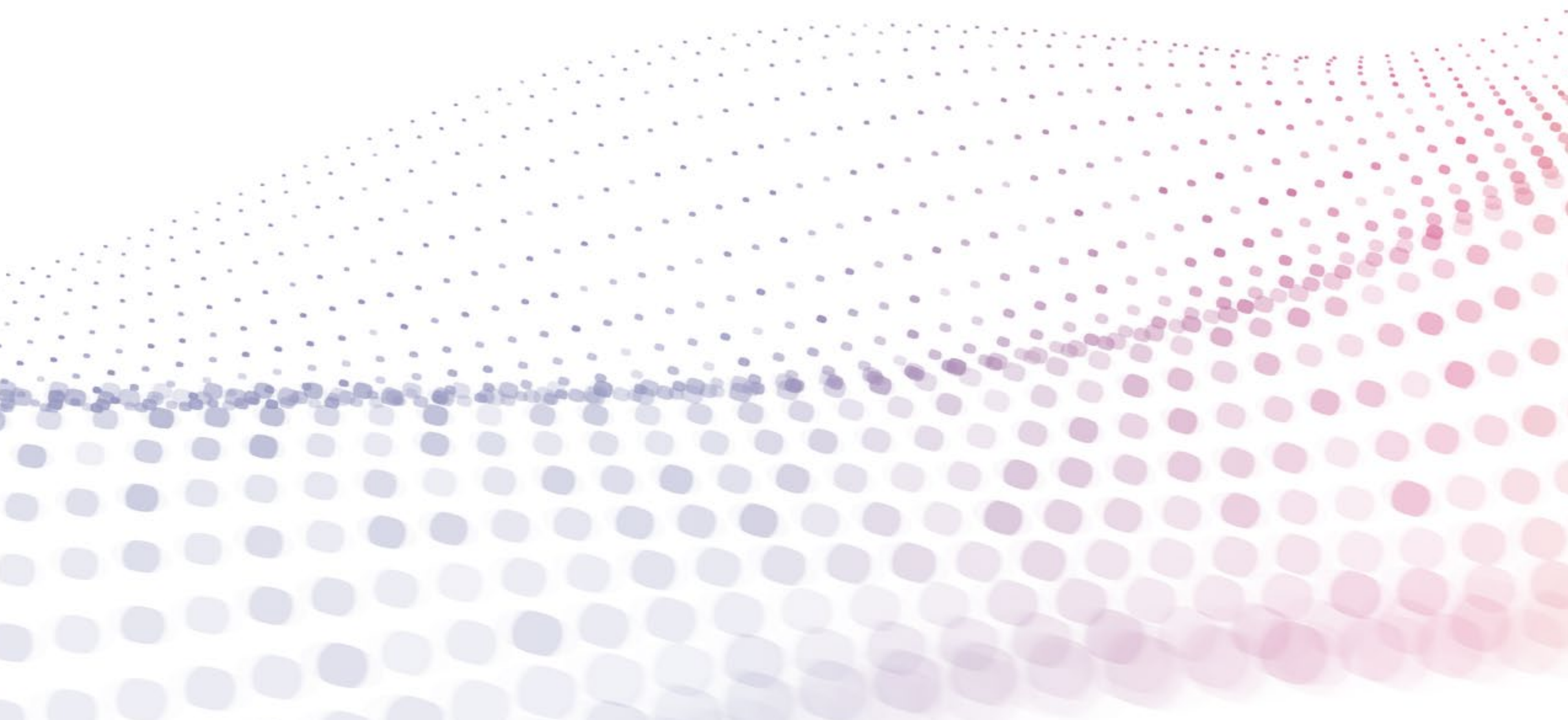
MarkLogic®

HHS Operational Data Warehouse

Real-time Operational Reporting with AI

MARKLOGIC WHITE PAPER · DEC 2019

With the pace of change in healthcare and human services delivery today, the wide variety of data formats and the extreme volumes of data, traditional relational data warehouses no longer suffice. Next generation healthcare data warehouses must not only handle the volume, variety, and velocity of data from any desired source – but do it without sacrificing enterprise features.



Contents

An Era of Change in Healthcare and Human Services

We are in an era of dramatic healthcare transformation. It goes beyond the sheer rate of change – beyond the new drugs, procedures, and treatments being developed. Wearables like Fitbits*, Apple Watches*, and Bluetooth*-enabled glucometers are increasing the volume and variety of healthcare data – and creating new security risks. Tomorrow's electronic health records are guaranteed to be radically different – and larger – from today's.

Value-based care and integrated systems of care are here to stay, and are on the rise. Regardless of politics, it appears the Medicare Access & CHIP Reauthorization Act (MACRA), or some flavor of the same, will continue to move providers from fee for service to value-based payments, with a major impact on both clinical and financial systems. Managed care organizations and government HHS agencies are increasingly looking to capture and utilize information on Social Determinants of Health in order to improve outcomes – particularly for medically and socially complex populations – while reducing overall healthcare utilization and spending.

Data and data management systems must evolve. The recent focus on interoperability and data exchange is driving rapid change in data exchange formats. Fast Healthcare Interoperability Resources (FHIR) is becoming an increasingly popular format; however, this standard is not universally adopted, and may yet only be a step towards the next "standard." In addition, various types of data – not just structured, but increasingly poly-structured (HL7 v3), semi-structured (SOAP Notes), and unstructured (provider and case manager narratives) all contain pieces of the puzzle that are essential for providing clinical and performance value. Because each of these varied, evolving, and emerging data formats only presents part of the picture, it's critical that data be combined, considered, and analyzed together, and for processing to happen in near or real time to serve the patient and the business. Data integration presents a significant obstacle to deriving benefit from this explosion of healthcare data.

Unfortunately, traditional technologies are built for static data models – and are not up to the job. Traditional relational-based data warehouse approaches are particularly ill-suited to rapidly changing data sets that must be integrated in real-time. The complexity of integrating and using all of this highly varied and continuously evolving data across a myriad of clinical, financial, and other systems presents a major challenge that demands a new approach.

What is truly needed to meet tomorrow's healthcare organizations' needs is a more flexible data warehouse that provides access to the broadest range of data, in near real-time, while supporting transactions.

The Evolving Importance of Data in Healthcare and Human Services Delivery

Data is frequently referred to as the new “oil” and is increasingly critical to health and human services organizations in improving program management to reduce costs, increase efficiency, and deliver better outcomes. Currently, data is generated and resides in a variety of sources or products that are used within or are interacting with a healthcare system. These sources and products traditionally exist in numerous different departments or are used to support the activities of different “personas” in the enterprise, including clinicians, case managers, accountants, analysts, etc. Examples include electronic medical records, decision support systems, nurse documentation, pharmacy, laboratory, radiology, billing, registration, and various other information systems. And, there is a growing interest in incorporating new information like Social Determinants of Health to improve outcomes and resource utilization. Analyzing requires even more flexibility due to the wide variety of additional data sources such as economic, financial, geospatial, education, social media, advertising, or news – each with their own methods of connection and rates of change, including continuous, real-time updates.

In a traditional data warehouse approach, the information from these systems is aggregated into a single, central system in order to make this data accessible and actionable. Unfortunately, data integration to the central system schema is challenging, because healthcare and service delivery data comes in the wide variety of forms and formats described above. They are incompatible, serve different “personas” and change rapidly – making it difficult to load them all into a single fixed schema, as is required by a relational data warehouse. For example, a typical EMR (Electronic Medical Record) can have hundreds of tables containing textual and numerical data, and a given enterprise can have a wide variety of ambulatory and hospital EMRs that it wants to look across or aggregate information from. To add to the complexity, radiology systems use images, older medical records may exist as PDFs, text-based notes and reports often contain the most important information, critical care units use waveform-based technologies, etc. Social Determinants of Health could include images of buildings, maps or complex geospatial layers data which represent the physical environment including walkability, access to parks, and housing. These types of data can provide distances to gyms, fresh food, bus, and public transit accessibility or lack thereof, which can have a profound effect on a person’s health. This geospatial information can also provide distance and time to get to work, which provide clues to levels of stress and time for rest and other essential non-work activities. Even if the data is structured, sometimes the same data exists in many different systems – e.g., when an individual is enrolled in multiple benefits programs – or in different forms or formats, each using different coding schemas, ontologies, or controlled vocabularies.

Relational database technologies are traditionally used as the basis for healthcare data warehouses. One of the primary issues with these databases is they require upfront data modeling – including a precise definition and unanimous agreement on every field and relationship – before users get access to any of it. This upfront data modeling is an expensive, time-consuming task requiring an intimate knowledge of all data sources and data usage patterns or analytics coupled with detailed ETL mapping, coding, and testing for precision. Furthermore, the query patterns need to be known in advance in order to create appropriate indexes to deliver acceptable performance. This approach requires a lengthy waterfall development process during which time the original requirements or formats can change. It therefore assumes a static environment where data sources and access patterns don't change over time. If the model is incorrect – e.g., if a data source was unaccounted for or its structure changes – the process must be revisited. If users want to ask questions of the data that the DBA didn't anticipate, additional data structures and indexes may be necessary. In many cases, this means further schema changes and a repeat of the model/ETL cycle, as depicted in Figure 1 below.

This process is too difficult for healthcare and human services data – or for any complex IT project. In fact, McKinsey surveyed companies and found that "on average, large IT projects run 45 percent over budget and 7 percent over time, while delivering 56 percent less value than predicted. Software projects run the highest risk of cost and schedule overruns… 17 percent of IT projects go so bad that they can threaten the very existence of the company."

To make matters worse, relational databases are really designed to handle structured data, whereas most of today's data sources in healthcare and human services involve semi-structured, poly-structured, or unstructured data (i.e., text, PDFs, images, and multimedia). Using these other data sources requires adding newer approaches and technologies (e.g., data lakes) in order to simply house or file the data for eventual use, plus various other data technologies and supporting databases (text search, security, NoSQL, etc.) in order to make the data useful or to allow access for comparison to other data stores.

Consider full-text search. Relational data warehouses cannot deliver full-text search on invaluable unstructured data such as provider notes, which often contain critical information not captured in the structured data. Also, consider semantic data, or "linked data" sets which are common in healthcare – these cannot be fully leveraged in a relational system to make inferences not directly stated in the data. Ontologies of semantic concepts are needed to expand queries to include generic drugs, drugs in the same class, or drugs with the same active ingredient(s). Relational databases again have trouble dealing with these ontologies (and even taxonomies) of semantically related terms.

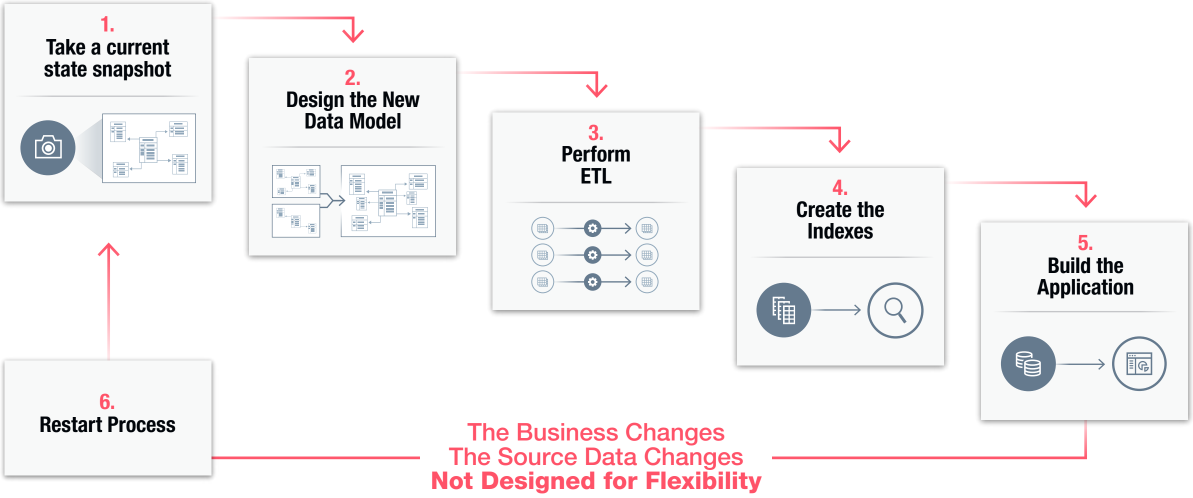


Figure 1: Challenges With the Traditional Relational Approach

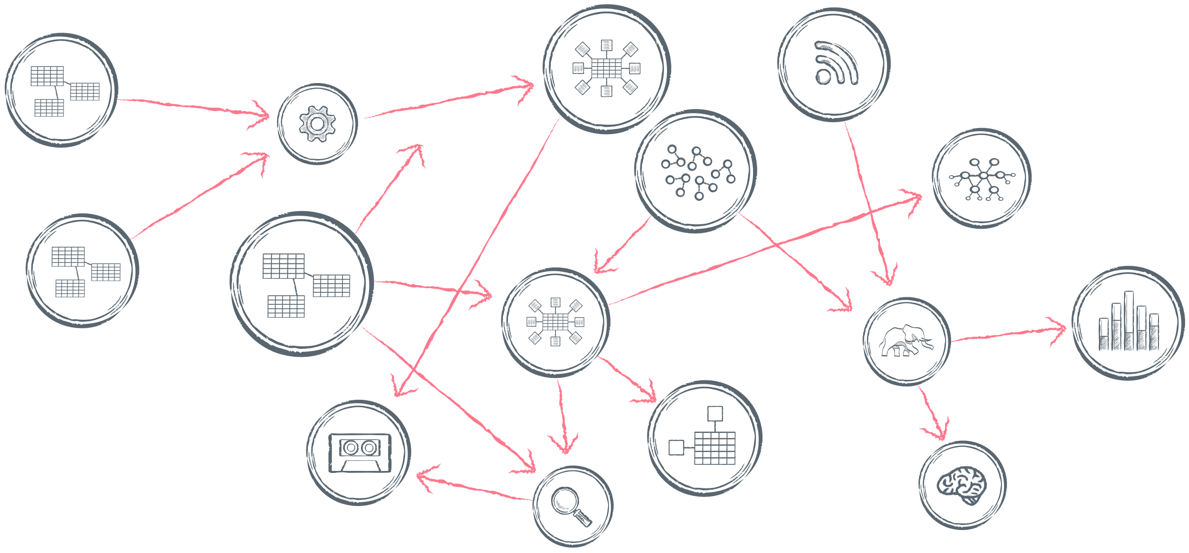


Figure 2: In a typical enterprise, data silos proliferate and require more and more time and resources to integrate.

The traditional solution to the limitations of the traditional data warehouse is… more software. Additional products are added to handle these other data formats, with additional integration development steps, brittle integration code, more trained staff, more provisioned hardware, more run-books (instructions for what to do for normal operations as well as emergencies), test plans, upgrade plans, vendor or expert consultant relationships, support contracts, monitoring, and processes to keep them all in sync. All this complexity makes projects take longer and adds to a higher total cost of ownership while still resulting in a system that is not easy to query across.

The result is that a typical data warehouse ends up with a complex array of various relational data stores, often with complex connections or "joins" between them, as well as specially created relational data marts designed to support the use of particular users.

It is critical that new technology not only keep pace in acquiring, housing, accessing, and analyzing this proliferation of health-related data, but do so without sacrificing the critical enterprise features present in relational databases, such as government-grade security, high availability, built-in disaster recovery, etc. that organizations have come to expect and need.

In addition, many users need real-time data. Ideally, real-time analytics will inform health care as it is being delivered. Real-time data delivery is needed to enable a true "learning organization" – where learning occurs (and interventions ideally applied) as care is delivered. Data from operational systems must be available as things happen. This is of particular importance in this modern era where real-time data includes the evolving internet of things (IoT), which is becoming the new standard to monitor and optimize care and costs in an evolving value- and risk-based environment.

Limitations of Current Approaches

The enterprise data model approach to traditional data warehouse design is a top-down approach that is mostly focused on the needs for healthcare analytics. In this approach, the goal is to model the perfect analytic database – determining in advance everything needed to be able to analyze to improve program efficiency, outcomes, safety, and citizen/patient satisfaction. And the database is structured accordingly. Although in theory this is as if a new system is being built from the ground up, in reality it is a secondary system that receives data from systems and databases that are already deployed and supporting various clinical, financial, or other operations. The problem is that exporting the data from all of these various systems and making them all work together in a new system is incredibly time-consuming and expensive. By the time all the data is in place, there are new questions, new data sources, or the questions the system was designed to ask are now not as relevant to current business needs.

There are also more fundamental limitations to this approach. Analytical relational data warehouses are not designed for operational, transactional, or real-time processing. Also, these data warehouses are almost exclusively focused on the structured data, whereas a majority of data in healthcare is currently (and increasingly) semi-structured or unstructured. The key is to find a way to use all the data, and to be able to do that in near or real time without negatively affecting the performance of existing systems that may be supporting various healthcare and human service operations. In addition, next generation data warehouses need to address other issues such as data privacy, security, governance, and other such matters of critical importance in healthcare data management today.

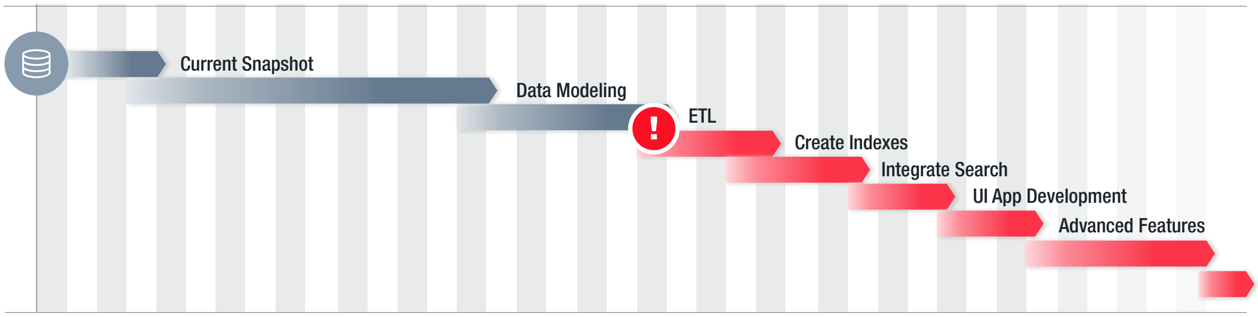


Figure 3: The traditional approach to designing and implementing a data warehouse requires significant upfront resources

Next-Gen Healthcare Data Warehouse Requirements

Document-centric

There are many types of data in patient healthcare, including claims, PHI docs following Clinical Document Architecture (CDA), clinical protocols, and many more that lend themselves better to formats that can only be captured and searched efficiently in documents, ideally as either JSON or XML. Consider a single claim with multiple line items and how that is currently broken apart in order to store it in RDBMS tables. Or consider a patient's health information held in a C32 document, with many sections that include rich hierarchical data that together provides a holistic view of a patient's healthcare. It logically makes sense to store this information together as one document entity. With an XML document model, and to some extent with JSON, the entire claim or C32 remains intact – which enables more information to be considered so the claim can be treated holistically without a number of complex joins.

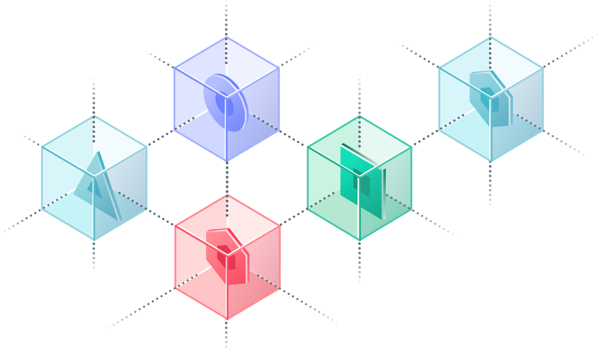
Electronic Medical Records are often represented as XML documents; for example, a CDA or HL7 v3 patient record is an XML document containing a wide range of patient health information for a given patient. A healthcare data warehouse should be able to load an HL7 v3 document "as-is" and have it be automatically indexed to support immediate complex queries. In contrast, loading this into a relational database requires the document's data to be shredded and spread in many tables with indexes set up to support the queries the DBA expects will need to be run. When a query is run, and when the indexes support it, it requires many complex joins to piece a patient's health information back together. As an analogy, you can think of the document-oriented vs. relational database approach to storage as akin to storing a car intact in a garage (document-oriented), vs. taking it apart and storing its parts in separate bins and reassembling when needed (relational). Being able to store these health documents as is ensures they are automatically fully indexed, and you can then immediately run queries of any complexity against them; this simplifies the data warehouse while also increasing its flexibility to support both known and unknown use cases.

MarkLogic provides exactly this capability. The MarkLogic® data platform uses documents represented as XML or JSON as one of its primary data models. Because it uses a non-relational data model and doesn't rely on SQL as its primary means of connectivity, MarkLogic is considered a "NoSQL database." Financial contracts, medical records, legal filings, and claims are all naturally modeled as documents. Relational databases, in contrast, with their table-centric data models, can't represent such data as naturally, and so they have to either spread the data out across many tables (adding complexity and hurting performance) or keep the data as unindexed BLOBs or CLOBs.

In addition to XML and JSON, the multi-model MarkLogic platform can store text documents, binary documents, and graph data as RDF triples. Text documents are indexed as if each were an XML text node without a parent. Binary documents are by default unindexed, but MarkLogic provides the option to automatically convert over 200 types to searchable XML, including PDF. Behind the scenes, MarkLogic turns RDF triples into an XML representation and then stores them as XML documents.

Multi-model

Data models determine how information is stored, documenting real-life people, things, and interactions in an organization and how they relate to one another. With a document-centric model, you can leverage the XML and JSON formats to represent records in ways that are richer than what is possible with relational rows.



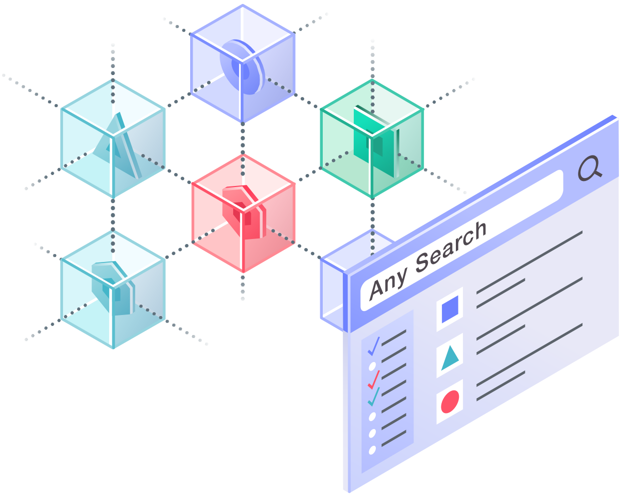
A triple store houses semantic graph data, ideal for representing facts about the world, as subject-predicate-object structures known as RDF triples. Triples describe relationships among data entities; for example, a specific person has a certain disease, which is part of a class of diseases, which is treated with a specific drug, that has specific active ingredients. Ideally triples can be stored in the same database as documents or even embedded within documents, to make it easier to group patient cohorts – with information in the triples serving as the link. For example, a group of patients without their own transportation, who only live near convenience stores with fresh groceries far from their home addresses. Triples could be used to link specific instances of convenience stores to the convenience store concept which makes it easier to write the query, but capture all possible convenience and grocery stores and calculate distances to these without explicitly stating the list of convenience and grocery stores. Triples as part of JSON or XML documents can also be used to link any business entity, such as members to claims, providers, services, or other family members.

MarkLogic has built native support for both the document model and the semantic model with specialized indexes allowing for complex queries crossing both models. MarkLogic is a multi-model database platform, which is what organizations need to store and query healthcare data in its many forms.

Search-enabled

Healthcare data includes much narrative text from things like Provider notes. As such, a healthcare data warehouse must support numerous search features including word and phrase search, Boolean search, proximity, "mild not," wildcarding, stemming, tokenization, decompounding, case-sensitivity options, punctuation-sensitivity options, diacritic-sensitivity options, document quality settings, numerous relevance algorithms, individual term weighting, topic clustering, faceted navigation, custom-indexed fields, geospatial search, and more. It should be possible to include search options along with advanced structured query constraints in the same query, as much of the information in narrative text does not exist in patients' structured records.

MarkLogic was created to allow database-style queries to be performed against unstructured data, so again includes features purpose-built for use cases such as semi-structured healthcare data access and analysis. To do this MarkLogic uses search-engine-style indexes that are fully composable and search-enabled.



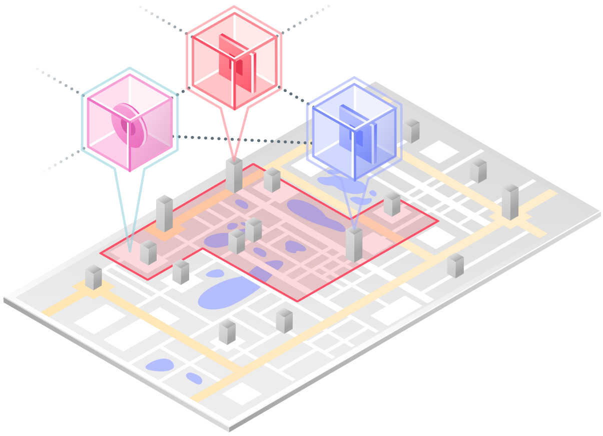
Geospatial and Structure-aware

Text and structure must be able to be queried together efficiently. For example, consider the challenge of querying and analyzing potentially incorrect prescriptions, or drug addiction; you may want to run a complex query such as "get me all patients with more than 100 Schedule II prescriptions in the past two years, who have visited pharmacies greater than 200 miles from their home address and do NOT have any mention of the word 'pain' in any doctor's visit notes," or another focused on social determinants, such as "Get me all persons who have two or more chronic conditions, have an income below $30,000/year, live more than a 10 minute walk from the nearest public transportation, and include the text 'fast food' or any fast food restaurant in their doctor's visit notes." For the latter you are not only analyzing structured data AND text, but also including complex geospatial and something called semantic query expansion to find matching terms for fast food restaurants and all chronic conditions and their many types, for example, all of the HCPCS codes for diabetes.

By using XML or JSON documents to represent patient data, structure-aware indexing, geospatial indexing and a semantic triple store to understand what's a walking distance, what's a fast food restaurant, chronic condition, or a Schedule II prescription, and which text is quoted and which isn't, a query like this is actually easy to write and highly performant in MarkLogic. Additional terms can be easily added without affecting performance to narrow the cohort, such as, by age range or gender.

Schema-agnostic

Next generation healthcare data warehouses need to be highly agile, support adding new data sources quickly, and flexibly respond to changes in the data from existing data sources. Data from any source should be able to be loaded as is, thereby speeding up discovery and development by allowing an incremental, agile approach. To do this, these warehouses cannot have a fixed single schema as is required from relational databases – they need to be schema-agnostic.



XML and JSON are self-describing in that each piece of data is preceded by or enveloped in their field or element names. This allows a document-oriented database to "know" what is being ingested, including the hierarchy of the data. A document database like this does not need to be told what schema to expect, any more than a search engine needs to be told what words exist in the dictionary.

Being able to index and query efficiently, without prior knowledge of a schema, provides real benefits with unstructured or semi-structured data where:

- A schema exists but is either poorly defined or defined but not followed
- A schema exists and is enforced at a moment in time but keeps changing over time and may not always be kept current
- A schema may not be fully knowable, such as intelligence information being gathered about people of interest where anything and everything might turn out to be important

It also benefits you when source data is highly structured and where source schemas exist by:

- Isolating applications from schemas that change over time, even with little or no notice
- Making it easy to integrate data from a variety of sources without having to design a master schema that can accommodate all data from all sources
- Allowing new data to be added to an existing database without having to redesign the schema for existing data or rebuild the database

These characteristics are present in MarkLogic via its Universal Index, making it the ideal technology for complex or large-scale data integration or data warehousing projects, like a healthcare operational data warehouse. Of course, MarkLogic also works well with data that does fully adhere to a schema. You can even use MarkLogic to enforce a schema.

In-line Text Enrichment

There is a significant amount of information locked up in text format, such as in a social worker's or provider's notes or in other narratives. This text often contains information that is not found elsewhere in a patient's electronic medical record, so ignoring it or being unable to query or otherwise leverage it leaves you with an incomplete picture of a patient's health. Just being able to search these narratives provides an advantage over relational storage, but there are additional steps that can turn this unstructured text into semi-structured content. Using entity enrichment or NLP (Natural Language Processing), it is possible to find the occurrences of any drugs, procedures, health problems, social history, dates, and more in the unstructured text, and to add XML tags around the text and additional information, such as the RxNorm or ICD code associated with the text.

An example of this is shown in the in-line enriched text below.

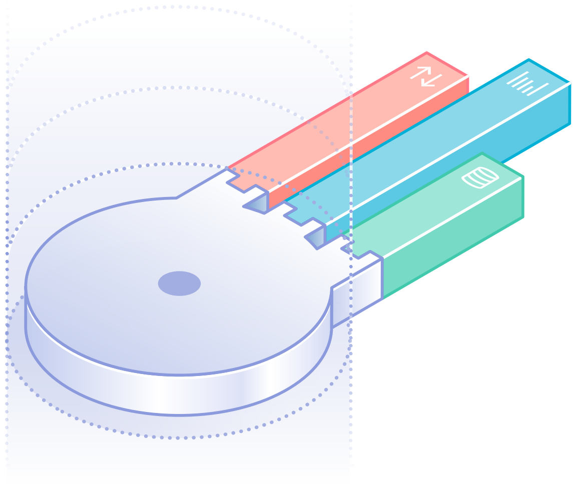
Enriched text like this is packed with additional information that can be indexed to support queries like "all patients who have received a PHQ-9 with provider notes from any time during the year 1971 that mention any drug in the MAO Inhibitor class within five words of the drug Nardil." There is a lot of queryable information in text that would usually be overlooked in traditional data warehouses. Next-generation healthcare data warehouses should be able to take advantage of enriched text. MarkLogic can, and what's more impressive is that the constraints in this query can be combined with any number of additional constraints hitting any data regardless of type or schema.

```
<physician-notes date="1971-03-12">

   The patient scored a 78 on his <cpt type="PHQ-9" desc="behavioral health assessment">
   PHQ-9</cpt> leading me to the conclusion that he is <ICD-9 code="296">episodically
   depressed</ICD-9>. An additional behavioral health assessment by another provider
   should be performed in two weeks.

   Enter episode as a <cpt code=99420" desc="administration and interpretation of a
   health risk assessment">99420</cpt>. The patient should get on an anti-depressant.
   First we'll try <drug ndc=0071-0350" class="Antidepressant, MAO inhibitor"
   generic="phenelzine">Nardil </drug> followed by <drug ndc="na" class="tricyclic
   antidepressants" generic="amoxapine">Asendin</drug> if mood does not improve and
   patient's fear and anxiety continues.

</physician-notes>
```



Simplified Data Access

While a data warehouse is generally used to observe things after the fact, next-generation healthcare data warehouses should be more adaptable and capable of real-time reporting, real-time access from a wide range of healthcare applications, including support for transactions from those applications.

In order to support access from a wide range of applications and reporting tools, MarkLogic exposes a set of core services through a nearly universally accepted HTTP-based REST API. They provide services for document insertion, retrieval, and deletion; query execution with paging, snippeting, and highlighting; facet calculations and server administration.

MarkLogic also supports SQL, SPARQL, a Java Client API, Node.js Client API, .NET API, and XQuery, and a host of APIs in other languages have been released as Open Source projects. These APIs let developers integrate a MarkLogic Operational Data Warehouse with SAS, Cognos, Business Objects, Tableau, Qlik, workflow tools, and other applications using the languages with which they are familiar.

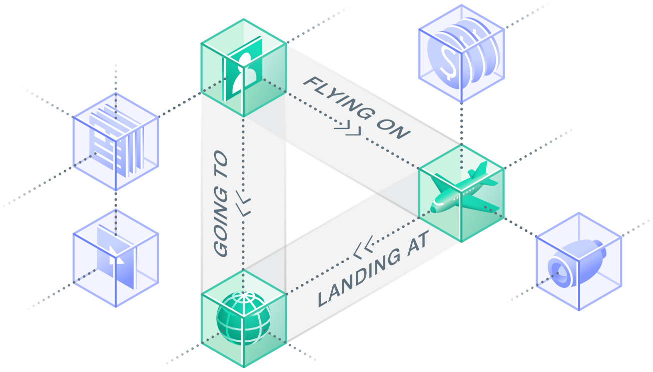
Semantic Data Support

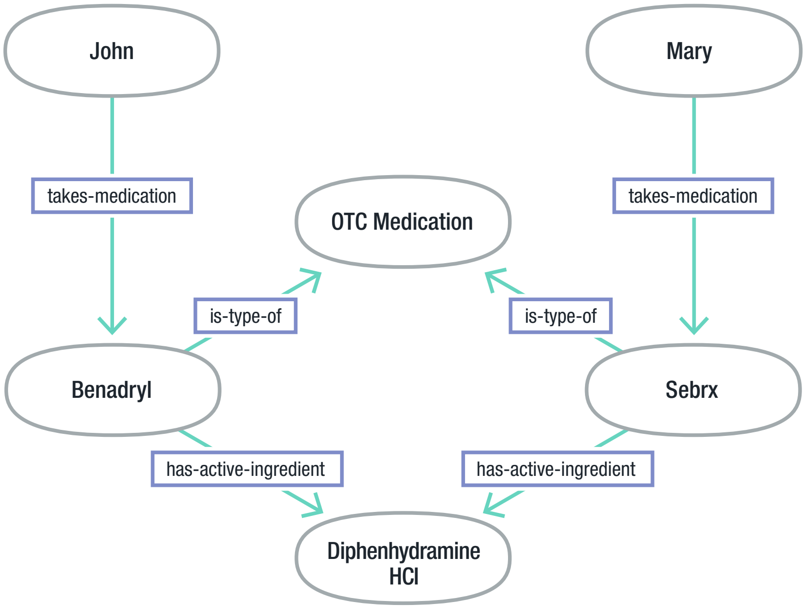
Semantic data sets are prevalent in healthcare and can do wonders for analyses by making it easier to find related data without deep knowledge of the domain. For example, the RxNorm semantic data set can be used to find all drugs with the same active ingredient and expand a search for patients taking one drug, into a search that includes all patients taking any drug with that same active ingredient. Similarly, queries can be expanded semantically to find all claims with similar procedures, or all claims for persons with any of the many forms of diabetes without requiring the user to know all the codes or description for all types of diabetes or what the similar procedures are for any given surgery.

Semantic data consists of facts stored as triples. A triple describes a relationship among data elements and consists of a subject, predicate, and object (similar to human language):

SUBJECT	PREDICATE	OBJECT
John	takes-medication	Benadryl
Benadryl	is-type-of	OTC Medication

A key aspect of semantic data is not just that it describes relationships (that let you answer the ques-





tion, "What over the counter medications is John taking?"), but that these relationships can be interlinked. You can add more triples describing John, and also triples describing Mary, and some of these facts will intersect, such as that Mary and John take OTC medications with the same active ingredients despite only knowing that John takes Benadryl and Mary takes Sebrx. This results in a connected web of information that can be represented as a graph.

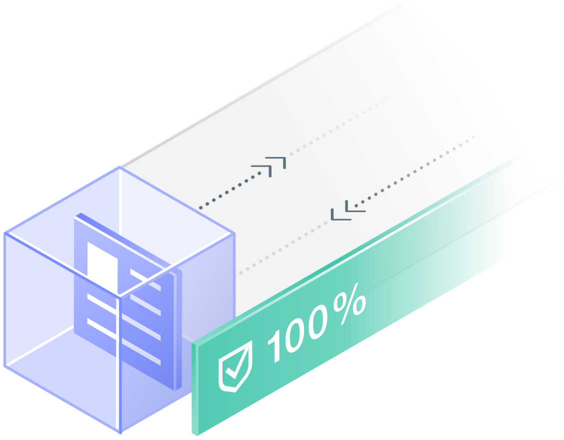
MarkLogic supports storing and querying semantic data by allowing data to be represented as triples and queried using the semantic standard query language, SPARQL. MarkLogic applications can query large collections of triples (or semantic graphs) to tell us interesting things. By traversing the triples in a graph, Mary and John can be grouped into the same cohort despite non-obvious connections between them.

Semantic data allows documents and other data to be easily linked together with context. These links do not only contain cardinality information like the links between tables in an RDBMS – they contain meaning.

Transactional

Tomorrow's healthcare data warehouses should not be limited to static data copied from operational systems. Researchers should be able to write to the data warehouse to annotate collections, and add their analyses or comments to the data. With a shared nothing architecture that clusters, a data warehouse can be easily scaled to even support operational use cases, for example, being the system of record to one or more HHS applications and providing those applications with the benefits of a more complete, 360-degree view of patients and other important entities. To support operational activities a data warehouse needs to be transactional while maintaining performance under simultaneous analytical workloads.

MarkLogic stores data within its own transactional repository. The repository wasn't built on a relational database or any other third-party technology; it was built with a focus on maximum performance.



Because of the transactional repository, you can insert or update a set of documents as an atomic unit and have the very next query be able to see those changes with zero latency. MarkLogic supports the full set of ACID properties: Atomicity (a set of changes either takes place as a whole or doesn't take place at all), Consistency (system rules are enforced, such as that no two documents should have the same identifier), Isolation (uncompleted transactions are not otherwise visible), and Durability (once a commit is made it will not be lost).

ACID transactions are considered commonplace for relational databases, but they are a game changer for schema-agnostic NoSQL databases and search-style queries.

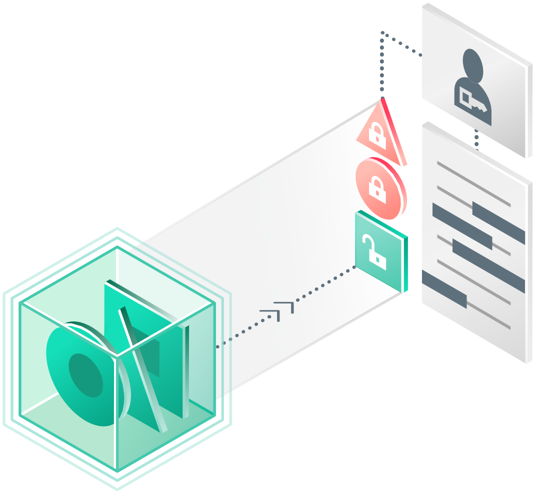
Security

With all the PII and PHI that goes into a healthcare data warehouse, strong security and auditing features must be present in the database, and good security policies need to be in place.

An operational data warehouse should support role-based access control along with element or cell level protections to prevent access to the most sensitive data, such as SSNs. These controls allow you to prevent sensitive details in a document from being revealed to anyone without the proper security role, while still allowing the user to see the rest of the document. The data warehouse should also support redaction to remove element values or replace them with synthetic values when exporting to external systems with less security, such as Business Intelligence and Reporting tools. Less granular security controls that can only prevent access to the patient record are not sufficient.

Increasingly, healthcare and government data professionals are choosing to go beyond security features that prevent access to data, to those that can protect the data itself in the event of loss of a laptop, or access to the disks that contain a database's data. As such, security professionals are choosing to further secure their health data with database encryption "at rest" and even demanding that the encryption keys be managed separately from the database – to keep any one person from controlling the so-called "keys to the kingdom," and to prevent physical loss from resulting in data loss.

MarkLogic includes the security features described above to enable element-level security, redaction, and database encryption at rest with external key management. Security is provided through role-based access control (RBAC) along with a "compartment security" option. MarkLogic's government-grade security has been



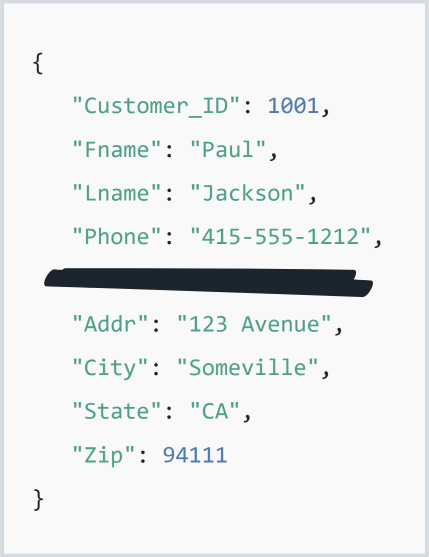
hardened through many DoD and Intelligence Community projects. The product has completed multiple vulnerability assessments and been validated by the National Information Assurance Partnership (NIAP) as complying with the Common Criteria DBMS profile at Evaluation Assurance Level 3 (EAL3) augmented with ALC_FLR.3 (highest level of flaw remediation). MarkLogic is installed and operational on government systems with demanding security policies. These policies include stringent measures for access, authentication, management, audits, role separation, and system assurance. For example:

- **NIACAP** (National Information Assurance Certification and Accreditation Process) – Developed by the U.S. intelligence community for certification and accreditation of computer and telecommunications systems that handle U.S. national-security information
- **NIST Special Publication 800-37** – Guide for Applying the Risk Management Framework to Federal Information Systems; supports the six-step Risk Management Framework (RMF)

Additionally, customers have received Authority to Operate (ATO) for information systems utilizing MarkLogic that involve almost all of the major systems security standards. These standards continue to evolve and MarkLogic stays up to date on the latest changes (for example, SSAE 18 will soon replace SSAE 16). The system security standards currently in place on systems running MarkLogic include the following:

- NIST 800-53
- ICD 503
- FIPS 140-2
- HIPAA
- SOX 02/404
- SSAE 16
- EU 95/46/EC

For more information on MarkLogic Security capabilities, see www.marklogic.com/trust.



```
{
   "Customer_ID": 1001,
   "Fname": "Paul",
   "Lname": "Jackson",
   "Phone": "415-555-1212",
[redacted]
   "Addr": "123 Avenue",
   "City": "Someville",
   "State": "CA",
   "Zip": 94111
}
```

Redaction

Similar to element-level security on query, redaction allows you to omit elements from result sets, change their values to some random string or number or replace the contents with the same characters for every instance, for example, every male becomes “John Doe.”

In healthcare, there are numerous use cases where you do not want to export certain PHI information, for example, to support testing environments, research, BI/Reporting, or even some claims. You want to mask, or redact, their names, SSNs, addresses, and anything that could be used to identify them.

Redaction is available in MarkLogic and is role-based – the redacted elements for the same document can vary based on role. Deterministic masking is also supported, which ensures the same value (such as a SSN) is always masked to the same replacement value thereby preserving connections across a variety of test data samples. MarkLogic uses NIST approved AES-256 encryption along with optional custom “salt” values such that although the replacement value is deterministic, reverse engineering the original is essentially impossible.

Auditing

A strong auditing capability is critical to being able to capture security-relevant events to monitor suspicious database activity and to satisfy applicable auditing requirements. Auditing is needed to perform the following activities:

- Enable accountability for actions. These might include actions taken on documents, changes to configuration settings, administrative actions, changes to the security database, or system-wide events
- Deter users or potential intruders from inappropriate actions
- Investigate suspicious activity
- Notify an auditor of the actions of an unauthorized user
- Detect problems with an authorization or access control implementation. For example, you can design audit policies that you expect to never generate an audit record because the data is protected in other ways. However, if these policies generate audit records, then you know the other security controls are not properly implemented
- Address auditing requirements for regulatory compliance

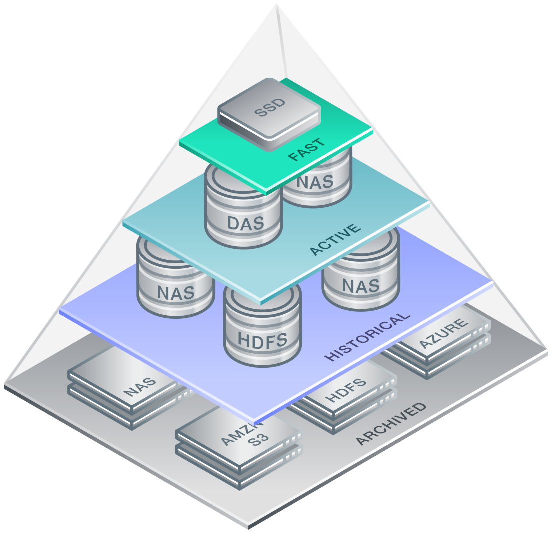
FISMA's risk-based assessment and monitoring model is strongly supported in MarkLogic through its effective integrated auditing and system monitoring capability.

Various kinds of actions related to data access and updates, configuration changes, administrative actions, code execution, and changes to access control can all be audited, including both successful and failed activities. These features can be used to perform the auditing-related activities listed above.

Tiered Storage

All storage media are not created equal. Fast but expensive storage is great for high-value documents and data that are accessed often. Slower but cheaper storage, including storage in the cloud, can be a good fit for older data that is rarely accessed but needs to be kept for historical or auditing purposes. Ideally, a data warehouse will be able to store data in any storage media and allow it to be freely moved from one to another type, for example from SSDs to spinning disk drives to Amazon S3, as the data ages. Tiered storage allows organizations to keep their data warehousing costs low, which means more of the data can be kept accessible, instead of archived, for inclusion in analytics and reporting.

MarkLogic has tiered storage capabilities to automatically store data on the media that's most appropriate. Data can be saved or dynamically moved to different locations based on query constraints that can be simple or unlimited in their complexity – for example, the date the documents were created or last modified, or moved to lower cost storage as they age. By storing data depending on access needs, tiered storage can help users get better performance at lower costs.



Tiered storage offers various operations to maintain your data over time:

- As data ages and you want to move it to lower-cost tiers, you can migrate partitions to different storage locations. Built-in functions and REST endpoints make this easy to do, even between local and shared storage locations.
- Take partitions online and offline. Offline partitions are excluded from queries, updates, and most other operations. You can take a partition offline to archive data and save RAM, CPU, and network resources but bring it back online quickly if you need to query it again.
- Data can even be moved to partitions residing on the Hadoop Distributed File System (HDFS).

Governance

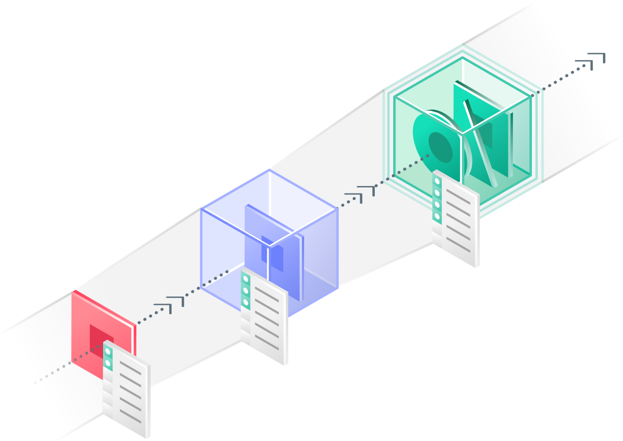
In the same vein as auditing and archiving with tiered storage is data governance support. Data governance has always been essential in government and healthcare, but is even more so with newer laws such as General Data Protection Regulation (GDPR) in Europe or California's Consumer Privacy Act (CCPA). Data governance concerns such questions like:

- Where did the data come from originally?
- Who updated what in the document?
- What did it look like before it was modified?
- Who has rights to see what fields?
- What application code was used to transform the record and with what version?
- How long should the record exist before it is archived?
- What compliance policies affect this record?
- Did that person opt-in to share their data, or opt-out to store their PII?

Traditional data warehouses lack the ability to easily store, view, and understand data about data. With a traditional data warehouse, this data is sometimes stored in a separate metadata catalog, in additional security tables in the warehouse, in new columns in existing tables, or frequently just not stored at all.

It makes the most sense to just store data and data governance information together. Because MarkLogic has a flexible schema and essentially unlimited support for metadata, adding any new data governance information is simple. The information can be added inline, to a metadata section of an XML or JSON document, as RDF metadata, or in a related properties document. Current values along with historical values can reside in the same document, along with who changed what when, which allows for easy debugging or problem solving when the record is not what one expects. For example, if a person modified a record incorrectly, a call center person with appropriate credentials viewing the record would be able to see who made the last modification and when, along with any historical values. Even if data was modified programmatically by a bot, all the bot's actions, even the bot code itself including all versions could be maintained in the same MarkLogic database platform. Inline tagging for access control can prohibit the wrong users from seeing or apply compliance policies to PII/PHI fields. Organizations' governance policies themselves can be stored alongside the data.

In short, everything required to support full auditing, accountability, governance, and security can and should exist in one unified platform. Robust data governance metadata and policies actually



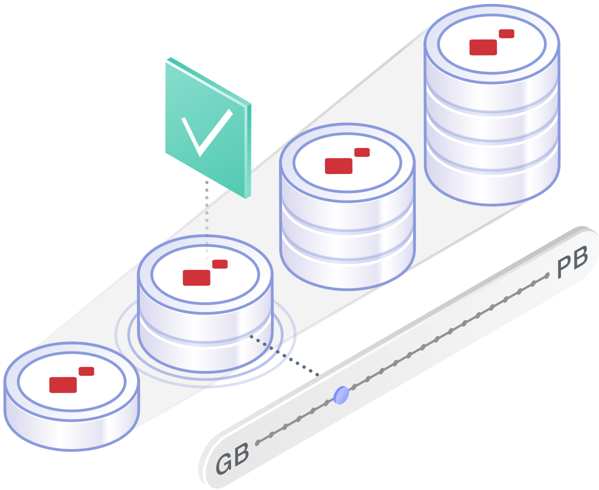
promote safe data sharing as opposed to locking down everything completely for fear of data breaches or lawsuits. Safe data sharing is critical for value-based care and researching Social Determinants of Health over a wide variety of data sources.

Highly Scalable

Data warehouses inherently need to be able to store large amounts of data. They should be able to grow continuously without requiring re-architecting or larger so-called “big metal” computers to support sustained growth. Ideally, they should support scale-out in the same manner that Google can grow to support a bigger internet by simply adding more computing hardware, ideally commodity hardware.

To achieve speed and scale beyond the capabilities of one server, MarkLogic clusters across commodity hardware connected on a LAN. A commodity server can be anything from a laptop (where much development usually happens), to a simple virtualized instance, all the way up to a high-end box with many CPUs—each with 16 cores, 512 gigabytes of RAM, and either a large local disk array or access to a SAN. A high-end box like this can store terabytes of data.

In addition to scale, clustering also enables high availability. In the event that a node should fail, that subset of the data needs to be brought online by another node. This can be done by using either a clustered file system (allowing another node to directly access the failed node's storage and replay its journals) or intra-cluster data replication (replicating updates across multiple node disks, providing in essence a live backup).



AI and Advanced Analytics Support

All Machine Learning/Artificial Intelligence (ML/AI) systems need quality, well-governed data. Without quality data inputs, the outputs are meaningless, if not damaging. ML/AI systems need quality training data, production data, and the associated metadata. Curating all of that data takes a specialized team of data scientists and developers to integrate the data, prep it, build the right models, and then deliver predictions. Traditionally, this process is time consuming, fraught with data quality issues, and lacks data governance. Was the right training data used? Could the training data cause implicit biases? Where did that data come from? Was it first de-identified or does it include PHI/PII and subject to compliance regulations? Those are all questions that need to be answered about the data going into and out of an ML/AI system.

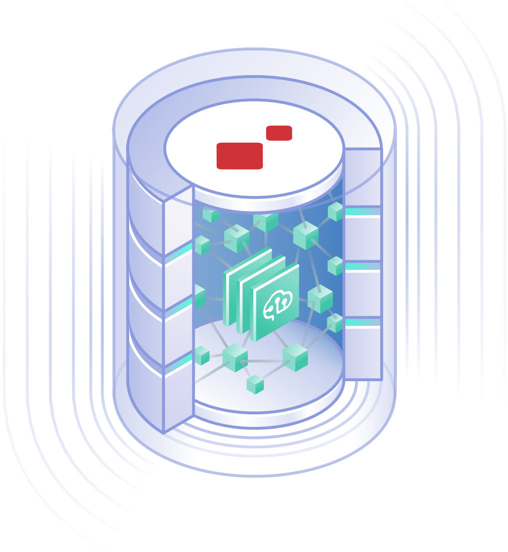
In traditional systems that feed ML/AI, data integration is slow and costly. Rigid schemas may split the data for a customer entity among 20 or more tables. The metadata is not stored with the data, and critical provenance and lineage information may be missing completely. If it is impossible to trace back an AI answer to the data that contributed to that answer, the findings are unlikely to be legitimate.

MarkLogic's approach to curating data is better, faster, and cheaper. And, MarkLogic is smarter than the average database, with AI technology built into the core platform to improve search and discovery. One of the overarching goals of all ML/AI systems is to make machines smarter. That means providing better answers to harder questions. Unfortunately, most databases are really not that smart. The fact that relational databases require specifications in advance for which one or two columns to query seems antiquated in the age of Google. MarkLogic doesn't have those restrictions, and its "Ask Anything" Universal Index and semantic capabilities differentiate it from the competition. The flexibility in indexing also provides a unique ability to save simple to complex queries as an index, and then alert, in real-time, when new data comes in that is a match. MarkLogic's Smart Mastering performs a number of algorithms for matching and merging data, ensuring the best quality of training and production data from the beginning. Mathematical functions including sophisticated sampling by query and confidence scores are supported to assist with finding the optimal training set. MarkLogic's built-in SVM classifiers, ontology-driven entity extraction, and clustering (k-means or lsi) queries, are all foundational capabilities of the platform which help organize and tag data.

MarkLogic's Embedded Machine Learning is also built right into the core of the MarkLogic database. ML/AI routines can run close to the data, in parallel across a MarkLogic cluster, under the umbrella of a secure environment. For data scientists, it's now simpler to just do the work of training and executing models right inside MarkLogic. In MarkLogic 10, the Microsoft Cognitive Toolkit (CNTK) functions are built-in functions. The following network types are supported:

- Feed-forward deep neural networks
- Convolutional neural networks (CNN)
- Recurrent neural networks (RNN), including Long-Short Term Memory (LSTM)

The machine learning toolkit has been designed for peak performance on both CPUs and GPUs and scales to multi-machine-multi-GPU systems. These operations can be executed on a MarkLogic cluster, distributing a training load over all available computing resources.



For instance, computations can be executed on both CPU and GPU devices, including support for multiple, asynchronous parallel evaluation requests. Evaluations are executed as part of the MarkLogic transaction, meaning they are fully ACID compliant and adhere to the MarkLogic security model. MarkLogic embedded Nvidia's CUDA libraries, providing the ability to leverage the powerful computing capabilities of graphics processors for machine learning operations. By leveraging this common platform, GPU-accelerated applications can be developed and deployed on laptops and desktops in an on-premises data center and in the cloud.

Lastly, MarkLogic includes the ONNX runtime, making it possible to deploy models developed with other frameworks in MarkLogic. ONNX is an open format with a large ecosystem that makes machine learning more accessible and valuable to all data scientists. Models can be trained in one framework and transferred to another for execution. This prevents tool or ecosystem lock-in and makes the sharing of models more universal. ONNX models are currently supported in Microsoft Cognitive Toolkit, Cafe2, MXNet and PyTorch, and there are connectors for the most popular frameworks like TensorFlow.

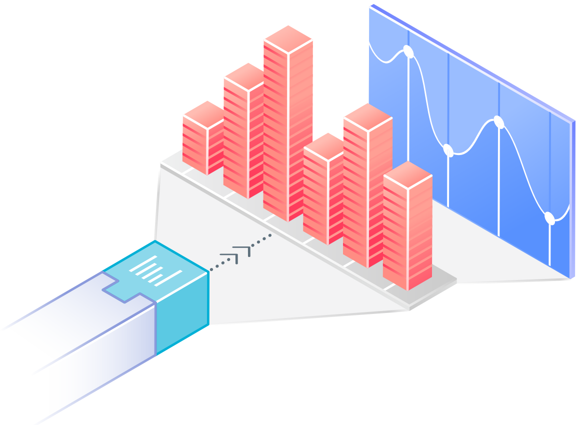
In addition to frameworks, ONNX also supports a wide variety of tools in the machine learning toolchain with a number of available converters, runtimes, compilers and visualizers available.

MarkLogic's machine learning libraries along with a MarkLogic healthcare operational data warehouse, in one platform, provide healthcare and human services organizations the ability to operationalize data, support research, and run predictive analytics to help intervene and avoid negative outcomes. The same 360-degree view the call center representative needs to service a claim dispute can also be used by an embedded neural network to predict behavior. Having access to more information, including the varied Social Determinants of Health data, allows ML/AI to train on the most comprehensive set of data possible, and it is well known that more data provides the most accurate models and even trumps even a better algorithm. The results of ML/AI can be directly saved as metadata, or the results can be translated to a query form and saved to trigger future alerts as soon as new matching data comes in. Therefore, MarkLogic offers the easiest and quickest integration between ML/AI results and operational effects to programs, which in turn may reduce events that tend to correlate with deteriorating health.

BI Support

It is critical that any data warehouse (healthcare data warehouses included) integrate with reporting or business intelligence (BI) tools. Most BI tools are designed to work with relational databases, but we've already stressed that next-generation healthcare data warehouses need to be able to store and query a wide range of data types that don't lend themselves to be easily queried using the ODBC, SQL-based interfaces that BI tools expect. Because relationally-oriented BI tools are ubiquitous and organizations have usually invested heavily in them, the next-gen healthcare data warehouse needs to be able to expose all of its data to these tools.

MarkLogic is a multi-model database platform, where the unit of storage and indexing is a document storing text, XML, JSON, or semantic triples. The document model makes it possible to express rich, related, varying structures – anything from a medical journal article to an enriched doctor's or nurse's notes, to a complex HL7 v3 patient health record. Users want to view parts of these rich structures as though they were simple tables – to see the data in those documents through a relational lens. It is possible to create and query SQL views, define Templates, or use the highly flexible Optic API all of which can be used with external BI tools to report on all of the data in a data warehouse regardless of data type or format.



SQL views can be created and queried through a SQL interface for integration with BI tools using MarkLogic's built-in ODBC application server. Templates are a simple, powerful way to specify a relational lens over documents, making parts of your document data accessible via SQL without changes to the document. In short, MarkLogic supports SQL queries over structured data, JSON, XML, or RDF. The Optic API supports all the things users have gotten used to in the relational world including joins and calculate aggregates across documents. But it also allows much more. With the MarkLogic Optic API, users can query across rows in exposed SQL Views, triples, and documents to basically combine the strengths of multi-model data and query into a single interface. Because BI can be performed over any data format, there is little justification for exporting data to a stand-alone BI data warehouse separate from the operational platform.

Enterprise Data Warehouse Comparison

A summary comparison between an Enterprise Data Warehouse (EDW) and an Operational Data Warehouse (ODW) starts with the schema-agnostic, structure-aware database that allows data from any source to be loaded as is without advance knowledge of its structure. An ODW is real-time interactive while an EDW is batch-oriented, often requiring users to wait weeks or even months between new data loads of an existing type, while new data source loads can take far longer for reasons covered earlier in this paper. An EDW provides after the fact analysis, instead of being able to support business operations with two-way analysis read and write and transaction support.

Traditional EDWs are focused on structured data only, with text relegated to be locked in CLOBS and BLOBs, while an ODW gives you access to all your data – unstructured, structured and semantically linked data. EDWs are model- and ETL-dependent, where the ODW uses a load as is approach, with auto-indexing of text and structure, immediate data discovery and modeling within the ODW for incremental gains that support an agile approach. Lastly, the EDW is reactive and query-based versus proactive with massively scalable alerting supported by queries of any complexity.

Faster Time to Market With Enterprise Reliability

Two top 5 healthcare Payers have built operational data warehouses on MarkLogic in 10-20% of the time that would have been required with relational-based data warehouses. Additionally, several other large managed care organizations are getting ready to build their next-generation healthcare data warehouses on MarkLogic. In addition to the time – and therefore cost – savings, they chose MarkLogic because enterprise reliability and security are critical and the 360-degree views provided by their operational data warehouses would help them with their audacious goals, that for one organization included doubling revenue through technological innovation.

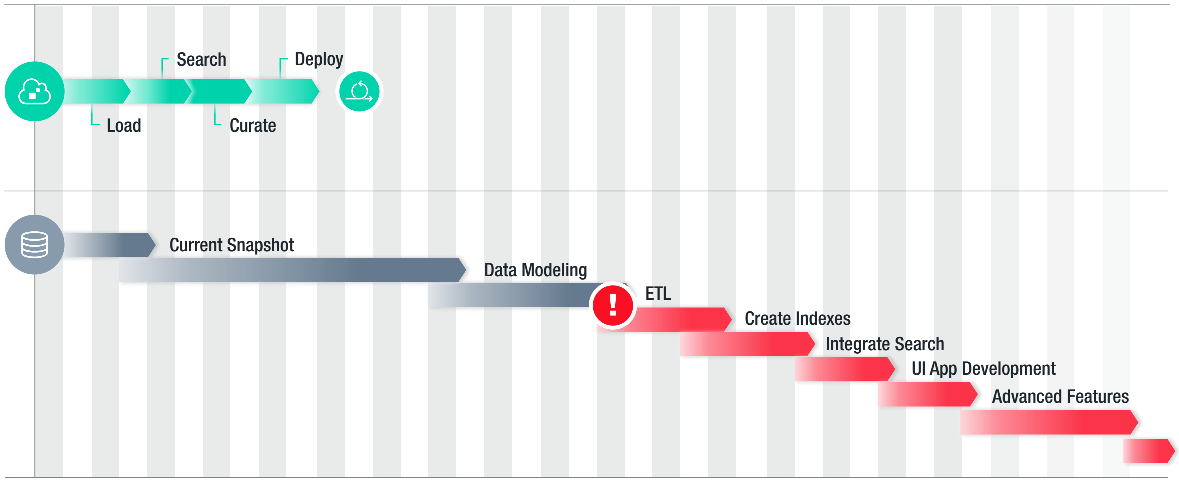


Figure 4: An ODW based on MarkLogic can be deployed much faster than a traditional EDW, and is more flexible so it can readily support new business requirements

Conclusion

Traditional data warehouse approaches are fine for analytics on data that does not change very often, has moderate volume, need not be accessed in real-time, and ignores large quantities of unstructured data.

However, with the extreme volume, velocity and wide variety of healthcare and social determinants data, new database technologies are needed to reduce risk and future-proof operations. Next-generation healthcare operational data warehouses need to be able to be agile without sacrificing any of the enterprise features that have become table stakes for any enterprise technology – features like element-level government-grade security, database encryption with separate key management, ACID transactionality, high availability with failover, and disaster recovery. These enterprise features are particularly critical when dealing with personal health information security, and together with all the other features previously discussed, represent the ideal feature set for next-generation healthcare and human services data warehouses.

About MarkLogic

Data integration is one of the most complex IT challenges, and our mission is to simplify it. The MarkLogic Data Hub is a highly differentiated data platform that eliminates friction at every step of the data integration process, enabling organizations to achieve a 360º view faster than ever. By simplifying data integration, MarkLogic helps organizations gain agility, lower IT costs, and safely share their data.

Organizations around the world trust MarkLogic to handle their mission-critical data, including 6 of the top 10 banks, 5 of the top 10 pharmaceutical companies, 6 of the top 10 publishers, 9 of the 15 major U.S. government agencies, and many more. Headquartered in Silicon Valley, MarkLogic has offices throughout the U.S., Europe, Asia, and Australia.

MARKLOGIC CORPORATION
999 Skyway Road, Suite 200 San Carlos, CA 94070
+1 650 655 2300 | +1 877 992 8885 | www.marklogic.com | sales@marklogic.com

MarkLogic®

999 Skyway Road, Suite 200 San Carlos, CA 94070
+1 650 655 2300 | +1 877 992 8885
www.marklogic.com | sales@marklogic.com