

**Denodo 9.0 Proof of Concept**

**Business & Technical Validation**



**Denodo POC: Business & Technical Validation**

# **Introduction**

A **Proof of Concept** (POC) demonstrates the feasibility of a Data Virtualization strategy. A POC with Denodo typically involves installation, configuration, and testing of Denodo’s software within the customer environment to validate a set of business, functional, technical and operational goals. A POC is broken down into these areas to meet the needs of the customer:

1. Technical and Functional Validation
2. Sizing of the Deployment needed so that configuration options and budget estimates can be provided
3. Services Assessment
4. Return on Investment (ROI) calculations

# **How to use this document**

This document is a template designed to help gather the information required for a successful POC with Denodo. It is organized in three different sections, described below:

## Business Use Case Requirements

It includes four areas:

* Overview of the business scenario, initiative, goals, etc.
* Description of the current situation
  + How are data integration projects currently executed?
  + What are the challenges faced by that approach?
  + List the success criteria for validation of the new DV-based solution. This information is used for estimating the ROI of implementing DV
* Usage Expectations for the Denodo deployment - needed for a sizing estimate of the production deployment

## Technical Evaluation Requirements

This section describes the functional tests for individual test cases. Each test case represents a slice of the larger use cases providing different scenarios. Each test case includes six sections, providing enough detail for a data engineer to start developing:

* Overview
  + Brief description of the test case
* Data sources
  + Details on data sources involved in the test case:
    - Nature of the data source: type, vendor, version, access method, etc.
    - Data source details: host, credentials, schemas, etc.
  + Tables or API calls containing the data needed for the use case
* Combination & transformation logic
  + Description of queries, pseudo-code or SQL.
  + How is the data combined and transformed to obtain the desired results?
    - This is especially important for the columns used to combine different data sources. For example: customer data will be joined between the EDW and the Finance system based on the customer SSN. Need to transform one side to remove the “-“
  + What are the data volume estimations? E.g. customer table has 50M rows
* Consumption
  + How will data be consumed by the end user?
  + This includes third-party client applications (e.g. dashboard in Tableau), usage of the Denodo Data Catalog and other access protocols (RESTful services, JDBC, etc.)
* Success Criteria
  + Description of success criteria in terms of productivity, features, performance, etc.
* Other requirements
  + Any other technical feature taken into account in this use case: cache settings, security requirements, data governance, etc.

## Common Data Virtualization Features - Checklist

In addition to the use cases, it may be interesting to check the functionality of specific features. This is usually for common operation tasks not tied to a specific test case: migrating code between environments, monitoring, clustering, etc. To simplify the scope definition of the POC in these areas, this section provides a list of the most widely used features in the Denodo Data Virtualization platform, organized into categories.

# **Annex I: POC Examples**

These examples highlight what most customers test during their evaluations. It also provides guidelines toward the level of detail required for a successful POC.

# **Annex II: Sizing Guide**

This section guides the estimation process for sizing deployments, based on the complexity of the scenario and the expected concurrency levels at peak loads.

# **Annex III: Services Assessment**

This section guides the estimation process on the services and organization needed to successfully deploy, operate and manage the Use Cases defined by the Customer to meet their milestones and success metrics. You will now have the knowledge to decide the distribution of using internal, partner and/or Denodo resources. The Assessment will document the organization you need to have in place, the skill sets and training required so you can properly budget and prioritize resources. If you want to outsource any of the work to a partner, you will have a scope you can provide them. If you want to utilize Denodo resources you will be aware of what services are included and what additional services would need to be added to your contract.

# **Annex IV: ROI Guide**

Based on the business requirements of the scenario and the current techniques used in your company to currently address similar requirements, this annex will help you calculate the ROI of a virtualized solution.

# **Customer Details and Org Chart**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Customer Name**  Name of the company | | | | | |
| **Contact Name** | **Title** | **Role (Exec Sponsor, Bus or Tech Sponsor, Arch, Dev. doing POC, etc.** | **Phone** | **Email** | **Location (needed for time zone coord.)** |
| Wael Elsaed | **Mr** | **Data Science Senior Expert** | **+201118002778** | **wael.elsaed@eand.com.eg** | **Cairo, Egypt** |
|  |  |  |  |  |  |

# **Denodo Contacts**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Contact Name** | **Title** | **Role (Sales RVP, Sales Exec, Sales Eng)** | **Phone** | **Email** | **Location (needed for time zone coord.)** |
| Jad Jabbour | **Mr** | **Sales Director** | **+971562564553** | [**jjabbour@denodo.com**](mailto:jjabbour@denodo.com) | **UAE-Dubai** |
| Rami Jarbou | **Mr** | **Senior Sales Engineer** | **+966503128711** | [**rjarbou@denodo.com**](mailto:rjarbou@denodo.com) | **KSA-Riyadh** |

# **Business Requirements**

Please fill the sections below with information about the scenario and use cases. If there is more than one use case, just copy and paste this section as many times as needed.

# **Business Use Case**

|  |
| --- |
| **Use Case Description**    Implement a GenAI-powered application layer that sits on top of existing data marts to enable natural language querying, data correlation across domains, and uncover hidden insights without manual SQL or BI dashboard creation. |
| **Challenges with Current Approach**  - Manual query writing requires technical expertise.  - Limited ability to correlate data across multiple marts.  - Insights discovery is slow and reactive.  - High dependency on BI teams for ad-hoc analysis.  - Static dashboards fail to answer dynamic business questions. |
| **Success criteria / Business Outcome and Next Steps**    - Enable business users to query data using natural language.  - Reduce time-to-insight.  - Improve decision-making with AI-driven correlations and anomaly detection.  - **Next Steps:** Pilot GenAI layer on one data mart → Expand to all marts → Integrate predictive analytics. |
| **Current Effort Estimation**  What would be the **timeline** for the development of this use case **without** Denodo (in days): ~30–60 days.  How many people to **develop** the project: 4–6 people (Data Engineers, AI Engineers, UI/UX).  How many people to **operate** the project: 2–3 people (Support, Monitoring). |
| **Future Usage Estimation**  How many **users** will use the system? 150–200 (Business Analysts, Managers).  What will be the **peak load** (# concurrent queries): ~50–75 concurrent queries during business hours. |

# **Technical Evaluation Requirements**

Please fill the sections below with information about the test cases to cover during this POC. If there is more than one test case where Denodo will be used, just copy and paste that section as many times as needed

# **Technical Test Case #1**

|  |
| --- |
| **Natural Language Querying & Insight Discovery** |
| **Overview**  Enable business users to query data marts using natural language and receive accurate results along with AI-driven insights (correlations, anomalies, trends) without writing SQL or relying on static dashboards. |
| **Data Sources**  **CVM Data Marts:**   * transactions, revenue, product details * demographics, churn indicators * campaign spend, engagement metrics |
| **Combination & Transformation logic**  **Step 1:** Convert natural language query → semantic SQL using LLM.  **Step 2:** Join relevant tables across marts based on entity relationships (e.g., subscriber ID, campaign ID).  **Step 3:** Apply aggregation, filtering, and enrichment logic for insights (e.g., churn prediction, correlation analysis). |
| **Output**   * Tabular query results. * Textual summary query results. * AI-generated insight cards (e.g., “Region X shows 15% higher churn correlated with low engagement”). * AI-generated recommendations * Visualizations (charts, trend lines). |
| **Others**  Cache, Security, Data Governance, Monitoring, Development & Deployment, etc.   * **Cache:** Frequently used queries cached for faster response. * **Security:** Role-based access control; sensitive fields masked. * **Data Governance:** Compliance with existing policies; audit logs for queries and AI recommendations. * **Monitoring:** Performance dashboards; alerts for latency or failures. * **Development & Deployment:** Containerized GenAI layer; CI/CD pipeline for updates. |
| **Success Criteria**   * **Accuracy:** ≥95% correct interpretation of natural language queries. * **Performance:** Response time <3 seconds for simple queries; <5 seconds for correlated insights. * **Insights:** At least 3 actionable insights per test scenario validated by SMEs. * **Scalability:** Handle 50 concurrent queries without degradation. * **Compliance:** 100% adherence to security and governance policies. |

# **Technical Test Case #2**

|  |
| --- |
| **<Name>** |
| **Overview** |
| **Data Sources** |
| **Combination & Transformation logic** |
| **Output** |
| **Others**  Cache, Security, Data Governance, Monitoring, Development & Deployment, etc. |
| **Success Criteria** |

# **Features Checklist**

For some scenarios you may prefer to evaluate individually some additional features of your interest. Please identify those features in the tables below and add your comments accordingly.

## Data Source Connectivity

|  |  |  |
| --- | --- | --- |
| **Feature** | **To validate (yes/no)** | **Comments** |
| Traditional relational databases (Oracle, SQLServer, DB2, MySQL, Postgres, etc.) | Yes | Core requirement: GenAI layer must query structured data from existing relational marts. |
| Data Warehouse databases (Netezza, Teradata, Vertica, Greenplum, etc.) | Yes | To validate connectivity and query optimization for large-scale analytical workloads. |
| In-memory databases (HANA, Oracle In-Memory, etc.) | Optional | Would be useful for real-time insights. |
| Cloud databases (Redshift, Snowflake, AWS RDS, etc.) | Optional | For hybrid/cloud architecture; ensure secure integration and performance if needed in the future. |
| SQL-on-Hadoop (Hive, SparkSQL, Impala, Presto, etc.) | Yes | Big data sources are part of the POC scope. |
| Delimited files, logs | Yes | Needed for semi-structured data ingestion and correlation with marts. |
| Excel | Yes | To validate upload and query capability as business users often rely on Excel. |
| Web APIs and SaaS (SFDC, Google APIs, Workday, etc.) | Optional | To validate integration for external data enrichment and real-time queries if needed in the future. |
| SAP ECC | No | ERP data is not required. |
| Multidimensional cubes (SAP BW, Essbase, MS SSAS, etc.) | Optional | If OLAP cubes are available and part of reporting requirements. |
| Hadoop HDFS files, Amazon S3 files, GCP Cloud Storage, Parquet, Delta and Iceberg files | Yes | To validate ability to query and correlate large-scale unstructured/semi-structured data. |
| NoSQL data bases (MongoDB, Cassandra, Neo4j, etc.) | Optional | Graph or document data might be needed for advanced insights. |
| LDAP | Yes | Required for authentication and role-based access control. |
| SDK for custom adapters (in Java) | Yes | To validate extensibility for future custom connectors. |
| Others (explain which ones) | Yes | AI/ML models for insight generation; to validate integration with GenAI layer. |

## Modeling

|  |  |  |
| --- | --- | --- |
| **Feature** | **To validate (yes/no)** | **Comments** |
| Relational modeling using GUI wizards (Joins, Unions, Aggregations, Projections, Flattening, etc.) | Yes | Important for simplifying logical model creation without coding; ensures business-friendly design. |
| VQL (Virtual Query Language) direct shell commands | Yes | Needed for advanced users and automation; to validate flexibility beyond GUI. |
| Extended Relational Model: native support for complex data types (register and arrays) to model hierarchical structures like web service APIs | Yes | Critical for integrating APIs and semi-structured data sources; to ensure GenAI layer can handle JSON/XML. |
| Support for multiple logical databases, and to use views from different ones in the definition of logical models | Yes | Required for cross-mart correlation and unified semantic layer. |
| Definition of associations between views of the model | Yes | Enables semantic relationships for GenAI to infer joins automatically. |
| Top-down design based on Interface Views | Optional | Useful for enterprise architecture alignment. |
| Import of models built in external modeling tools (ERStudio, Erwin, Power Builder, etc.) | Optional | Existing models might need to be reused; reduces migration effort. |
| Data transformation library: string manipulations, date manipulation, math, trigonometry, etc. | Yes | Essential for query enrichment and AI-driven calculations. |
| SDK for custom functions (UDFs) | Yes | Needed for extending functionality and embedding custom business logic. |
| Stored Procedure Support (VQL, Java) | Optional | If procedural logic is required for complex transformations. |
| Textual similarity (fuzzy join) functions | Yes | Important for matching inconsistent data (e.g., customer names across systems). |
| Geo-spatial functions | Optional | If location-based insights are part of the POC scope. |

## Data Delivery

|  |  |  |
| --- | --- | --- |
| **Feature** | **To validate (yes/no)** | **Comments** |
| SQL-based access via JDBC, ODBC and ADO.NET | Yes | Critical for enabling BI tools and external applications to query the GenAI layer seamlessly. |
| Access from reporting and BI tools (Tableau, PowerBI, BO, OBIEE, Microstrategy, etc.) | Yes | Must validate compatibility for direct integration with enterprise BI platforms. |
| Publishing as web services: SOAP, RESTful, OData | Yes | Required for exposing data and insights to external systems and microservices. |
| IDU operations via SQL, SOAP and REST | Optional | If Insert/Update/Delete operations are needed for transactional scenarios. |
| Access via JMS queues (Sonic, Active MQ, IBM’s, etc.) | Optional | Useful for event-driven architectures; if real-time messaging is in scope. |
| Batch delivery to files and databases | Yes | Important for scheduled data exports and integration with downstream systems. |

# **Denodo Data Catalog**

|  |  |  |
| --- | --- | --- |
| **Feature** | **To validate (yes/no)** | **Comments** |
| Complete web catalog of Denodo’s data model | Yes | Essential for browsing and understanding available views and logical models. |
| View of relationships, data lineage and model descriptions | Yes | Critical for governance and transparency; helps users trace data origins. |
| Classification of the views in tags and business categories that can be browsed in the form of a directory | Yes | Improves discoverability for business users; supports semantic organization. |
| Advanced metadata search to quickly find a model | Yes | Needed for efficiency; to validate ability to locate views by name, tags, or attributes. |
| Content search using indexed data (Lucene or ElasticSearch) | Optional | Useful for large-scale deployments; if full-text search is required in the future. |
| Easy to use web-based query wizard for non-technical users | Yes | Key for adoption; ensures business users can build queries without SQL knowledge. |
| Easy to use data preparation wizard | Yes | Important for self-service data wrangling and enrichment before analysis. |
| Assisted query using natural language (GenAI and NLP) | Yes | Core POC requirement; to validate GenAI integration for NLQ. |
| Smart query recommendations (based on DC query history) | Optional | Useful for productivity; to validate whenever personalization is needed. |
| Export results to the user desktop | Yes | Must support CSV/Excel export for offline analysis. |
| Definition of personal reports (“My queries”) | Yes | Enables user personalization and quick access to frequently used queries. |
| Quickly share data with other users | Yes | Important for collaboration; to validate secure sharing options. |

# **Security**

|  |  |  |
| --- | --- | --- |
| **Feature** | **To validate (yes/no)** | **Comments** |
| Integration with LDAP / Active Directory | Yes | Required for enterprise identity management and single sign-on. |
| User and Role based authorization | Yes | Critical for enforcing access control at the view and query level. |
| Kerberos integration | Optional | To be validated if Kerberos is part of e&’s authentication strategy. |
| Fine grained authorization: masking, row and column restrictions | Yes | Needed for compliance and secure data exposure (PII, PHI). |
| Attributed based access control based on session attributes | Yes | Important for dynamic security policies based on user context. |
| Global security policies based on tags (PII, PHI, GDPR, etc. data) | Yes | Ensures compliance with regulatory frameworks and data classification. |
| Encrypted data in motion via TLSv1.2 | Yes | Mandatory for secure communication between components. |
| Advanced authentication for web services: SAML, OAuth2, SPNEGO | Yes | Required for secure API access and integration with enterprise identity providers. |
| Credentials pass-through to data sources | Yes | Needed for maintaining source-level security and audit trails. |

# **Optimizer and Execution Engine**

|  |  |  |
| --- | --- | --- |
| **Feature** | **To validate (yes/no)** | **Comments** |
| Modern Cost Based Optimizer (CBO) | Yes | Critical for efficient query execution and minimizing resource usage. |
| Advanced query push-down capabilities | Yes | Ensures heavy processing is delegated to source systems for better performance. |
| Complex execution plan optimizations to deal with large result sets (e.g. partial aggregation pushdown, join under union pushdown, join pruning, etc.) | Yes | Needed for large result sets and complex queries; to validate optimizer intelligence. |
| Automatic simplification of queries based on metadata rules (e.g. virtual partitions) | Yes | Reduces query complexity and improves performance automatically. |
| On-demand data movement to other sources (as temp tables) for query execution | Optional | Useful for performance tuning in hybrid environments; to be validated if required. |
| Support for tables that exist in more than one data source (alternative sources) | Yes | Important for redundancy and failover; ensures flexibility in query execution. |
| Option to add manual hints for individual query executions | Optional | To validate if manual tuning is needed for edge cases. |
| Detailed execution plan available for analysis | Yes | Essential for troubleshooting and performance optimization. |
| Resource manager to configure throttling restrictions based on rules (user, role, client, source IP, time of day, etc.) | Yes | Needed for governance and preventing resource contention. |
| Smart query acceleration using summaries and AI recommendation wizard | Yes | Aligns with GenAI goals; accelerates queries using precomputed summaries and AI suggestions. |
| MPP Presto engine for query acceleration (data lakes, memory intensive queries) | Optional | To validate if large-scale data lake queries are part of the POC scope. |

# **Caching**

|  |  |  |
| --- | --- | --- |
| **Feature** | **To validate (yes/no)** | **Comments** |
| Ability to easily enable cache for a view based on configuration | Yes | Critical for performance optimization; should be simple and configurable per view. |
| Support for different types of databases as caching engines (in-memory, MPP, cloud, etc.) | Yes | Needed for flexibility in choosing cache backend based on workload and infrastructure. |
| Advance controls to load/refresh data | Yes | Important for scheduling, prioritizing, and controlling refresh strategies. |
| Cache co-location with existing data to maximize query push-down | Optional | To validate if co-location improves performance in hybrid environments if required. |
| Support for incremental scheduled updates on cached data | Yes | Essential for reducing refresh overhead and maintaining near real-time accuracy. |
| Support to refresh data based on external messages (JMS) | Optional | Useful for event-driven refresh; to validate if real-time triggers are needed. |
| Support for hybrid queries: cached content + real time “delta increments” | Yes | Key for scenarios requiring up-to-date data without full refresh. |
| Ability to cache incomplete data sources (e.g. web services with input parameters) | Yes | Important for optimizing performance when dealing with parameterized APIs. |

# **Hadoop Integration**

|  |  |  |
| --- | --- | --- |
| **Feature** | **To validate (yes/no)** | **Comments** |
| Hadoop as a data source (Hive, Impala, Presto, SparkSQL) | Yes | Critical for querying big data environments; to validate connectivity and performance. |
| DenodoConnect adapters for HDFS (Avro, Parquet, Map files, etc.) and HBase | Yes | Needed for accessing semi-structured data formats commonly used in Hadoop ecosystems. |
| Hadoop as cache system (Hive, Impala, Presto, SparkSQL) | Optional | To validate if Hadoop-based caching improves performance for large-scale queries. |
| Leverage Hadoop as cache to load data into Hadoop without losing governance | Yes | Important for maintaining security and governance while using Hadoop for caching. |
| Massive Parallel Process Acceleration using SQL-on-Hadoop engines | Yes | Required for accelerating queries on large datasets; to validate scalability. |
| Massive Parallel Process Acceleration co-location with existing data to maximize query push-down | Optional | Useful for performance optimization; validate if co-location is feasible in e& architecture. |

# **Metadata Governance**

|  |  |  |
| --- | --- | --- |
| **Feature** | **To validate (yes/no)** | **Comments** |
| Data lineage that traces each column down to the base views and data sources, and displays intermediate transformations | Yes | Critical for transparency and compliance; ensures full traceability for audits and impact analysis. |
| Detection of changes in the sources | Yes | Important for maintaining data integrity and alerting when schema or data changes occur. |
| Impact analysis of metadata modifications in derived views | Yes | Needed to prevent breaking downstream dependencies when changes are made. |
| Ability to add descriptions to views and columns | Yes | Improves usability and governance; supports business-friendly documentation. |
| Import native metadata from sources: indexes, PKs, FKs, statistics, descriptions, etc. | Yes | Essential for query optimization and accurate modeling. |
| Native Integration with version control systems (SVN, TFS and Git) for collaborative development | Yes | Required for DevOps alignment and team-based development workflows. |
| Integration with 3rd party catalogs (Collibra, Alation, Informatica Data Catalog, IBM Business Glossary, Microsoft Purview, Data.World, etc.) | Yes | Important for enterprise data governance and catalog synchronization. |

# **Denodo Scheduler**

|  |  |  |
| --- | --- | --- |
| **Feature** | **To validate (yes/no)** | **Comments** |
| Scheduler for time-based planned execution of batch jobs | Yes | Essential for automating recurring tasks like data refresh and exports. |
| Configurable time-based cache refresh | Yes | Critical for maintaining data freshness without manual intervention. |
| Data indexing and crawling for keyword-based search | Yes | Needed for fast metadata and content search across large datasets. |
| Planned metadata management tasks: source changes, statistics gathering, cache maintenance, etc. | Yes | Important for governance and system health; to validate automation capabilities. |
| Export data to target database and to file dumps | Yes | Required for integration with downstream systems and archival processes. |
| Email notifications | Yes | Useful for alerting stakeholders about job completion, failures, or anomalies. |
| Support for configurable retries and job dependencies | Yes | Ensures robust execution of workflows and recovery from transient failures. |

# **Denodo’s Solution Manager**

|  |  |  |
| --- | --- | --- |
| **Feature** | **To validate (yes/no)** | **Comments** |
| Global web console for management of the full Denodo deployment | Yes | Essential for centralized administration and monitoring across environments. |
| Centralized license manager for clients and servers with license usage log | Yes | Important for compliance and tracking license utilization. |
| Centralized distribution of software updates to clients | Yes | Needed for streamlined patching and version consistency across nodes. |
| Centralized management of logs generated by the different servers in the environment | Yes | Critical for troubleshooting, auditing, and operational visibility. |
| Control of migrations/promotions directly from the web UI | Yes | Simplifies DevOps workflows; reduces manual intervention for environment promotions. |
| Configuration and management of environment dependent properties (URLs, users, passwords, etc.) | Yes | Required for smooth migration between DEV, QA, and PROD environments. |
| Definition of migration strategy for each environment, including cluster and load balancer control | Yes | Ensures high availability and scalability during deployments. |
| Graphical definition of metadata promotions (revisions) | Yes | Improves usability for managing revisions and promoting changes visually. |
| API for integration with external lifecycle management systems like Jenkins | Yes | Critical for CI/CD automation and integration with enterprise DevOps pipelines. |

# **Monitoring and Auditing**

|  |  |  |
| --- | --- | --- |
| **Feature** | **To validate (yes/no)** | **Comments** |
| Integrated web-based monitoring tool: CPU, memory, sessions, queries, cache, etc. | Yes | Essential for real-time health checks and performance monitoring of the Denodo environment. |
| Support to log to disc (or database) the activity of the system for later analysis and integration with log-based analysis tools like Splunk | Yes | Critical for audit trails and integration with enterprise observability platforms. |
| Ability to load logs to “go back in time” and graphically represent the status of the system in the past | Yes | Useful for root cause analysis and historical performance reviews. |
| Auditing of all queries and activities on the system, including user, source IP, time, query, tables involved, etc. | Yes | Mandatory for compliance, security, and governance requirements. |
| Log of logging attempts (successful and erroneous) | Yes | Important for security monitoring and detecting unauthorized access attempts. |
| Monitoring data is exposed via SNMP and JMX for integration with external monitoring tools | Yes | Needed for enterprise integration with existing monitoring frameworks. |
| Denodo Dashboard (Finops) based on Apache Superset container | Optional | To validate if financial operations and cost optimization dashboards are required for e& use case. |

# **Deployments**

|  |  |  |
| --- | --- | --- |
| **Feature** | **To validate (yes/no)** | **Comments** |
| Support for clustered and HA deployments with a load balancer | Yes | Critical for high availability and fault tolerance in production environments. |
| Deployment in containers (e.g. Docker) | Yes | Needed for modern DevOps workflows and portability across environments. |
| Deployments in cloud (AWS, Azure, GCP, Alibaba) | Yes | Essential for hybrid or full cloud strategies; validates multi-cloud compatibility. |
| Multi-tiered (Denodo to Denodo) deployments | Yes | Important for distributed architectures and scalability across regions or business units. |

**Annex 1: POC Example**

# **Business Requirements**

# **Business Use Cases**

|  |
| --- |
| **Use Case Description**  We would like to test Data Virtualization capabilities to provide a more agile method to access and use data from our heterogeneous data ecosystem. The architecture has to enable use of data across diverse source types in an efficient and simple way, to achieve the following goals:   1. Faster turnaround for BI and data science projects to reduce the “Time to Data” 2. Better Competitive & Operational Business Intelligence that uses a broader range of sources that are currently unavailable 3. Definition of a business semantic model that is shared across multiple reporting tools that centralizes the definition of core metrics, while presenting an easy-to-use schema to end users 4. Enable certain degree of Self-Service to reduces the dependency on IT and eliminate the need for “Shadow IT” |
| **Challenges with Current Approach**  Current data ecosystem is fragmented in multiple siloed data stores. We have data in the cloud, in the EDW, in Hadoop and in multiple application databases and files. Current solutions to integrate disparate data involve creating copies of data via ETL to staging databases, to the Hadoop data lake, or even to Excel files owned by business users.  This process is slow, very manual, and difficult to govern. Uncontrolled data copies are usually unmaintained and stale, and there is low reusability on previous work. Core transformations and metrics are often not supervised and generated on top of old copies, which creates a perception of low trust on key data.  Moreover, all these issues add a burden to IT and slow down the adoption of stat-of-the-art BI solutions |
| **Success Criteria**  At a high level, the following goals need to be accomplished to validate the adoption of a new data virtualization based architecture:   * Significant time reduction in the definition of new reports that involve data from multiple data sources * Compatibility with the rest of the company ecosystem: data sources, consuming applications, and corporate security * Good performance with large data sets for real time access to data   + Flexible caching options for scenarios when fresh data is not necessary * Scalability for high concurrency access, to ensure that the data virtualization layer can act as a centralized data delivery layer * Powerful logging and auditing capabilities to comply with current laws * Integration with our current DevOps procedures |
| **Current Effort Estimation**  What would be the **timeline** for the development of this use case **without** Denodo (in days): 30  How many people to **develop** the project: 5  How many people to **operate** the project: 3 |
| **Future Usage Estimation**  How many **users** will use the system? 1000  What will be the **peak load** (# concurrent queries): 50 |

**Technical Evaluation Requirements**

# **Technical Test Case #1**

|  |
| --- |
| **Data Warehouse Extension** |
| **Overview**  In many cases, we need to generate reports that cross corporate information from the warehouse with data that is not available in that data model. For example, data from some of our cloud applications like Salesforce, files, and operation databases. Currently, we have two options: modify the EDW schema to add the new data via ETL scripts or blend it directly in the reporting tool. The definition and approval of the new ETL jobs is very time consuming, and data blending queries in reporting tools do not work well when data volumes are massive.  We would like to use Denodo to approach scenarios like this in an easier way. For that, we want to test the definition of a new report that integrates data from SFDC and the EDW so that it can be consumed as a report in Tableau.  In addition, we also want to test connectivity with local Excel files and one of our Oracle databases, to validate the extensibility of this test to other common scenarios |
| **Data Sources**  Data warehouse   * System: Teradata 15 * Servername: teraprod * Credentials: denodouser / denodopassword * Database: EDW   Salesforce   * OAuth tokens will have to be generated during the POC * We have registered a new application (denodo\_test) and have the corresponding secret and key   Excel   * We have copied some sample Excel files to the folder /POC\_DV/Excel   Oracle   * System: Oracle 12c * Server: crmdb:1521 * SID: CRM * Credentials: denodouser/denodopassword |
| **Combination & Transformation logic**  The sample report will obtain some metrics on the revenue collected for new accounts. Since the detailed information on the sales processed is managed in SFDC by the sales reps in the field, it’s hard to maintain and update up-to-date values in the EDW  To build this report we will need the following:  From the EDW we will use the tables ORDERS\_FACT and CUSTOMER\_DIM. These tables can be joined based on the customer\_id. The Orders table is a big table, with ~200 million rows. Customer is around a 100k  From SFDC we need the following entities:   * Contact * AccountContactRole * Account * Opportunity   Those entities are associated based on their IDs:  Contact(contactId) -> AccountContactRole(accountId) -> Account(AccountId) -> Opportunity  We need those entities to evaluate what opportunities have been closed in the last quarter (based on field “closedate”)  The email field from the contact table will allow to join with the CONTACT\_EMAIL in the  CUSTOMER\_DIM table.  The final report will have to aggregate by Customer to obtain some metrics on the revenue collected for those new accounts |
| **Output**  Report data will be queries from Tableau and PowerBI. We will generate some bar charts to verify compatibility |
| **Other**   * Performance: is the main concern for this scenario, as data volumes in the orders table in the EDW are quite large. * Caching: We will also like to expose some of the caching features, especially to reduce the latency of the data coming from SFDC * Security: Access should respect our current security protocols and profiles. Denodo must integrate with our Active Directory and apply certain restrictions based on our user groups |
| **Success Criteria**  Definition of the new report should be quick. Current integration process takes several days to weeks in order to productionalize a report like this. We expect to reduce the development to a few hours or days.  Performance that allows online usage as well as compatibility with our current ecosystem (reporting tools like Tableau and Power BI and our security requirements) are also key pieces to consider this approach a success |

# **Technical Test Case #2**

|  |
| --- |
| **Logical Data Lake** |
| **Overview**  We own a Data Lake based on Hadoop where we store historical data that is not managed in the EDW or operational systems any longer. This system is also used by data scientists that work with massive amounts of data to produce advanced analytics, like predictive analysis.  Our initial approach of dumping all kinds of content into the lake has created an unmanageable catalog with hundreds of ungoverned tables, very often multiple copies of the same object with outdated data.  Therefore we would like to offer our data scientist a simple way to access all company data, so that they can be efficient in their day to day job, but at the same time we want to add some governance and policies about what to dump into the lake. We think virtualization can help us and would like to use this use case to validate this approach |
| **Data Sources**  Data Lake   * System: Cloudera Impala * Authentication: Kerberos based   EDW   * See connection details in use case #1 |
| **Combination & Transformation logic**  Combine the current purchasing data from the EDW and the historical dumps in the lake to produce some metrics around the purchasing trends of certain products. We will need the sales fact, as well as the product and time dimensions  Structures in the lake and the EDW should be very similar and easy to combine based on their PK and FK. The most up-to-date Product catalog data is available in the EDW. The date dimension is available in both systems. |
| **Output**  Data should be accessible via standard reporting tools (like Tableau and PowerBI), but also accessible from R for statistical analysis |
| **Other**  The results of their calculations should be available to persist as a new table in the data lake. Data scientists also need exports to CSV files sometimes, so we would like to see if that’s also possible |
| **Success Criteria**  Tight integration with the Hadoop ecosystem to read, process and write data back  Performance comparable to current usage  Flexible SQL-based interaction with the system for Power Users  Compatibility with reporting (Tableau, PowerBI) and data science tools (R) |

# **Technical Test Case #3**

|  |
| --- |
| **Web Service Provider** |
| **Overview**  Some applications to application integrations use JSON web services as integration protocol. Currently, those web services are developed manually on an ad hoc basis. Since Denodo enables access to the data model via web services, we would like to evaluate the ease of use of this feature, as well as the technical capabilities of the published web services |
| **Data Sources**  Same as test case #1 |
| **Combination & Transformation logic**  Test capabilities to shape the models into the hierarchical formats that JSON services usually offer. For example, if an user has multiple phone numbers, show a list with them |
| **Output**  Data will be exposed as a JSON RESTful endpoint. The endpoint needs to offer flexible RESTful capabilities like:   * Address rows individually by ID * Navigate to related resources * Filter based on any column * Pagination * Swagger self-documentation (aka OpenAPI) |
| **Other**  Security is an important part of the application to application communication. Authentication must be based on OAuth 2.0 and network traffic must be encrypted with TLS |
| **Success Criteria**  Besides the demonstration of the aforementioned capabilities, the development of the web services must be UI-driven and significantly reduce the development time compared with current manual development  Performance of the web services must be equivalent to the existing web services |

# **Technical Test Case #4**

|  |
| --- |
| **Data Virtualization as Data Delivery Layer** |
| **Overview**  Beyond the specific test cases #1 and #2, we would also like to understand the capabilities that Denodo offers to be a company-wide enterprise access layer. This entails a broader usage by a high number of concurrent clients of multiple types (reporting tools, direct SQL from custom apps, web services, etc.)  In addition such system should provide some sort of catalog and documentation of the data model to enable business user to browse through the tables as well as search and find what they need |
| **Data Sources**  Same as test cases #1 and #2 |
| **Combination & Transformation logic**  N/A |
| **Output**  Access to the full data model should be available via a web-based catalog.  End users should be able to easily search the metadata (table names, descriptions, columns, etc.) see the data lineage and descriptions/documentation, etc.  Users with the right privileges should be able to classify the metadata into different categories and tags, that will be used by business users to easily navigate and find what they are looking for  The tool should also offer data preview capabilities |
| **Other**  This tool must integrate with our security ecosystem (based on Kerberos and Active Directory) so that users can only see what they are allowed to |
| **Success Criteria**  The catalog needs to fulfill the needs of business users to navigate the company data models and improve the current governance |

**Annex 2: Sizing Guide**

**Introduction: Sizing for your Denodo Deployment**

This annex provides some guidelines for a preliminary sizing and capacity planning for your Data Virtualization deployment.

The result of this sizing exercise will be the number of cores that you will need to support your scenario. You can use these numbers to estimate your HW needs.

# **Characterizing your Environment**

In terms of sizing, your environment can be characterized based on the following details:

1. **Type of Workload**. This will give us a hint on the data volumes and concurrency that you will expect. Workloads can be classified as **operational** (small transactional queries) or **Informational** (heavier BI and analytic queries. Lower concurrency but higher data volumes).
   * In many cases, your system may have a hybrid workload, but for sizing purposes, we recommend you to break down the metrics on those two and then add the results to get your final numbers.
2. **Query complexity**. The complexity is determined by a number of factors, like the number of data sources, the amount of push-down that is possible, the operations involved, etc. It will translate into CPU usage.
   * Complexity in virtualization is different from complexity in your relational sources. A very complex SQL statement that is completely pushed down to your EDW requires minimal work for your Denodo server. However, a simpler query that involves several data sources with operations that need to be post-processed in the Denodo server will increase the complexity and CPU usage.
   * Our engineers will help you understand those differences during the POC. The queries you have selected for your test cases can be used as a base line for this estimation.
3. **Level of concurrency**. The number of concurrent queries running in parallel. This is usually determined based on the number of active users and the type of workload. For example, with operational scenarios, the concurrency is usually 10-20% of the number of active users. With informational scenarios, usually around 5%.
   * In some cases, it will be useful to measure average and peak load, to produce a range of cores instead of a single number. Since this is an estimation, a range will give you more flexibility to make your HW recommendations.

# **Estimating the Size of Your Deployment**

Once you have gathered those details, we can use the table below to estimate how many cores you will need. Keep in mind that if you have mixed workloads, the numbers will be easier to calculate if you break down the calculations in different categories, and then add up the final numbers.

The table below estimates the number of queries that a single CPU could handle. Once you have identified the row and column that corresponds with your deployment, divide your number of concurrent queries by the number in the table to get the number of CPU you will require.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **a. Low Complexity** | **b. Medium Complexity** | **c. High Complexity** |
| **1. Operational / Transactional # rows processed per query < 10 K** | 16 | 10 | 6 |
| **2. Operational / Transactional # rows processed per query < 100 K** | 12 | 8 | 5 |
| **3. Informational  / Analytical # rows processed per query < 10 M** | 9 | 6 | 4 |
| **4. Informational  / Analytical # rows processed per query > 10 M** | 5 | 4 | 2 |

***Table****: Number of concurrent queries per core based on workload characterization*

For example, if you have 100 concurrent low complexity operational queries with low data volumes (cell 1.a = 16), mixed with 30 concurrent high complexity medium volume informational queries (cell 3.c = 4), your sizing estimate would be:

*100/16 + 30/4 = 6.25 + 7.5 = 13.75 cores*

With that estimate, you can define your HW needs. For example, if you need High Availability, you can have a cluster with 2 nodes with 8 cores each (2x8 = 16 cores total)

**Annex 3: Services Assessment**

To create a Customer Success Plan, three (3) questions need to be answered:

1. **Use Case** – is it the same as defined in POC or different
   1. If different please include it below
2. **Success Metrics** - is it the same as defined in the ROI section of this doc
   1. If different please include below
3. **Milestones**

|  |  |  |
| --- | --- | --- |
| **Focus Area** | **Description** | **Milestone Date** |
| Strategy |  |  |
| Implementation – HW & SW |  |  |
| Design |  |  |
| Development |  |  |
| Operations – Perf Opt, Cache strategy, Scheduler, |  |  |
| Management – Security, Monitoring, VCS, License |  |  |
| Roll Out = User Training, Acceptance Testing |  |  |
| Internal support model/COE to answer 1st line questions that are relevant only to their environment such as naming conventions to follow |  |  |
| In Production – Go Live |  |  |
|  |  |  |

**Output**

Denodo Services needed:

* [Success Services Sessions Included in Contract](https://www.denodo.com/en/document/datasheet/denodo-success-services)
* Services Packages
  + E.g. Development Quick Start (40 hours included with contract or $12K)
  + E.g. Operations Quick Start (40 hours included with contract or $12K)
* Open Services
  + E.g. purchase of 40 hours to supplement Success Services sessions ($200 / hr x 40 = $8,000)
  + High level SOW describing general use of these Services
    - E.g Build out Semantic model & Naming conventions best practices

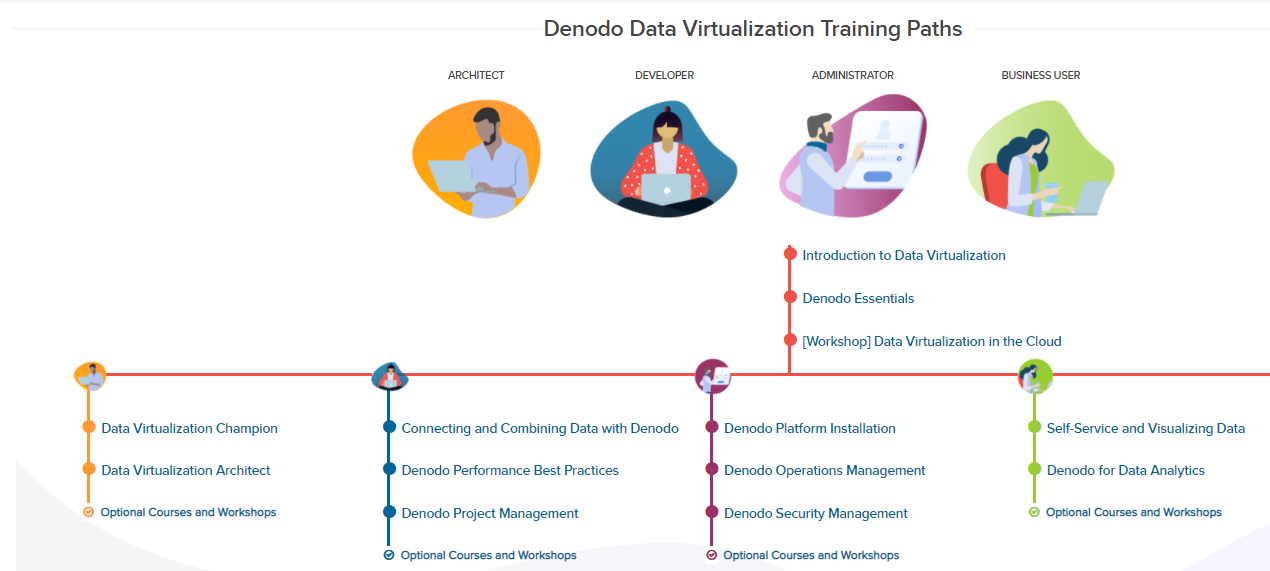
e.g. Total Services added to the Contract

* Success Services - priced as appropriate
* Services Modules - priced as appropriate
* Open Services - $8k

**Proposed Customer Staffing Levels –Org Chart, Roles, Skills and Education Recommended**

[**https://www.denodo.com/en/denodo-platform/services/education**](https://www.denodo.com/en/denodo-platform/services/education)

|  |  |  |
| --- | --- | --- |
| **Role** | **Skills** | **Denodo Training Courses** |
| Denodo Manager |  |  |
| Architect |  |  |
| Developer (Qty ?) |  |  |
| Admin. (Qty ?) |  |  |
| Business User |  |  |

****

**Annex 4: ROI Guide**

The table below summarizes the most relevant parameters that characterize the costs of your current data integration efforts. Based on those parameters and the definition of the use cases, we can help you estimate the ROI of an alternative data virtualization deployment.

# **IT Baseline Costs**

**Informational Projects (e.g. DW, Data Marts, ETL, Big Data)**

Describe the current process involved in the development of new reports, ETL jobs, data marts, etc. For example:

* How many people work building and editing per month?
* How many projects per month?
* How long does it take to create a new project? How about editing an existing one to adapt to new requirements?

**Operational Projects (e.g. Data Services, Single view of Customer, etc.)**

Describe the current process involved in the development of these projects. For example:

* How many people work building and editing per month?
* How many projects per month?
* How long does it take to create a new project? How about editing an existing one to adapt to new requirements?

**Consuming Applications (e.g. BI and Dashboards, App. Development, etc.)**

Describe the current process involved in the development of these projects. For example:

* How many people work building and editing per month?
* How many projects per month?
* How long does it take to create a new project? How about editing an existing one to adapt to new requirements?

# **Business Impact**

**Business Users**

Time spent in Business Processes, for example Financial Analysis, reporting, customer service, claim management, etc. For example:

* How many users?
* How many hours spent per month looking, prepping and downloading data?

**Time to Market today**

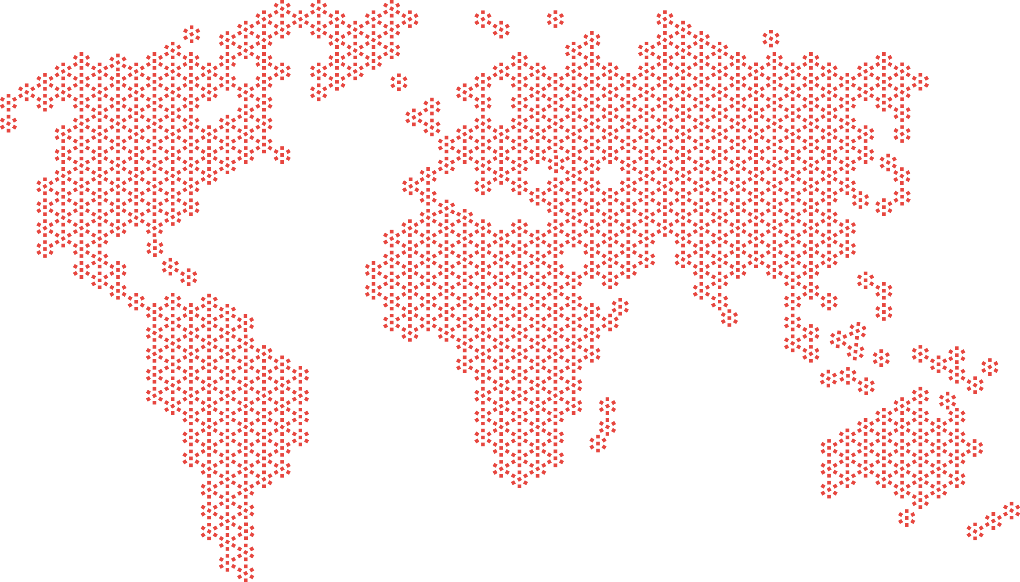
For each product, program or initiative that requires data, describe the current process

* How many team members?
* How many hours are spent per month?

**Business Operation Cost per Business Unit**

For example, projects around security, compliance, data governance, data quality, etc. Describe the current process to build out the data sets and maintain it

* How many team members?
* How many hours are spent per month?

****

Denodo

Denodo is the leader in Data Virtualization – providing unmatched performance, unified access to

the broadest range of enterprise, Big Data, cloud and semi-structured sources, and the most agile data services provisioning and governance – at less than half the cost of traditional data integration.

Denodo’s customers have gained significant business agility and ROI by creating a unified virtual

data layer that serves strategic enterprise-wide information needs for agile BI, big data analytics,

web and cloud integration, single-view applications, and data services across every major industry.

Founded in 1999, Denodo is privately held.

Visit www.denodo.com Email info@denodo.com twitter.com/denodo