is truly exploited in this module when multiple queries with different run times across multiple data sources are all required to resolve a single composite query. The execution engine also does dynamic plan re-writing based on actual results found at run time.

- **Query Plan Enhancement Phase** – An out-of-the-box installation of CIS will do a very good job of optimizing queries. However, you can also enhance these query plans to improve optimization results.

During plan generation and execution, extensive statistics are collected and logged for use in a powerful query plan visualization tool found within the Studio. CIS developers use these insights to learn about the cardinality and distribution of data in the data source’s tables, which in turn allows query optimization to make better decisions about join algorithms and ordering.

Join options let you manually specify some of the run-time behaviors of the join instead of letting query optimization automatically determine the behavior. For example, you can select the actual algorithm with which the join will be processed.
Query hints can be included to help query optimization do its job. For example, you can provide the expected cardinality of either or both sides of a join. All available query hints are outlined and explained in the CIS Reference Manual.

### 2.1.3 Metadata Management Components

CIS provides a number of features to manage the metadata required to create, maintain, and control views and data services.

The Metadata Repository stores and manages the links between CIS views or data services and underlying source systems. The repository handles explicit and derived metadata, such as metadata that may be generated for web services.

Also, all data source metadata is “persistent,” so the repository can track and understand changes that have occurred to the structure of the underlying data sources. Any composition logic, packaged queries, and transformations are also stored here. The repository also captures all user and group information, event and log data, and system management and configuration information. Finally, the server ships with views, much like system tables, providing visibility into all metadata and underlying data, and the metadata repository can be exported to be shared, backed up, and support migration.

Key metadata management subcomponents include:

- **Views and Data Services Metadata Repository** – CIS resources are the core building blocks within the platform. This repository forms an enterprise ontology that is driven and created by the processes of building views and data services. The transformations, joins, entitlements, and data source and view dependencies captured as metadata in each CIS data service are managed in this repository. The business metadata associated with that view is also captured and searchable by users.

- **Version Control** – CIS supports improved collaboration capabilities, including resource externalization and resource locking, to enable better resource sharing across projects and tighter design-time controls as new data services are designed and developed. Read more about version control in the Team Development section below.

### 2.1.4 Caching Components

Caching enables CIS to store a queried result-set for a period of time so it can be re-used to satisfy another client request. The cached result-set might be an intermediate result that can be used to satisfy a variety of higher-level requests, or it might be a very specific final result that satisfies a single specific request. Either way, the CIS developer can specify when, where, and how often result-sets get cached.

Result-set caching is generally used for two reasons. First, the cached result-set is immediately available to satisfy a new request, which improves performance by reducing the overall latency of the request and increasing responsiveness to the client. However, it is probably more accurate to characterize caching as a reduction in the amount of work the system needs to do to satisfy a request. The system is more efficient and therefore provides improved responsiveness.

The second reason to use result-set caching is to reduce potential impact on underlying data sources. As CIS satisfies requests at the time they are made, each request usually results in a corresponding data request to one or more underlying data sources. Depending upon the frequency of requests, it is possible to place significant load on
underlying systems to satisfy the requests. In some situations this may be necessary and desirable, but in other cases it may be prudent to buffer requests to the underlying systems by using result-set caching. For example, a transactional system needs to be most responsive to the transaction creation process (such as order taking) during peak hours and it should probably not be servicing reporting needs at the expense of transactional needs. By employing result-set caching, the CIS developer can protect an underlying system from too much activity while still providing recent (cached) data to reporting applications. In short, result-set caching enables alignment of the currency needs of the consumer with the volatility of the underlying data, whatever it may be.

Result-set caching may be applied to any view or service created in the Studio. By establishing a caching policy for a resource, the CIS developer instructs the Information Server where to cache the result-set and how often the cache is to be updated. The cached result-set may be stored in a local file or a specified database. If the resource being cached is a procedure (including scripts, java components, and web services), multiple result-sets will be cached, each corresponding to a unique set of input parameters.

During query planning, CIS’s query optimizer recognizes that a particular resource has been cached, and it modifies the query plan to use the cached result-set instead of the original resource definition. This is interesting for two reasons. First, entire sub-trees of a query plan can be eliminated because of the availability of a cache for a resource. Second, because the cache itself is a data source (either a file or a database), the query optimizer can apply further optimization techniques on the cached result-set.

It is worth re-iterating that any view or data service resource at any level may be cached, and that resource may still be used to build other higher-level resources. This granular approach to result-set caching allows the CIS developer to truly take advantage of data encapsulation, and balance the volatility of the data with the currency needs of the consumer.

The caching policy for a resource is established at design time and allows the data designer to dictate storage location, storage mode, update strategy and update timing as follows:

- **Storage Location** – A result-set cache may be stored in a file or a database. Files work very well for small and relatively static result-sets in development phase. For production use, databases serve as primary storage for larger and more volatile result-sets.

  CIS supports the following cache repositories: File; Greenplum; DB2; Microsoft SQL Server; MySQL; Netezza; Oracle; Sybase ASE; Sybase IQ; Teradata; Vertica.

  If a CIS instance is participating in a cluster, using a database for the cache storage allows a single result-set cache to be shared by all nodes in the cluster.

- **Storage Mode** – In multi-table caching mode, CIS uses several physical tables to store cached resources. Having one table to store each snapshot improves performance by allowing data to be deleted very quickly and enabling more efficient use of the source resource’s indexes at query execution time. Multi-table caching works very well for caches that contain substantial amounts of data to be retained for a long time, whereas the one table mode works better for caches that contain relatively little data as it’s easier to manage. Multi-table caching also removes conflicts between reading cached data while new data is being loaded into a new snapshot.

- **Update Strategy** – A result-set cache can be updated by performing a complete refresh or by incrementally modifying the cache with new changes. The complete refresh strategy is the most straightforward approach,
and it can be automatically applied to any cached resource. An incremental refresh strategy requires specific business logic, which the CIS developer can provide through scripting or programming. A resource can be cached individually or assigned to a cache policy where the cache refresh or other settings are managed at the policy level. Cache policies allow resources to be organized into groups so that cache schedules and other settings can apply to the group, rather than to individual resources.

- **Update Method** – CIS leverages target repository native load functions to load and refresh the cache. This native cache method improves cache load performance by bypassing CIS and loading data directly into the cache tables from the source tables. In addition to native cache, CIS uses multiple threads to load the cache in parallel.

- **Update Timing** – A result-set cache may be updated periodically based on the clock, or it may be updated opportunistically when the cached data exceeds a certain age. It is also possible to refresh a result-set cache manually only, which is very useful for caching reference tables that seldom change.

In addition to the multiple result-set caching options described above, CIS also supports Procedural Caching and Table Caching.

### 2.1.5 Security Components

CIS provides a complete set of capabilities to regulate who can perform operations on resources by authenticating users and enforcing user and group privileges. At run time, security regulates access to CIS resources and the data it returns. At design time, security establishes ownership of resources and enforces resource sharing and modification. Security is enforced for every request made to CIS and on all resources involved in a request.

**Security Privileges**

First, security is used to grant access to CIS, which is called authentication. Second, security is used to allow and enforce privileges on resources in CIS, which is called authorization. Resource privileges include:

- **Read, write** — At design time (i.e., when working in the Studio), these commands specify whether a user or group may see or modify a resource.

- **Select, insert, update, delete** — At run time, these commands specify whether a user or group may perform the given SQL command on a tabular resource (such as a view or column).

- **Execute** At run time, this command specifies whether a user or group may execute a procedure (such as a web service or a script), and also see the result of the procedure execution.

- **Grant** — At design time, this command specifies whether a user or group may grant privileges for a given resource to users or groups. The owner of a resource always has grant privileges.

Privileges may be granted to users (individuals) or to groups, which are collections of related users. Users and groups may be created within CIS, or the Information Server can leverage an external authentication system (such as Active Directory).

**Security Authentication**

Users attempting to access CIS must present credentials that identify themselves and allow CIS to authenticate them (usually in the form of a username and password). Users and groups may be defined in CIS or in an external
authentication system. When configured to use an external authentication system, CIS requests the external system to perform the authentication, which preserves the separation of roles.

Security Authorization

Whenever a user attempts to perform any action on a resource (such as executing a procedure), CIS checks the user's privileges, and the privileges of groups to which the user belongs, to determine whether the user should be allowed to perform the action. Authorization occurs on every resource involved in the operation, which means the user may need privileges on resources in addition to the resource they directly access. If a user does not have the required privilege to perform the action, an appropriate error is returned.

Executing Authentication and Authorization

When provisioning a data source, the administrator specifies whether the data source should be accessed with a single shared set of authentication credentials, or with the specific credentials of the current user. If a single set of credentials are specified, all authorization performed in the underlying data source is done using those credentials. If user-specific credentials are specified (which is also called “pass-through authentication”), then authentication to and authorization within the underlying data source are performed using the current CIS. To be clear, the security (authentication and authorization) is enforced by the underlying data source because it uses the same ID that it originally generated. CIS merely passes this ID from the requester down to the underlying data source.

Row-level Security

CIS supports row-level security using techniques commonly implemented by other systems. Specifically, the runtime system provides access to the current user, which can be used at design time to attach constraints to data being returned by a resource. The logic for performing row-level security can be arbitrarily complex, providing ultimate flexibility because the row restrictions are embedded in the data resource (such as a view).

For example, if you create a view that returns compensation information for employees (one row for each employee) it may be desirable to restrict access to these rows based on a management relationship. That is, only my manager and his or her managers (up to the company CEO) should be allowed to see my compensation information. You could place a row-level filter on the view’s definition, using the employee ID of the current user, and only rows that satisfy the manager constraint would be included in the result set.

For increased flexibility, CIS provides an option to nest policies together and allows the CIS administrator to determine how policies are enforced within that group of policies.

Token-based User Authentication

CIS includes a pluggable authentication module that allows administrators to integrate authentication services from any external system, including those based on tokens, such as Kerberos.

WS-Security Standards for Web services

Web service requests in CIS are processed using a pipeline that may include any meta-operation on the request. CIS includes WS-Security operations in this pipeline, and you can add your own operations as well. As WS-Security standards inevitably evolve, CIS will be able to modify the pipeline to accommodate the latest standards. The collection of operations that are performed in the pipeline is configurable by the administrator.

Data Encryption on the Wire
CIS offers encryption for web services data transport. This encryption is often accomplished by using SSL as the connection mechanism, and it can also be accomplished by adding granular encryption to the web service processing pipeline and JDBC access. Wire encryption for ODBC/ADO.NET connections is not currently available, but it is on the product roadmap to be included in a future version of CIS.

2.1.6 Data Delivery Components

Data delivery allows data consumers to request and receive data from CIS. An assortment of protocols and request types allow a variety of data consumers to access the views and services offered by the CIS.

CIS operates on a request-response paradigm. For example, a client makes a request and receives data as the response to that request. This paradigm is available to clients as SQL-based queries and service-based calls. Support of both SQL-based queries and service-based calls provides greater flexibility and reuse across a wider range of consumers including Business Intelligence tools and Enterprise Service Buses.

- **SQL-based queries** – SQL-based queries function the same way they do with a traditional relational database. A client establishes a connection to CIS using a driver, issues SQL queries, and reads results. This form of the request-response paradigm establishes a persistent connection through which multiple request-response cycles may occur. When the client is finished, the connection is closed. The predominant data model is rows and columns of values (tabular data). There are no practical size limits on the result sets. Result sets can be as large as necessary to answer the query because CIS streams the results through the persistent connection as long as the client continues to read the result. All of the usual metadata that is expected from a relational database is available through this type of connection.

For SQL-based queries, there are three client drivers available: JDBC, ODBC, and ADO.NET. All three drivers adhere to the latest standards for the particular driver model, and all three drivers accept SQL queries based on the SQL-92 and core SQL-99 standards. If a data client has the ability to connect to a database using one of these three mechanisms, they can successfully connect to and query CIS.

- **Service-based calls** – Data services follow a web service model. Specific service operations are made available, and their interfaces are documented through a WSDL, which the client uses to form a request. There is no persistent connection. Instead a call is made, the result is returned, and the session is closed. The predominant data model is XML documents for both the request and the response, which places practical limits on the size of the result set.

For service-based calls, several web service standards can be used. These include SOAP, REST, and raw-XML. The request-response cycle can occur over HTTP or JMS (a message bus). There are many additional options within each of these standards, and also within the CIS processing pipeline. All available options are outlined in the documentation.

2.1.7 Transaction Processing Components

CIS provides a transaction processing engine to support updates across multiple protocols and data sources, including flat files, relational databases, and web services.

Transactional Insert, Update, and Delete are supported against underlying data sources, whether or not those sources offer native support for these functions. CIS supports compensating processes for data sources that are not inherently transactional like flat files and web services.
Applying this technology, transactional views or services can be created that function directly with the underlying data sources or act upon other transactional views or procedures. These data services can even be created that are semi-transactional. Users need only apply transactional overhead where it is needed.

As a result of this best-of-breed approach to transaction protocols, CIS provides true distributed transaction management across heterogeneous data sources, executing queries with maximum efficiency while protecting against core system failures.

### 2.1.8 Triggers

Triggers cause activity within CIS at a given time or as the result of a given event. They are a mechanism to make something happen inside of a CIS instance independent of the normal request-response cycle that clients use. For example, it may be desirable to have a particular data source swapped to a different source at night during a maintenance window. A developer could write a script that employs the Admin API to perform the switch, and then use a trigger to invoke it at the appropriate time. Triggers are also used internally by CIS to implement some timed features in CIS. For example, a timed cache refresh is implemented using a trigger.

A trigger resource has two parts. The first part defines what condition causes the trigger to “fire” or execute. For example, a trigger could be set up to fire whenever a particular pre-defined system event occurs. An execution thread in CIS fires the trigger when appropriate conditions arise.

- **Timer** – The trigger will fire at the specified time. It can also be set up to repeat periodically.
- **System Event** – The trigger will fire whenever the given system event occurs. There is a pre-defined list of system events in CIS, all of which are documented.
- **User-Defined Event** – The trigger will fire whenever the given user-defined event occurs. User-defined events are posted by invoking a system procedure from inside CIS or through the public Admin API.

The second part of a trigger defines what action occurs as a result of the trigger firing. For example, a trigger could invoke a procedure. Once a trigger fires, its action is run in an independent execution thread. There are two main options for what a trigger does when it fires:

- **Execute Procedure** – This procedure allows the trigger to invoke any procedure available in CIS, which may include user-written procedures or system procedures.
- **Send an E-mail** – This procedure is actually a superset of the previous action because it begins with a procedure execution, followed by sending an e-mail with the results of the procedure execution. You can also skip the procedure execution and simply send an e-mail as a notification.

Between these two actions, you can see how triggers can be employed to perform almost any task in CIS. The trigger is simply the mechanism that initiates the task; the logic is implemented elsewhere.

### 2.2 CIS Development

#### 2.2.1 CIS Studio

CIS Studio is the primary modeling, view development, and resource management environment. The modeling environment presents a familiar resource-tree view of available physical data sources, a workspace area where queries are created and tested, and a data services area where CIS data services are published as views, data
services, or caches. Studio provides a data modeling environment that is similar in look and feel to other data modeling environments IT users have experienced. It masks a very complex and innovative multi-data source type modeling process.

Figure 3. CIS Studio

2.2.2 Relational Views

Relational views, or just views, are resources in CIS that integrate tabular data. A CIS view is conceptually equivalent to a traditional relational database view: it looks like a database table to the consumer. The view may be a final integration, which can be published for clients to query using SQL. Alternatively, a view may represent an intermediate result that can then be used with other data integrations (views or other resources). In CIS, views are created and published with CIS Studio.

Views in CIS are defined as a single SQL statement. CIS Studio development environment features a graphical view editor that allows you to create views using drag-and-drop techniques, and the resulting SQL statement is created automatically. You can also edit the SQL statement directly (instead of using the graphical editor).

CIS also lets you trace column and table lineage downstream and upstream. When making a change in a column or table in a view, users can visualize the data lineage for quick impact analysis. This powerful visualization answers questions such as where data originates, how it is used and transformed, and which views reference data in the current view.
The key distinction between traditional views in a relational database and those that you create in CIS is that CIS views can read data from multiple data sources. This behavior is implemented in CIS SQL by allowing you to reference any CIS resource in the FROM clause of the view’s SQL statement.

For efficient and effective analysis, the transformation editor enables the CIS developer to perform “any to any” and “one to many” transformation of data from any source structure (such as relational and hierarchical) to any target structure. The transformation happens in real time and supports mapping of multiple source resources to multiple target resources.

A view’s SQL conforms to the SQL-92 standard. However, it is also possible to provide the CIS optimizer with processing hints. These hints are specified as CIS-specific syntax because there is no syntax for hints as part of the SQL standard. The available hints are listed and described in the CIS reference documentation.

2.2.3 Data Services

Data services are web service operations that return data to the requestor (as opposed to implementing some sort of transactional business logic). Data services are used when you need to provide integrated data in the context of a service oriented architecture (SOA). Data services follow SOA standards and therefore have a high degree of interoperability with other components in the environment. CIS supports both Contract-First and Design-by-Example methods, as well as Service Component Architecture (SCA) and any XML standard.

![Figure 4. Contract-First Data Services](image)

The data services artifacts created in CIS studio are hosted in CIS. Data services can be provided in either SOAP or REST models. The transport can occur over HTTP or the Java Message Service (JMS) message bus.
many additional options within each of these standards and within the CIS service processing pipeline. All of the options are outlined in the documentation for client access.

### 2.2.4 Team Development

Team development occurs when multiple people work on the same set of resources within CIS, usually as part of a project development team. Team development carries certain challenges that are not present when one person is working on a set of resources. CIS provides functionality to assist with those challenges.

There are two situations where CIS team development features are important.

- **Shared Resources** – Shared Resources are useful when a team of people create and modify a shared set of resources over a period of time—usually during a project life cycle. This situation is similar to when a group of software developers work on a collection of source code files during the development of a piece of software. In both cases, a source code control system (SCCS) is helpful to keep track of the set of shared files, and to track changes and versions of the files. CIS integrates with popular SCCSs to allow resources to be checked in and out of the SCCS directly from the Studio user interface.

- **Versioning** – Versioning is valuable when two people accidentally modify the same resource in CIS at the same time. In this situation, it is important that neither person unknowingly overwrites the other person’s work when they save their changes. To prevent this, CIS always checks the version of a resource that is being edited against the version that has been saved, and CIS will warn the person trying to save the resource that a newer version exists in the metadata repository. This safety feature is usually unnecessary, especially with well-managed projects, but it is certainly a welcome feature when this undesirable situation occurs.

You can think of each resource in CIS as being analogous to a source code file in a software system. Whether it's a view, script, or any other resource, its implementation is the “source code” for that resource, and it is normally stored in the CIS metadata repository.

In order to get a resource in or out of a SCCS, the resource must exist in a file (because that's the unit that SCCS software uses). When you use Studio to “check in” a resource to a SCCS, Studio first externalizes the resource into a file using an XML-based representation of the resource, and then checks the XML file into the SCCS. Likewise, when you use Studio to “check out” a resource from an SCCS, Studio checks out the XML file from SCCS, and then imports the file into CIS.

CIS is natively integrated with Subversion (SVN) and Perforce SCCS. Cisco provides an open and supported API for integrating with any SCCS. Under normal circumstances, creating a new SCCS integration for CIS takes a couple days of effort, and the integration can then be deployed across the entire organization that has adopted a particular SCCS.

## 3. Cisco Data Virtualization Platform Options

Cisco Data Virtualization provides several optional products that extend CIS and complete the Cisco Data Virtualization Platform architecture.
3.1 Business Directory

Business Directory is an intuitive web-based tool for users to communicate, collaborate, and distribute information about data assets across various audiences within their organization. Business Directory allows business users and IT to better collaborate in the data development process. IT gains better control of data access while enabling a self-service directory of virtualized business data to empower users to get the data they need. Users can:

- **Search** – To find the data they are looking for. Simple Google-like searches to advanced parameterized searches make it easy to find the data you need.

- **Browse** – By data type, format, category, etc. to learn about all the available data.

- **Use** – Their favorite analytic and BI tools to query the desired data set from Cisco Information Server using the access information provided by Business Directory.

- **Categorize** – Data into system or user-defined groups to better organize large, diverse data sets.

- **Document** – By adding custom definitions, properties, links, and status codes to enrich the content about the data.

- **Collaborate** – With other users and IT through the use of comments to improve quality and usefulness of the data.

- **Personalize** – Data by watching and staying up-to-date on data resources that matter most to them.

- **Provision** – By registering additional Cisco Information Server instances to make more IT-curated data available to users.

Security profiles maintained in Business Directory ensure that users see only the data for which they are authorized.
3.2 CIS Discovery

CIS Discovery is an option in the Cisco Data Virtualization Platform.

CIS Discovery enables IT professionals to go beyond profiling to examine data, locate key entities, and reveal hidden connections in their enterprise data. You can use that knowledge to quickly build rich data models for data virtualization and other information initiatives. The models allow you to access and show live data, making it easier to validate business requirements with end users.

Once you have validated that the models match business requirements, you can transfer them to CIS with a push of a button to continue the software development life cycle. CIS applies the query optimization algorithms and techniques necessary for high-performance production operations.

CIS Discovery simplifies the upfront data modeling process to accelerate your data virtualization projects. CIS Discovery capabilities let you uncover data sources by locating relevant entities across multiple silos in your complex data environment. You can also discover relationships across disparate entities using formal keys and fuzzy matching. This provides greater insight into a complex data environment to accelerate data modeling and improves project planning and design.

CIS Discovery supports:

- **Visual Modeling** – Shape and refine data models in a rich graphical environment, enabling simple and flexible data modeling.
- **Pragmatic Bottom Up Data Modeling** – Build models from the ground up with metadata tied to actual data sources.

- **Easy-to-use Design Functions** – Capabilities includes transformation, filtering, parameter passing.

- **Agile SQL builder** – Studio’s unique SQL builder enables automatic SQL code generation using simple menu selection methods. Users do not need to know SQL.

- **Adaptable Rules Engine** – Provide hints to the relationship discovery process to improve matching accuracy.

- **Seamless Integration** – Transfer models to CIS with one click, enabling fast and seamless deployment.

3.3 **CIS Active Cluster Option**

The Active Cluster Option supports substantial scaling of CIS deployments while ensuring continuous availability to fulfill service level agreements. CIS Clustering allows multiple instances of CIS servers to work together to handle user requests. Each CIS instance in the cluster is called a "node." Clustering configurations are flexible based on reliability, availability and scalability requirements.

- **Scalability** – To achieve scalability (increased capacity to service requests), additional nodes running with their own computing resources (CPU cores, RAM memory, and hard disk storage) are added to the cluster.
as needed. For purposes of scalability, it is not necessary for each node to run on a completely separate server as long as there are sufficient computing resources for multiple nodes running on the same server.

- **High Availability** – To achieve high availability, a cluster is created with nodes running on separate servers. Spanning multiple servers protects the cluster from server hardware failures. As insurance, the capacity of the cluster must have enough processing headroom to accommodate the loss of one or more nodes so that all requests can continue to be serviced when a server failure occurs.

The CIS clustering architecture is peer-to-peer. No single node is more important than any other node in the cluster, and there can be up to sixteen nodes in a single cluster. With its active approach, all instances of CIS in the cluster are active at all times.

### 3.3.1 Distributing Queries Across a Cluster

The usual practice for distributing client requests is to install a load balancer product in front of the CIS cluster (just as you would for a cluster of application servers). The load balancer distributes requests among the individual nodes according to a pre-configured policy (such as round robin).

If a client request is a simple request-response (stateless) request (such as a web service invocation), then subsequent requests from the same client can be distributed to different nodes by the load balancer. Alternatively, if a client request creates a persistent (stateful) connection, as is the case with JDBC/ODBC, then the connection remains bound to a specific node in the cluster for the duration of the session. In this situation, if the node to which the client holds a persistent connection fails, the client would receive an error and take steps to re-establish a connection to the cluster, while the load balancer distributes the request to another node.

### 3.3.2 Metadata Synchronization

The cluster exists as a peer-to-peer network of CIS nodes. Each node holds its own copy of the metadata repository, but each node’s metadata repository is continuously synchronized with all other nodes in the cluster. If a metadata change is made to one node, that change is immediately propagated around the cluster to all other nodes.

Given this operational model, publishing new CIS resources to a CIS cluster is a simple matter of publishing the resources to a single node. The cluster takes care of propagating the updates to all other nodes.

### 3.3.3 Caching within a Cluster

A result-set cache in CIS may be stored in a local file, a relational database, or in memory. If a result-set cache is stored in a local file, then caching in a cluster works no differently than caching in a single CIS instance. In this case, each node creates and maintains its own copy of the result-set cache according to the caching policy of the resource. This may or may not be the desired behavior, so a second option exists as well.

If the result-set cache is stored in a relational database, the nodes in the cluster can share a single copy of the result-set cache. This arrangement provides better cache coherency and makes more efficient use of resources. When the cluster is sharing a single cache, the nodes must coordinate the cache update procedure so that only a single node performs the update. This coordination is done through a bidding and assignment algorithm that is common in peer-to-peer architectures, and it ensures that only a single node performs the job of updating a shared cache.
3.4 CIS Adapters

CIS Adapters simplify and accelerate high-performance access to a wide range of data sources, including popular enterprise applications, relational and multi-dimensional data sources, “big data” stores, and web content.

Beyond simple metadata (schema) mapping and lowest common denominator access standards such as ODBC or JDBC offered by other vendors, CIS Adapters provide performance optimization capabilities and development acceleration. Key capabilities include:

- **Certified Vendor-Specific Connectivity API** – Access proprietary data using vendor-approved access methods
- **Interactive Metadata (Schema) Mapping** – Enable fast and accurate modeling
- **Standard SQL to Vendor-specific SQL Resolution** – Ensure precise SQL translation and execution
- **Statistical Analysis and Cardinality Estimation** – Accumulate critical metrics used by the CIS cost-based optimizer
- **Capability Introspection and Coordination** – Determine configurations, functionality, and parameters required to enable optimal performance
- **Vendor-specific Engineered Functions** – Supercharge performance beyond vendor’s standard capabilities
- **Relational Representation of Source Data** – Standardize data structures to accelerate development
- **Business Canonical Abstraction of Source Data** – Standardize data semantics to simplify development

For SAP, Oracle, Siebel, and Salesforce.com applications, CIS Adapters provide far more than simple access. They add value by delivering hundreds of the most common application objects such as invoices, shipments, customers, employees, and more unusual object such as pre-built views or web services. Supported Enterprise Applications include Oracle E-business Suite, SAP R/3, Siebel, and Salesforce.com.

For SAP BW, CIS Adapter exposes this multi-dimensional source in a relational paradigm which allows the broader SQL-skilled development community to leverage this important data in a far wider set of use cases and analysis tools.

For Cloud and business-to-business (B2B) applications, and public web content, CIS Adapters provide real-time access to the underlying database behind web pages and forms, including data tables, data structures and data types—all through the existing web interface with no need for direct database or API access.

For common enterprise data sources, CIS Adapters intelligently evaluate and leverage underlying data source capabilities to ensure optimal federated query performance. Supported data sources include Greenplum, Hadoop/Hive, HP Vertica, IBM DB2, IBM Informix, IBM Netezza, Microsoft Access, Microsoft Excel, Microsoft SQL Server, Oracle Database, Oracle MySQL, SAP Sybase, SAP Sybase IQ and Teradata.

3.5 Deployment Manager

Deployment manager is a comprehensive tool for enabling fast and easy CIS deployments at the enterprise level. By automating the migration of artifacts, configurations, and settings from one CIS instance to another, Deployment
Manager minimizes deployment risks while promoting enterprise software implementation best practices. For example, promoting artifacts from the development environment to staging to production in a large data virtualization environment would generally require such a tool. The tool’s thorough documentation and logging also improves the migration process.

Deployment Manager supports:

- **Resource Migration** – To allow administrators and users to transfer or promote (create/update/delete) artifacts from one CIS instance to another.

- **Cache Setting Migration** – To transfer or promote cache table names, caching methods, refresh method, cache policies, and cache schedules.

- **Remote Server API Calls** – To allow administrators to execute remove API calls on CIS instances to execute custom operations such as update server configurations (for example, enabling/disabling triggers).

- **User/Group Migration** – To transfer or promote user/group IDs, security profile, and other information.

3.6 CIS Monitor

CIS Monitor provides a comprehensive, real-time view of your Cisco Data Virtualization Platform environment. Whether the environment is a single CIS or a cluster of instances, CIS Monitor displays all the system health indicators necessary to assess current conditions. If processes slow down or operations fail, the IT operations staff can use these insights to guide the actions required to maintain agreed service levels.
CIS Monitor is designed specifically for CIS, so no additional systems integration or custom set up is required. Features include:

- **Intuitive Displays** – Graphical displays are easy to read and interpret. You can quickly assess the state of a data virtualization environment by clicking through a few intuitive displays.

- **Configurable Alert** – Turn on and off various alerts and set tolerance gradients to meet desired levels of services.

- **Logging and Audit** – Twenty four hours of history is kept for review, investigation, and reporting.

- **Request Search** – Easily find requests to quickly diagnose problems and trace the sources of trouble.

### 4. Cisco Data Virtualization Systems Management

System management keeps the Cisco Data Virtualization Platform running well, so the views and resources are continuously available to the consumer and service level agreements are met. The focus of system management is the configuration, monitoring, and maintenance of the running CIS, rather than design and development activities.
4.1 Cisco Data Virtualization Platform System Management Tools

Cisco provides five tools that combine to support effective management of the Cisco Data Virtualization systems:

- **CIS Manager** – This browser-based set of consoles facilitates configuration of, and visibility into, a running CIS instance. Among other things, the user can view and modify server settings, see active client sessions, and investigate running queries. The Manager is the primary tool for managing a single CIS instance.

- **Monitor** – Monitor is the primary tool for monitoring CIS instances. This browser-based client presents a dynamic view of a running CIS environment including all nodes and clusters. You can see collections of clients and their level of activity, monitor memory and CPU usage of the host machines, and display dashboards that include overall health indicators.

- **SNMP** – Simple Network Management Protocol (SNMP) is a standard used in many network operations centers (NOCs) to monitor the health and activity of network components. Although it is generally more focused on hardware assets than software assets, it is still important for software servers to participate in an SNMP monitoring strategy. CIS publishes a standard SNMP MIB that defines numerous available “traps” that can be monitored by the NOC dashboards and personnel.

- **Administration API** – Cisco provides a public, documented API for programmatically manipulating all metadata inside a CIS instance. Although the Admin API serves a broader scope than system management, it includes many APIs that are useful for system management. For example, all CIS configuration variables...
can be read or changed using the Admin API. Customers who wish to create their own custom system management strategies for CIS can integrate their scripts with the Admin API.

- **Manager Pane in CIS Studio** – Although CIS Studio is primarily a development tool for data integration resources, it also has integrated manager functionality that is similar to the browser-based CIS Manager. Most of the functionality found in the CIS Studio manager pane has been ported over to the CIS Manager, and this trend continues. Eventually, the CIS Studio manager pane may be deprecated and then removed to simplify CIS Studio.

## 5. Conclusion

The Cisco Data Virtualization Platform is a leading edge, proven approach for meeting business goals by federating data across silos and simplifying real-time access to critical information.

Designed as a lower cost, more agile data integration approach that overcomes data complexity and silos, Cisco data virtualization helps businesses increase profitability, reduce costs and time-to-market, manage risk and compliance, and increase agility. Cisco’s Data Virtualization Platform combines CIS with multiple options to provide a complete data virtualization development and deployment resource.

### Why Cisco Services

Realize the full business value of your technology investments with smart, personalized services from Cisco and our partners. Backed by deep networking expertise and a broad ecosystem of partners, Cisco Services enable you to successfully plan, build, and run your network as a powerful business platform. Whether you are looking to quickly seize new opportunities to meet rising customer expectations, improve operational efficiency to lower costs, mitigate risk, or accelerate growth, we have a service that can help you.

### For More Information