UNIVERSIDAD
DE LA REPÚBLICA
URUGUAY

FACULTAD DE
INGENIERÍA

Calidad de Datos basada en Contexto en *Data Lakes*

Informe de Proyecto de Grado presentado por

José Pedro de León

en cumplimiento parcial de los requerimientos para la graduación de la carrera de Ingeniería en Computación de Facultad de Ingeniería de la Universidad de la República

Supervisores

Adriana Marotta
Flavia Serra

Montevideo, 23 de julio de 2025

Agradecimientos

A mi familia, gracias por acompañarme durante toda mi trayectoria estudiantil. Mamá y Papá, gracias por todas las oportunidades que me dieron para tener una buena educación. Belén y Clara, gracias por su compañía, y por todas las veces que me ayudaron a distenderme cuando el estrés me sobrevino.

A mis amigos, gracias por siempre estar ahí, en las buenas y en las malas. Muchas gracias por todas las risas que compartimos, y por sus palabras de aliento que me propulsaron hasta el final.

A los amigos que conocí por esta carrera, no pasa un día que no agradezca que nuestros caminos se hayan cruzado. Gracias por todas las horas de estudio llenas de risa, la carrera no hubiera sido lo mismo sin ustedes.

Y a mis tutoras, Adriana y Flavia. Muchas gracias por alentarme siempre y acompañarme en este proceso. Gracias por el soporte que me dieron, tanto académico como emocional. Gracias por ayudarme a que este proyecto fuera lo mejor que puede ser.

Resumen

En la actualidad, el uso de tecnologías de *Big Data* se ha vuelto cada vez más presente en diversos sectores, llevando a que las empresas deban basar sus estrategias en decisiones tomadas a partir de grandes volúmenes de datos. A medida que crece la dependencia en estos sistemas, también se vuelve fundamental contar con herramientas y estrategias que permitan evaluar la calidad de los datos utilizados, ya que una mala calidad puede generar análisis erróneos y afectar el valor obtenido de los datos. Sin embargo, el estudio sistemático de la calidad de los datos en entornos de *Big Data* sigue siendo limitado y representa un área de investigación en desarrollo.

Este proyecto se basa en dos trabajos previos: el proyecto de grado de (Cortés, 2024), que propone una arquitectura genérica de *Big Data* con capacidades de gestión de la calidad de los datos, y la tesis de doctorado de (Serra, 2024), que propone el modelado del contexto de los datos y la definición de modelos de calidad de datos que consideran el modelo de contexto.

En este trabajo se unifican ambas propuestas, brindando la capacidad de incorporar modelos de contexto al proceso de gestión de calidad de los datos, en la arquitectura genérica de *Big Data*. Para ello, se realiza un análisis sobre la incidencia de los componentes de contexto sobre los datos en las distintas zonas del *Data Lake*, a través del cual se identifica un nuevo componente de contexto, denominado "*Data Lineage*". En base a este análisis, se proponen modificaciones al modelo de metadatos de la arquitectura, especificando las nuevas entidades y relaciones necesarias para poder representar modelos de contexto y modelos de calidad de datos basados en ellos. Para demostrar la viabilidad de nuestra propuesta, se diseñó e implementó una prueba de concepto, utilizando el DBMS de bases de datos de grafos *Neo4j*, para la implementación de los metadatos de calidad de datos y de contexto.

Palabras clave: Big Data, Data Lake, Calidad de Datos, Contexto de los Datos

Índice general

Capítulo 1

Introducción

En este capítulo se presenta la motivación y descripción del proyecto, los objetivos planteados y los resultados esperados.

1.1. Motivación

En el mundo moderno, las empresas se enfrentan a un entorno altamente competitivo y dinámico que exige la toma de decisiones rápidas y basadas en información confiable. Como resultado, el manejo y análisis de datos se ha convertido en un aspecto crucial para su desarrollo y sustentabilidad. Sin embargo, con el avance de las tecnologías digitales, las organizaciones generan y recopilan volúmenes de datos cada vez mayores provenientes de diversas fuentes, como transacciones comerciales, redes sociales, sensores con capacidades *IoT*, o sistemas internos de las empresas.

El concepto de "***Big Data***" hace referencia a estos grandes volúmenes de datos, y ha supuesto un gran desafío en el desarrollo de sistemas de información que permitan procesar y extraer valor de estas grandes cantidades de datos heterogéneos de forma eficiente. Por este motivo, en las décadas recientes han surgido varias propuestas de arquitecturas de *Big Data*, que buscan satisfacer la necesidad de almacenar cantidades elevadas de datos con estructuras diferentes, pero manteniendo los costos bajos. En base a estas necesidades, surge el concepto de *Data Lake* (DL), que consiste en un almacenamiento *low-cost*, en repositorio central, en el cual se almacenan todos los datos, sin importar su fuente o su formato, y que permite procesar estos datos con gran rapidez (Harby y Zulkernine, 2022).

Sin embargo, la flexibilidad de estas arquitecturas también hace que sean susceptibles a presentar problemas de calidad de datos. Esto se debe a que, al permitir la ingestión de datos de varias fuentes sin una estructura o validación estricta previa, es común que se acumulen datos incompletos, inconsistentes, duplicados o irrelevantes. Esta falta de control puede derivar en lo que se conoce como un *"data swamp"*, donde la utilidad de los datos almacenados en el DL se

ve comprometida, dificultando el análisis y la obtención de datos confiables. Para evitar estas situaciones, cobra importancia la implementación mecanismos y estrategias efectivas de gobernanza y gestión de calidad dentro de estos entornos.

La **gestión de la calidad de los datos** es un proceso integral que abarca un conjunto de tareas que se enfocan en el manejo adecuado de los datos, con el objetivo de que los datos tratados se ajusten lo mejor posible al caso de uso al que se aplican. En años recientes, se ha introducido el concepto de *contexto de los datos*, que refiere a la influencia que tiene la realidad asociada a los datos y su uso sobre la evaluación de su calidad. A partir de esto, se habla del concepto de *calidad de datos basada en el contexto* (Strong, Lee, y Wang, 1997).

Si bien existen metodologías generalmente aceptadas en la bibliografía para la gestión de la calidad de los datos en sistemas de información estructurados (como los son los *Data Warehouse*, DW), no existen técnicas globalmente aceptadas para la gestión de la calidad en *Big Data* (B. Inmon, Levins, y Srivastava, 2021).

1.2. Descripción del proyecto

Este proyecto de grado se encuentra enmarcado en el proyecto "Calidad de Datos en la Preparación para el Análisis de *Big Data*", financiado por la Comisión Sectorial de Investigación Científica (CSIC) (Grupo Gema, InCo, 2023), el cual es desarrollado por el grupo de investigación Gema (Gestión, Modelado y Análisis de Datos), del Instituto de Computación de la Facultad de Ingeniería de la Universidad de la República. Este proyecto busca generar soluciones para la gestión de calidad de datos en sistemas de información de *Big Data*.

Para el desarrollo de este proyecto, se toma como base una propuesta de arquitectura genérica de *Big Data* presentada en el proyecto de grado de (Cortés, 2024), que se encuentra desarrollado en el marco del proyecto del grupo Gema, y la propuesta de gestión de calidad de datos basada en el contexto desarrollada en la tesis de doctorado de (Serra, 2024).

Este proyecto se realizó de manera simultánea con el proyecto de grado de (Castro, Gómez, y López, 2025), que también se desarrolló en el marco del proyecto del grupo Gema, y utiliza la misma arquitectura. Por este motivo, en el transcurso de este proyecto se buscó que ambas propuestas fueran compatibles entre sí, para permitir una eventual unificación de ambos resultados.

1.3. Objetivos

El objetivo general de este proyecto es extender la arquitectura de *Big Data* existente, agregándole modelos de contexto, y permitiendo la definición y ejecución de modelos de calidad de datos basada en contexto.

Para alcanzar este objetivo, se definen los siguientes objetivos específicos:

1. Estudiar la bibliografía existente sobre arquitecturas de *Big Data* y *Data Lakes*.

2. Estudiar en profundidad la arquitectura genérica de *Big Data* utilizada en el proyecto del grupo Gema, haciendo foco en la gestión de la calidad de los datos que esta incluye.

3. Estudiar en profundidad la propuesta existente de gestión de calidad de datos basada en el contexto presentada en (Serra, 2024).

4. Proponer una extensión de la arquitectura genérica de *Big Data*, que permita la definición y el uso de modelos de contexto y de modelos de calidad de datos basados en contexto.

5. Implementar la propuesta y aplicarla a un caso de estudio.

1.4. Resultados esperados

En la elaboración de este proyecto, se espera obtener los siguientes resultados:

1. Arquitectura de *Big Data* que permita la definición de modelos de contexto y de modelos de calidad de datos basada en contexto.

2. Prototipo implementado de la arquitectura propuesta.

3. Prueba de concepto de la solución propuesta, mediante su aplicación a un caso de estudio.

1.5. Organización del documento

Este informe se organiza en 6 capítulos, incluyendo el actual. En el capítulo 2 se presenta el Marco Teórico, en el que se presentan los conceptos y definiciones básicas de la bibliografía que son necesarios para la comprensión de este proyecto. En el capítulo 3 se presentan los dos trabajos que sirven como fundamento principal para la elaboración de la propuesta realizada. En el capítulo 4 se presenta la propuesta de calidad basada en contexto para el *Data Lake*. En el capítulo 5 se plantea un caso de estudio ficticio para ejemplificar la implementación de la propuesta presentada en el capítulo 4. Por último, en el capítulo 6 se detallan las conclusiones y el trabajo a futuro.

Capítulo 2

Marco teórico

En este capítulo se detallan los conceptos y nociones básicas necesarias para la comprensión de este trabajo, incluyendo la definición de los conceptos principales de *Big Data* y de Calidad de Datos.

2.1. Conceptos de Big Data

El concepto de **Big Data** refiere a colecciones masivas de datos heterogéneos, que por su tamaño y complejidad requieren de sistemas diferentes a las bases de datos tradicionales para ser almacenados y procesados (Ramchand y Mahmood, 2022).

Tradicionalmente, el concepto de *Big Data* está relacionado a tres 'Vs' (Laney, 2001). Sin embargo, por los cambios recientes en la naturaleza de los datos procesados en los negocios, se agrega una cuarta (Schroeck, Shockley, Smart, Romero Morales, y Tufano, 2012), por lo que se definen las cuatro 'Vs':

- Volumen: Refiere a la masividad de los datos almacenados y procesados.
- Velocidad: Refiere a la frecuencia con la que se generan nuevos datos y se procesan y utilizan los datos existentes.
- Variedad: Refiere a la diversidad de los datos que se almacenan, tanto en estructura como en contenido.
- Veracidad: Refiere a la calidad y la precisión de los datos almacenados, que afecta el nivel de confianza en las decisiones que se toman sobre ellos.

Para cumplir con el requisito de almacenar los grandes volúmenes de datos, surgen distintos sistemas con diferentes capacidades.

2.1.1. Data Warehouse

En la década de los 90' se popularizó el uso de *Data Warehouses* (DWs) como repositorios de datos centrales para empresas. Consiste en una base de

datos relacional, dedicada al uso en aplicaciones de toma de decisiones y análisis de negocio.

El foco de los DW es el almacenamiento de datos estructurados de manera específica, por lo que tienen un esquema general predefinido. Esta estructura está diseñada con el objetivo de poner a disposición los datos para tareas de análisis y de inteligencia de negocio (*Business Intelligence*, BI, en inglés). Los datos que ingresan al DW provienen de fuentes pre-existentes, y son sometidos a un proceso de extracción, transformación y carga (*extract, transform and load*, ETL) para ser limpiados y adecuados al esquema del DW. (Yessad y Labiod, 2016)

Bill Inmon, uno de los primeros autores en escribir acerca del concepto de DW, identifica una serie de características que poseen: (W. H. Inmon, 2002)

- *Subject-oriented* (orientado a temas): Los datos se vinculan a la realidad de negocio de la empresa que es propietaria del DW, y están organizados y estructurados de acuerdo a la función que van a cumplir.
- *Integrated* (integrados): Los datos que se alojan en el DW que provienen de distintas fuentes, son sometidos a procesos que normalizan su estructura, y resuelven problemas de heterogeneidad de sus esquemas y potenciales diferencias o discrepancias en su contenido.
- *Time-variant* (Variables en el tiempo): Los datos se identifican por su fecha, y se guarda un historial de los cambios en los datos del DW (datos históricos).
- *Nonvolatile* (no-volátil): Los registros del DW no son modificables. Una vez que se cargan, se utilizan para realizar consultas del tipo *read-only*. Los cambios en los datos se representan con registros nuevos, y se conservan los datos históricos.

En los modelos habituales de DW, los datos se acceden a través de *Data Marts*, que son agrupaciones de los datos (virtuales o materializadas) que dividen los datos del DW según unidades de negocio o propósito de análisis. Los *Data Marts* tienen un esquema específico, diseñado y ajustado al uso que se le dará a los datos (W. Inmon, 2001).

Algunos de los ejemplos de sistemas de DW y herramientas comerciales para su gestión son los siguientes:

- Amazon Redshift (Amazon, 2024a)
- Microsoft Azure Synapse Analytics (Microsoft, 2024b)
- Google BigQuery (Google, 2024a)
- IBM Db2 Warehouse (IBM, 2024c)
- Pentaho Data Inegration (Pentaho, 2025)

2.1.2. Arquitecturas de *Data Warehouse*

A continuación se presentan algunas de las arquitecturas para sistemas de DW propuestas por los autores en distintas generaciones.

Modelo de Inmon

En la década de los 90, Bill Inmon propone un modelo en el que separa el ambiente de los datos de una compañía en cuatro niveles (Figura 2.1).

En primer lugar están los datos operacionales, que son los datos generados diariamente para registrar eventos y transacciones de la compañía. Esta información ingresa al DW atómico a través de un proceso ETL. La información del DW, luego, es dividida en *Data Marts*, que son entidades separadas del DW. Cada *data mart* corresponde a una unidad de negocio. El último nivel de los datos está conformado por la nueva información que se generan por los individuos que analizan y explotan los datos a través del uso de herramientas dedicadas. Esta nueva información incluye reportes, la generación de *dashboards* y gráficas, entre otras cosas (Yessad y Labiod, 2016).

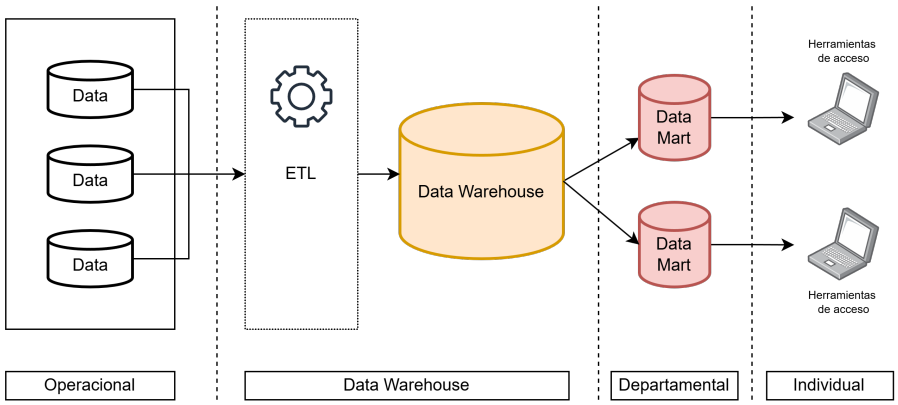

Figura 2.1: Modelo de DW de Inmon

Modelo de Kimball

Ralph Kimball propone una variación sobre el modelo de Inmon (Figura 2.2), basado en el concepto de modelado dimensional, donde los datos del DW son organizados en dimensiones que funcionan como 'ejes' para realizar distintos análisis.

Los datos se almacenan en tablas de hechos ('Fact tables'), que registran los eventos y datos observables que son pertinentes a la realidad estudiada (por ejemplo, la información de una venta). Los registros de la tabla de hechos pueden ser agrupados según las dimensiones definidas.

Una distinción importante del modelo de Kimball en comparación al de Inmon, es que Kimball considera que los *data marts* son componentes del DW, y de hecho el DW es la composición de todos los *data marts* (Yessad y Labiod, 2016).

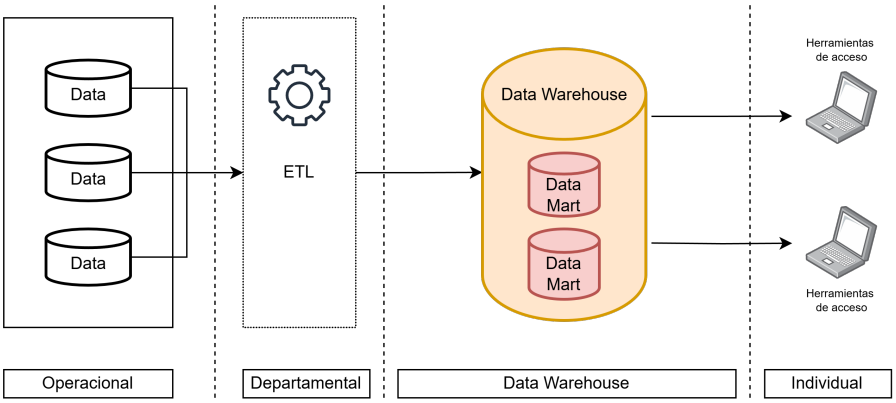

Figura 2.2: Modelo de DW de Kimball

Modelo de Data Vault

En la década del 2000, Dan Linstedt propone un modelo muy diferente al de Inmon y Kimball, el modelo de *Data Vault*. Este modelo hace énfasis en el registro del linaje de los datos (de dónde provienen), y también prioriza la posibilidad de adaptarse fácilmente a cambios en el negocio.

En el modelo de *Data Vault*, existen tres entidades centrales, que se representan en distintos tipos de tablas (Fentaw, 2014):

- *Hubs*: tablas que alojan claves que identifican a distintas entidades de negocio, como pueden ser clientes, productos, empleados, etc. No contienen atributos descriptivos. Estas claves son estables, es decir que no se espera que cambien en el tiempo.
- *Links*: tablas que mantienen asociaciones entre *hubs*. Representan interacciones o transacciones entre entidades de negocio. Tiene su propia clave, pero tampoco contienen atributos descriptivos.
- Satélites: tablas que mantienen información acerca de los atributos ligados a los *hubs* y *links* al momento de su carga en el *Data Vault*.

En esta arquitectura (Figura 2.3), los datos operacionales primero transitan por una zona de preparación ('*staging*'), antes de introducirse en el *Raw Vault* (también denominado '*Enterprise Data Warehouse*', EDW), donde se realiza un proceso ETL para obtener las tablas de *Hubs*, *Links* y *Satellites*. Dentro

de la *Raw Vault*, puede implementarse una '*Business Vault*' (o '*Business Data Warehouse*', BDW), que almacena resultados pre-calculados de métricas de negocio. Por último, se compone una capa de presentación mediante un conjunto de *data marts* creados por modelado dimensional, tomando datos tanto del *Raw Vault* como del *Business Vault*. Estos *data marts* también proveen cubos OLAP para ser utilizados para tareas de análisis y generación de reportes (Yessad y Labiod, 2016).

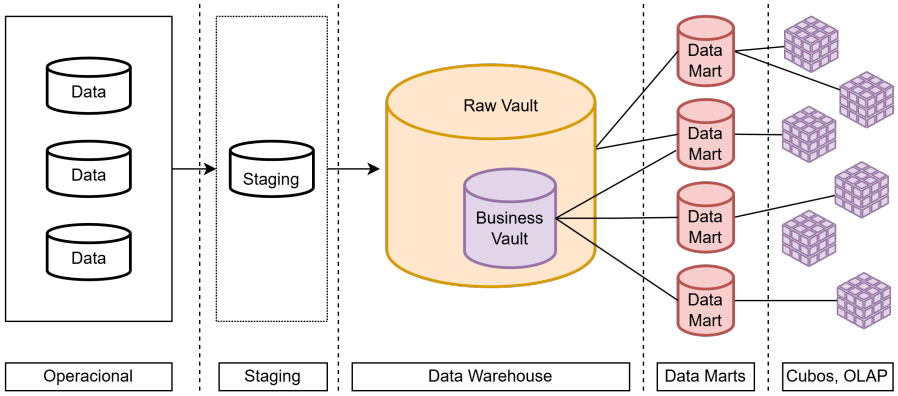

Figura 2.3: Modelo de DW de Data Vault

2.1.3. Data Lake

Con el pasaje del tiempo y la evolución de los usos de la tecnología en la vida cotidiana, surgen nuevas necesidades de negocio que exigen el procesamiento de volúmenes exponencialmente mayores de datos, a altas velocidades, y de diferentes formatos. Por ejemplo, las innovaciones en los dispositivos de IoT, y el crecimiento de la popularidad de las redes sociales, significa la generación constante de datos heterogéneos (archivos de texto, audio, audiovisuales, etc.) que deben ser ingeridos, procesados y relacionados a altas velocidades con tal de generar valor. Frente a estas necesidades de negocio, el DW presenta limitaciones tanto en su espacio de almacenamiento, así como en su poder de procesamiento, y su rigidez ante la estructura de los datos almacenados (Harby y Zulkernine, 2022).

En base a estas necesidades de almacenar grandes volúmenes de datos con estructuras heterogéneas, con procesamiento de alta velocidad, surgen los sistemas *Data Lake* (DL).

El autor Ben Sharma define al DL como 'un repositorio central en el cual se almacenan todos los datos, sin importar su fuente o su formato' (Sharma, 2018).

A diferencia de los DW, que tienen un enfoque de almacenamiento de datos estructurado y centrado en la definición de esquemas específicos, el enfoque de

los DL se centra en almacenar grandes cantidades de datos que se procesan con gran rapidez, sin importar su estructura.

El uso de DLs presenta una gran variedad de ventajas técnicas frente al uso de DWs. Principalmente, la capacidad del DL de almacenar datos en su formato original ('crudos' o '*raw*'), sin forzar la compatibilidad con esquemas predefinidos, permite mayor flexibilidad en la manipulación de los datos, y adicionalmente permite tener a disposición un flujo de datos mayor para análisis. Además, la tecnología de DL responde a la necesidad de negocio de ser altamente escalable, tanto en costos como en tiempo de procesamiento, frente a crecimientos exponenciales en las cantidades de datos ingeridos (Sharma, 2018).

Algunas soluciones comerciales de DLs incluyen:

- Amazon S3 (Simple Storage Service) (Amazon, 2024b)
- Microsoft Azure Data Lake Storage (Microsoft, 2024a)
- Google Cloud Storage (Google, 2024b)
- IBM Cloud Object Storage (IBM, 2024b)

Sin embargo, los DLs presentan una serie de desventajas relacionadas a la dificultad de gestionar volúmenes tan altos de datos sin estructura. Principalmente, si existen demoras en el procesamiento de los datos 'crudos' que ingresan al DL, este tiene el potencial de convertirse en lo que se conoce como un '*data swamp*', que refiere a que, a pesar de almacenar muchos datos, estos no se explotan, no se utilizan, o quedan rápidamente obsoletos. Este riesgo puede ser mitigado si se tiene un gran poder computacional y un alto grado de paralelismo en el procesamiento de los datos ingeridos, y si se realiza un manejo eficiente y extenso de metadatos (Harby y Zulkernine, 2022).

2.1.4. Arquitecturas de *Data Lake*

En esta sección se presentan algunas de las arquitecturas para el modelado de sistemas de DL propuestas en la bibliografía.

Two Layered architecture

La primera arquitectura definida, la más simple, consiste en dos capas (Figura 2.4):

- *Landing Zone*: zona de tránsito de los datos, es el punto de acceso inicial de los datos al DL. Esta capa realiza el proceso de extracción de los datos de las fuentes, y los traslada a la zona de *Raw Data*.
- *Raw Data*: zona de persistencia de los datos, en su formato original.

Este modelo no exige la definición de una estructura sobre los datos ingeridos. Por lo tanto, es especialmente útil para datos de alta variedad que se

acceden principalmente a través de consultas *ad-hoc* (Hlupić, Oreščanin, Ružak, y Baranović, 2022).

Su principal desventaja es el bajo refinamiento de los datos, al únicamente almacenar los datos en su formato original. Como consecuencia, la responsabilidad de adaptar los datos al caso de uso es delegada a los usuarios o aplicaciones que consuman los datos.

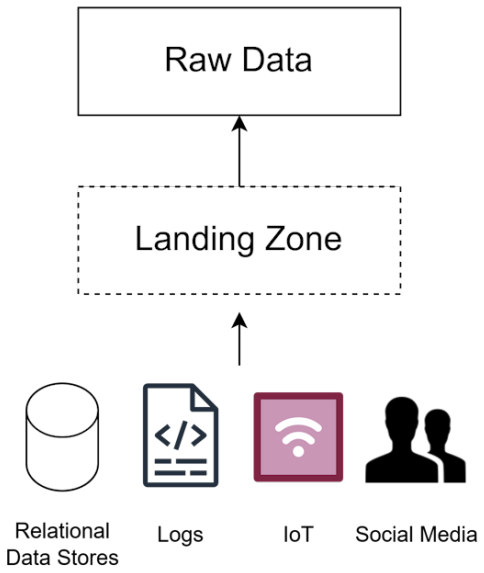

Figura 2.4: *Two Layered Data Lake architecture*

Multi-Layered architecture

Como una evolución del modelo *Two Layered*, se presenta el modelo *Multi-Layered*, que propone la implementación de cuatro capas (Figura 2.5). La separación de las capas es determinada por una distinción granular de responsabilidades (Hlupić y cols., 2022):

- *Ingestion Layer*: capa encargada de realizar la extracción de datos de las fuentes, y se implementa un proceso automatizado de extracción de metadatos, que se aplica sobre los datos estructurados y semi-estructurados. Estos metadatos se almacenan en un repositorio dedicado (repositorio de metadatos).
- *Storage Layer*: capa de persistencia que contiene el repositorio de metadatos y un repositorio de datos crudos (*raw data*). La complejidad interna en esta capa es ocultada del usuario final, y en su lugar se ofrece una interfaz que provee la capacidad de realizar consultas sobre los datos almacenados.

- *Transformation Layer*: capa encargada de la transformación de los datos para adecuarse a un uso final. En esta capa se implementan procesos de *data cleansing* y *data integration*, y los datos pueden agruparse en *data marts*, para facilitar su consumo. Los metadatos identificados (o generados) por estos procesos son almacenados en el repositorio de metadatos de la *Storage Layer*.

- *Interaction Layer*: capa que provee una interfaz al usuario para acceder tanto al repositorio de metadatos como a los datos transformados en la *Transformed Layer*.

Esta arquitectura es más adecuada que la Two Layer Architecture para casos de uso donde se conoce el propósito que les será dado a los datos por los usuarios finales, y se desea que accedan a datos con un cierto grado de refinamiento. Sin embargo, presenta la desventaja de que toda lectura debe ser realizada a través de la Interaction Layer, por lo que si se desea acceder a los datos del *Raw Data repository* de la *Storage Layer*, debe implementarse una interfaz dedicada a ello en la *Interaction Layer*.

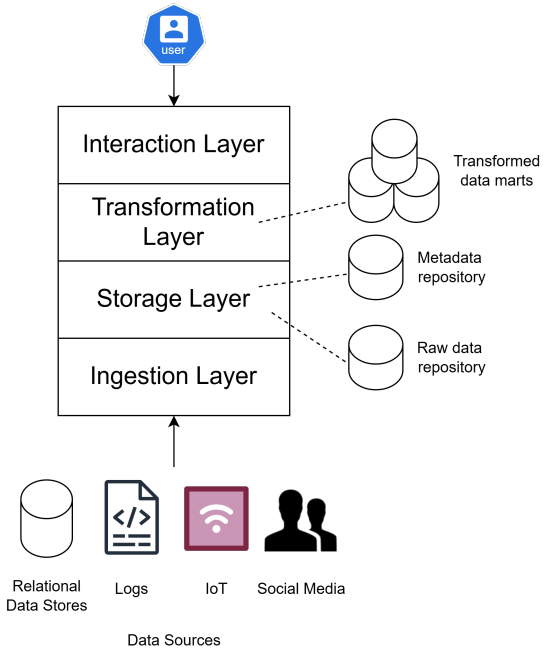

Figura 2.5: *Multi-Layered Data Lake architecture*

Data Pond architecture

La arquitectura de *Data Ponds* (Figura 2.6), propuesta por Bill Inmon (B. Inmon, 2016), define una estructura de zonas de datos (denominados *ponds*), que agrupan los datos según su estructura y su uso. A diferencia de los modelos anteriores, estas zonas no se organizan en una estructura secuencial. La arquitectura define cinco *ponds* (Hlupić y cols., 2022):

- *Raw Data Pond*: zona que ingiere los datos crudos de las fuentes. Funciona como una zona de tránsito, donde los datos se almacenan temporalmente, hasta que sean consumidos por los otros *ponds*, luego de lo cual son eliminados de esta zona. En esta zona, no se realiza ningún tipo de procesamiento de metadatos.
- *Analog Data Pond*: zona donde se almacenan datos semi-estructurados de alta velocidad, típicamente provenientes de flujos de datos constantes (por ejemplo, provenientes de dispositivos de IoT).
- *Application Data Pond*: zona similar a un DW, que almacena datos estructurados, que son ingeridos desde el *Raw Data Pond* a través de procesos tradicionales de ETL.
- *Textual Data Pond*: zona que almacena datos de texto no-estructurados, que se desean conservar para análisis de texto.
- *Archival Data Pond*: zona que se utiliza para almacenar datos inactivos de los *Analog*, *Application* y *Textual ponds*.

Este modelo presenta la ventaja de separar los datos según su uso, y permitir la interacción del usuario final con cualquiera de los *ponds* (a excepción del *Raw Data Pond*). La principal desventaja es la falta de persistencia de los datos 'crudos', que puede ocasionar pérdidas de información.

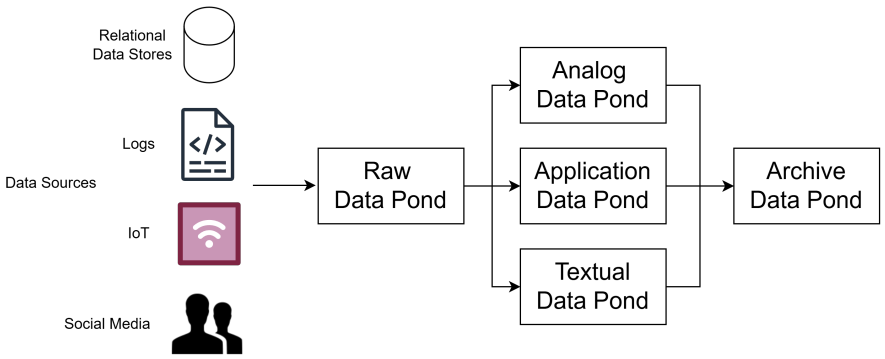

Figura 2.6: *Data Pond Data Lake architecture*

Zaloni Zone architecture

La compañía Zaloni presenta una propuesta de arquitectura de *Zaloni Zones* (Figura 2.7), al igual que la arquitectura de *Data Ponds*, divide los datos del DL en zonas según su estructura y uso (Sharma, 2018). Este modelo presenta cinco zonas (Hlupić y cols., 2022):

- *Transient Landing Zone*: zona de almacenamiento temporal de los datos que ingresan al DL. En esta zona se pueden realizar controles de *data compliance* (cumplimiento de requerimientos técnicos y de negocio) para limitar el ingreso de datos al DL. También se pueden implementar controles de calidad de datos.
- *Raw Zone*: zona que almacena los datos crudos, una vez que fueron aceptados en el DL tras cumplir con los requisitos impuestos en la *Transient Landing Zone*. Los datos son persistidos de forma permanente en su formato original.
- *Trusted Zone*: zona que almacena datos sobre los cuales se realizaron chequeos adicionales de calidad y de *data compliance* con un propósito de negocio específico. La fuente de los datos de esta zona es la *Raw Zone*, y estos son sometidos a procesos de transformación y *data cleansing* para ajustarse a las políticas y necesidades del negocio para el que serán usados.
- *Refined Zone*: zona que almacena datos adaptados a un formato específico para su consumo. Su propósito es disponibilizar los datos a usuarios finales con la estructura adecuada para realizar sus tareas.
- *Sandbox*: zona que permite el relevamiento de datos de cualquiera de las otras zonas para realizar tareas de exploración y análisis de datos *ad-hoc* por *Data Scientists*. Los resultados de estos análisis pueden ser almacenados como datos adicionales en la *Raw Zone*.

Este modelo presenta la ventaja sobre el modelo de *Data Ponds* de persistir los datos crudos en una zona dedicada.

2.1.5. *Data Lakehouse*

Para combinar las características deseadas del DL y el DW y abordar sus debilidades con el objetivo de acelerar la extracción eficiente de conocimientos, surge un nuevo modelo de solución denominado *Data Lakehouse* (DLH). Los autores en (Zaharia, Ghodsi, Xin, y Armbrust, 2021), pertenecientes a la empresa Databricks, proponen una arquitectura para implementar un sistema de DLH que satisface requerimientos de alta accesibilidad, escalabilidad, disponibilidad, distribución jerárquica y una administración de datos integral.

Este modelo consiste en la implementación de una capa de metadatos, indexado y cacheado sobre un DL, y la implementación de APIs para proporcionar funcionalidades de análisis avanzado y de manejo de los datos. En particular,

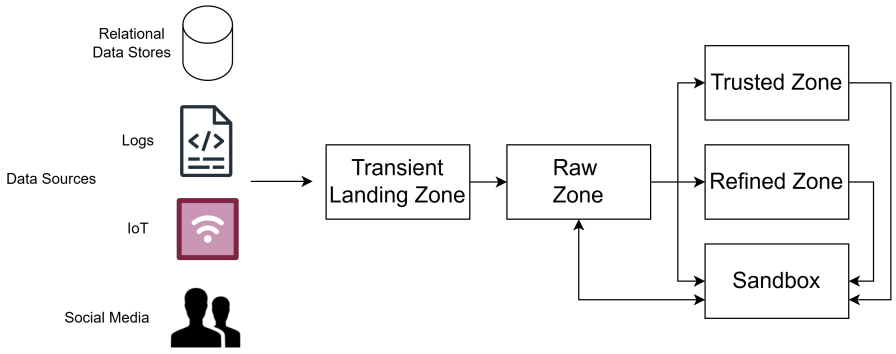

Figura 2.7: *Zaloni Zone Data Lake architecture*

implementan transacciones del tipo ACID (del inglés, *atomic, consistent, isolated, durable*).

La Figura 2.8 muestra una comparación de las tres arquitecturas de *Big Data* discutidas.

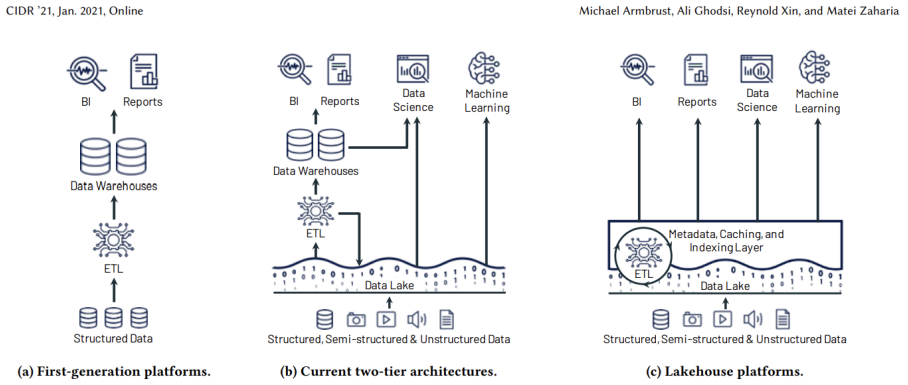

Figura 2.8: Evolución de arquitecturas de *Big Data*.
(Zaharia y cols., 2021)

2.2. Conceptos de Calidad de datos

La Calidad de Datos es el área de investigación que abarca "la totalidad de las características de los datos que influyen sobre su habilidad de satisfacer las necesidades implícitas y explícitas"de los usuarios de los mismos. Esto se

asocia con la noción de adecuación al uso (*fitness for use*) de los datos (Batini y Scannapieco, 2016).

Contar con datos de buena calidad es relevante en una amplia variedad de dominios. Por ejemplo, en escenarios empresariales se procesan grandes cantidades de datos diariamente con el fin de tomar decisiones estratégicas para distintas actividades, como lo pueden ser optimización de procesos de negocio o ajustes de inversiones en campañas de marketing. A continuación, se presentan algunas de las principales ventajas de manejar datos de buena calidad (IBM, 2024a):

- Toma de decisiones informadas: la calidad de los datos influye directamente sobre las decisiones tomadas. Por ejemplo, en un escenario empresarial, la toma de decisiones en base a datos incompletos o erróneos puede resultar en pérdidas de oportunidades de desarrollo, lo que puede generar un impacto negativo financiero para la empresa.

- Eficiencia operativa: contar con datos incorrectos o incompletos puede ralentizar operaciones, aumentar los costos y afectar la productividad de los procesos productivos de las empresas.

- Cumplimiento normativo y reputación: En muchos sectores, la calidad de los datos es crucial para cumplir con regulaciones y estándares legales. Errores en los datos pueden resultar en sanciones y problemas legales. Además, empresas que ponen a disposición sus datos pueden dañar su reputación y perder la confianza de sus clientes y socios si estos datos son de mala calidad.

2.2.1. Definiciones básicas de Calidad de Datos

La calidad de los datos puede ser estudiada y medida según **dimensiones**, que encapsulan distintos aspectos relacionados a la calidad de un conjunto de datos, o bien de los procesos que los manipulan. A continuación se detallan algunas de las dimensiones más frecuentemente utilizadas en aplicaciones generales de Calidad de Datos (Batini y Scannapieco, 2016), (Strong y cols., 1997):

- *Accuracy* (Exactitud): refiere a la distancia entre los datos registrados y los datos reales.

- *Completeness* (Completitud): refiere a la capacidad de representar todos los aspectos de interés de la realidad.

- *Freshness* (Frescura): refiere a qué tan actualizados están los datos.

- *Consistency* (Consistencia): refiere a la capacidad de los datos de adherirse a la realidad de interés sin contradicciones.

- *Uniqueness* (Unicidad): refiere a la no-duplicación de los datos.

- *Amount of data* (Cantidad de Datos): refiere a la cantidad de registros almacenados, y al tamaño general de los datos.

Sin embargo, con el surgimiento de *datasets* cada vez más masivos, especialmente con la introducción de los sistemas de DL, se comienzan a estudiar más dimensiones que están vinculadas con la cuarta 'V´ de *Big Data*, 'Veracidad´ (Batini y Scannapieco, 2016):

- *Accessibility* (Accesibilidad): refiere a qué tan fácil o rápidamente accesibles son los datos al momento de ser consultados.
- *Reliability* (Confiabilidad): refiere al grado de confianza que se tiene sobre los datos.
- *Integrity* (Integridad): refiere al grado de deformación que sufren los datos al ser transformados.

A su vez, cada dimensión puede refinarse en un conjunto de **factores** que representan aspectos particulares de la misma (Marotta y Serra, 2025). Por ejemplo, para la dimensión de Exactitud, se puede hacer referencia por un lado a la Exactitud Sintáctica de los datos, que contempla el grado en el cual los datos están escritos correctamente, mientras que la Exactitud Semántica es el factor que contempla qué tan acertados están los datos con respecto a la realidad a la que están asociados.

De los factores se definen **métricas** de calidad, que definen formas cuantitativas de medir aspectos concretos de un factor de calidad de datos (Serra, Peralta, Marotta, y Marcel, 2022). Una métrica debe definir su propósito, y además tiene definida una granularidad y el tipo del resultado. La granularidad se refiere al nivel de agregación de los datos sobre el cual sobre los cuales se calcula la métrica (por ejemplo, 'tabla', 'atributo', o 'tupla' en el caso de un modelo relacional). El tipo de resultado es el dominio del valor obtenido al ejecutar la medida de la métrica (por ejemplo, '*Boolean*', '*Integer*', o un rango de valores) (Serra, 2024). Por ejemplo, dado el factor de Exactitud Sintáctica en un conjunto de datos donde es importante el formato de la fecha, una métrica de calidad puede referir a si las fechas en los datos están en el formato esperado, con la granularidad 'celda', y retorna un valor del tipo 'Boolean'.

Cada métrica puede ser medida por uno o más **métodos de medición**, que son procesos que implementan a las métricas. Los métodos definen el algoritmo para medir una métrica. De la instanciación de los métodos sobre un conjunto de datos, se obtiene una **medida de calidad**.

En la Figura 2.9 se ilustra la relación de jerarquía entre los conceptos de calidad detallados anteriormente.

2.2.2. Modelo de Calidad de Datos

Para realizar una medición y evaluación global de la calidad de un conjunto de datos estudiados, se define un **modelo de calidad**, que es un conjunto de dimensiones, factores, métricas y métodos de calidad de datos.

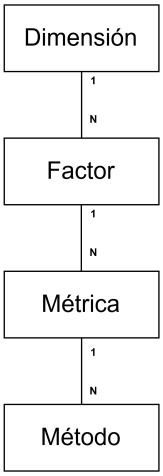

Figura 2.9: Jerarquía de conceptos de Calidad de Datos.

Para la definición de un modelo de calidad para un conjunto de datos, se seleccionan las dimensiones que sean relevantes para la realidad estudiada, y a partir de ellas se escogen factores, métricas y métodos que sean de interés. Para esto, típicamente se realiza una tarea previa de *data profiling*.

En este trabajo, se considerará el **metamodelo de calidad** (Figura 2.10), presentado en (Serra y cols., 2022). La instanciación del metamodelo es un modelo de calidad.

El metamodelo contiene la jerarquía de los conceptos de calidad de datos presentados en la sección anterior, junto con un conjunto de atributos que compone a cada uno de ellos. Además, en el metamodelo se incluye el concepto *Applied DQ Method*, que es la aplicación de un método para un conjunto de datos particular.

En la Figura 2.11 se presenta la especificación completa del metamodelo, tomada de (Serra, 2024).

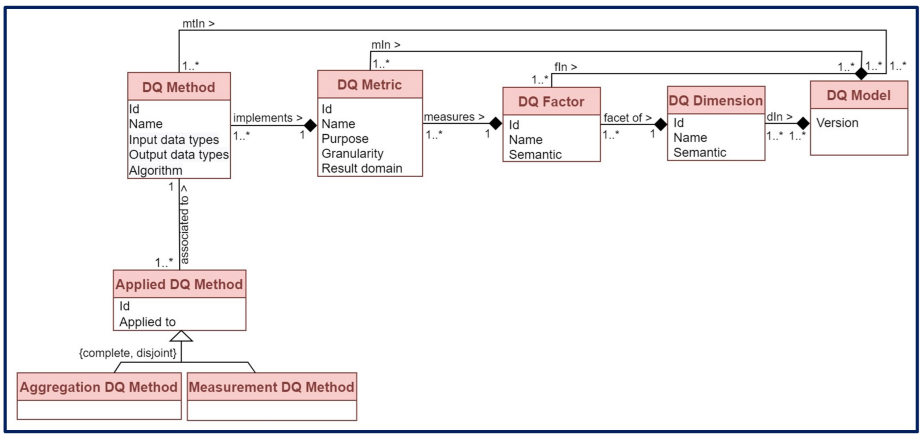

Figura 2.10: Metamodelo de calidad.
(Serra, 2024)

DQ Model	
Version	The version that identifies the DQ model.
DQ Dimension	
dIn	The DQ model versions to which it belongs.
Id	DQ dimension identifier.
Name	DQ dimension name.
Semantic	The type of DQ problem the DQ dimension addresses.
DQ Factor	
fIn	The DQ model versions to which it belongs.
Id	DQ factor identifier.
Name	DQ factor name.
Semantic	The DQ factor semantic, more specific than the one recorded for the corresponding DQ dimension.
Facet of	The DQ dimension to which it refers.
DQ Metric	
mIn	The DQ model versions to which it belongs.
Id	DQ Metric identifier.
Name	DQ Metric name.
Purpose	The measurement objective.
Granularity	The data granularity to which the measure is associated, and is strongly dependent on the data model (e.g. in the relational model, it could be table, attribute, tuple or value).
Result domain	The domain of the DQ value obtained in the measurement, for example, a Boolean, an Integer, an interval, etc.
Measures	The DQ factor that it measures.
DQ Method	
mtIn	The DQ model versions to which it belongs.
Id	DQ Method identifier.
Name	DQ Method name.
Input data type	Data type expected as input parameters.
Output data type	Data type expected as output parameters.
Algorithm	The process that implements the DQ metric.
Implements	The DQ metric that it implements.
Applied DQ Method	
Id	Applied DQ method identifier.
Type	Applied method type (measurement OR aggregation).
Associated to	The DQ method to which it is associated.
Applied to	Data schema attributes on which the method is applied.

Figura 2.11: Especificación del metamodelo de calidad de datos. (Serra, 2024)

Capítulo 3

Trabajos Previos

En este capítulo se presentan estudios previos realizados por el grupo Gema, que se utilizarán como base para el desarrollo de este proyecto.

3.1. Gestión de calidad de datos basada en el contexto

En el análisis de la calidad de los datos, cobra un peso significativo la realidad asociada a los datos estudiados. Esto se debe a que distintos dominios de negocio, distintos tipos de usuarios y distintos sistemas de almacenamiento y procesamiento pueden presentar diferentes necesidades y exigencias sobre los datos utilizados. Por este motivo, no solo es importante considerar la realidad que representan esos datos, sino también el **contexto de los datos**, a la hora de seleccionar las dimensiones, factores, métricas y métodos para definir un modelo de calidad.

En esta sección se presenta un resumen de la propuesta presentada en (Serra, 2024) para modelar el contexto de los datos, y qué relación tiene el mismo con el metamodelo de calidad de los datos mencionado en la Sección 2.2.2.

3.1.1. Metamodelo de Contexto

Con el objetivo de proveer una definición precisa y representable del contexto de los datos, en (Serra, 2024) se define el **metamodelo de contexto**, cuya instanciación es un **modelo de contexto**.

El modelo de contexto está definido por un conjunto de **componentes de contexto**. A continuación se detallan todos los componentes de contexto presentes en el metamodelo (Serra, 2024):

- *Application Domain*: el dominio de los datos evaluados.
- *Business Rules*: relaciones entre los valores de los atributos de los registros, que se derivan de la semántica del dominio de los datos y de las políticas

del negocio propietario de los datos. Por este motivo, usualmente están fuertemente vinculadas al *Application Domain*.

- *DQ requirements*: requisitos de calidad de datos que los datos estudiados deben satisfacer al momento de ser consultados.
- *Data filtering*: requisitos específicos de selección de datos para realizar una tarea dada.
- *System requirements*: requisitos funcionales y no funcionales asociados al alojamiento y consumo de los datos. Son impuestos por una o muchas *tasks at hand*.
- *Task at hand*: la tarea para la cual son utilizados los datos evaluados.
- *Users Types*: los tipos de usuarios que consumen los datos, agrupados según perfil o preferencias.
- *Metadata*: datos que describen los datos evaluados (por ejemplo, explicando la semántica de su contenido, describiendo el tamaño de los registros, etc.). Se diferencian en dos tipos de metadatos: *DQ Metadata*, que comprende a los valores obtenidos como resultado de la medición de la calidad de los datos, y *Other Metadata*.
- *Other data*: conjuntos de datos que dan contexto a los datos evaluados.

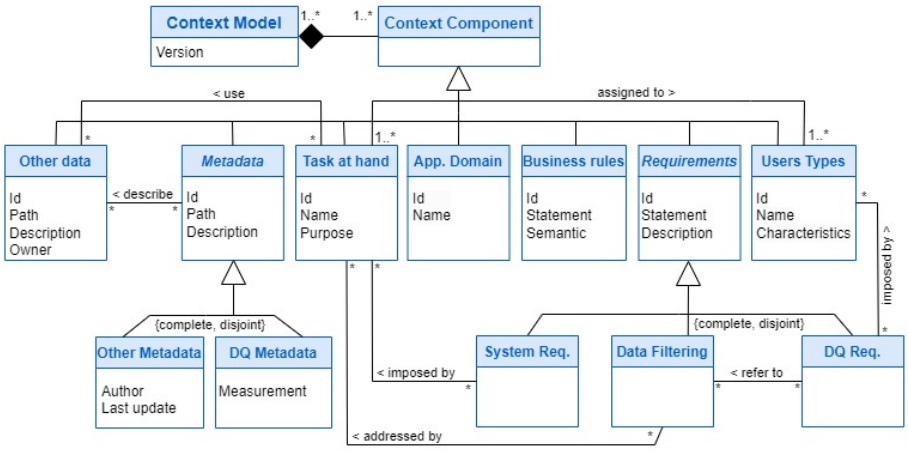

Figura 3.1: Metamodelo de contexto.
(Serra, 2024)

En la Figura 3.1, tomada de (Serra, 2024), se presenta el metamodelo de contexto, señalando todos los componentes descritos anteriormente con sus respectivos atributos, y las relaciones entre ellos. Se destaca que el metamodelo de contexto presenta un diseño extensible, que permite agregar, cuando sea necesario, nuevos componentes de contexto, atributos y relaciones.

3.1.2. Metamodelo de Calidad de Datos basada en contexto

En el marco de la calidad de los datos, el modelo de contexto es altamente influyente al momento de determinar métricas para ser evaluadas. El conocimiento del modelo de contexto permite la definición de métricas específicas y altamente pertinentes con respecto a la realidad de los datos. Por lo tanto, el contexto de los datos permite contextualizar no solo a los datos, sino también a modelos de calidad que se instancien sobre ellos.

Por este motivo, en (Serra, 2024) se presenta la relación que existe entre los modelos de contexto y los modelos de calidad de los datos. En la Figura 3.2, tomada de (Serra, 2024), se muestra el **metamodelo de calidad de datos basada en contexto** (*Context-aware DQ Metamodel*), que incluye al metamodelo de contexto con todos sus componentes, y la interacción de los mismos con los elementos del metamodelo de calidad de datos.

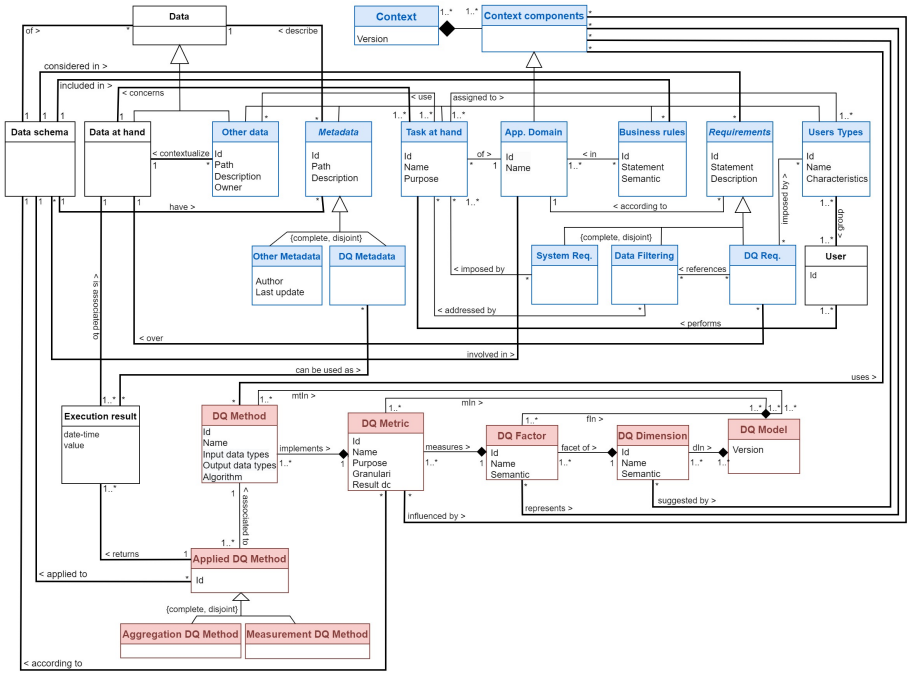

Figura 3.2: Metamodelo de Calidad de Datos Basada en el Contexto. (Serra, 2024)

Se destaca que en este metamodelo se incluyen clases auxiliares, que no pertenecen ni al metamodelo de calidad ni al metamodelo de contexto, pero que son necesarias para expresar la interacción entre ellos:

- *Data at hand*: los datos en los que se centran tanto el modelo de contexto como el modelo de calidad. Son los datos para los cuales estamos evaluando

la calidad.

- *Data*: *Data at hand* y *Other data*.
- *Data schema*: el esquema de los datos en *Data*. Sobre estos esquemas se definen las métricas de calidad.
- *User*: los usuarios que son agrupados por los *Users Types* en el modelo de contexto.
- *Execution result*: El resultado de ejecutar una medición de calidad. Incluye el valor obtenido y la fecha y hora de ejecución.

3.2. Arquitectura genérica de Big Data

Este proyecto se basa en la especificación de una arquitectura genérica de *Big Data* propuesta en el proyecto de grado (Cortés, 2024), que se muestra en la Figura 3.3. Esta arquitectura busca proveer una solución genérica y adaptable para aplicaciones de *Big Data*.

La arquitectura está principalmente inspirada en la *Zaloni Zones arquitecture* (descripta en la Sección 2.1.4), pero incorpora elementos de otras arquitecturas presentadas en la literatura con el objetivo de cumplir con una serie de requisitos relevados de la bibliografía. En particular, se busca incorporar características de gobierno de datos y gestión de calidad de datos dentro de las capacidades de la arquitectura.

La arquitectura soporta además el almacenamiento de *datasets* estructurados, semi-estructurados y no estructurados.

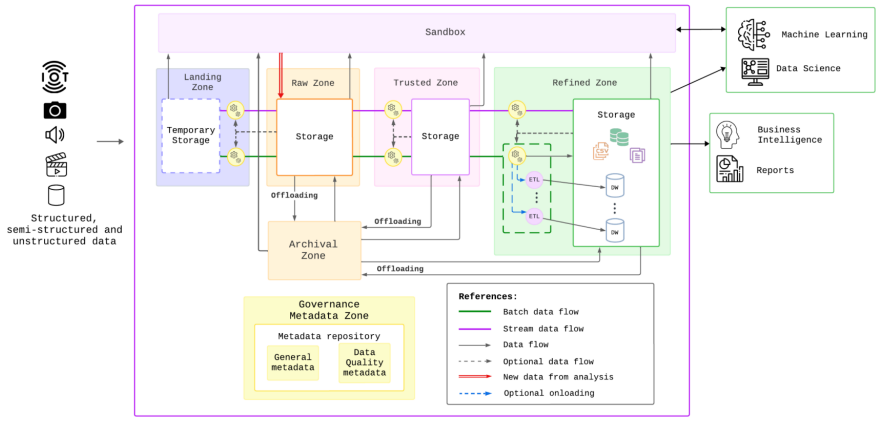

Figura 3.3: Arquitectura genérica de *Big Data*.
(Cortés, 2024)

3.2.1. Zonas de la arquitectura

A continuación se realiza una descripción de cada una de las zonas de la arquitectura:

- *Landing Zone*: zona encargada de la extracción de datos de las fuentes de entrada. Se realizan controles mínimos de calidad y confidencialidad. Los datos no se persisten en esta zona, sino que se propagan a la *Raw Zone*. Si existen metadatos disponibles en las fuentes, se extraen y se almacenan en la *Governance Metadata Zone*.

- *Raw Zone*: zona donde se almacenan los datos extraídos de las fuentes por la *Landing Zone*, conservando su formato original. En esta zona no se ejecutan procesos de transformación de los datos.

- *Trusted Zone*: zona en la cual se procesan y almacenan los datos de la *Raw Zone*, con el objetivo de cumplir con políticas de negocio, confidencialidad o calidad. Esta zona permite realizar integraciones de datos parciales, es decir que involucren únicamente ciertos subconjuntos de los datos. En este diseño, no se exige que los datos estén integrados bajo un mismo enfoque de modelado como se sugiere en, ya que el esquema de modelado elegido puede no ser compatible, o bien generar trabajo redundante, considerando los esquemas de modelado que se impondrán en la *Refined Zone* para adaptar los datos a los casos de uso finales.

- *Refined Zone*: zona encargado del modelado de los datos en una estructura diseñada para su consumo por usuarios o herramientas finales. En este diseño, no se exige un formato único común para todos los datos alojados en esta zona, sino que se permite el modelado en múltiples formatos para cada caso de uso que se desea soportar. Además, en esta zona se soporta la implementación de DWs para brindar soporte a herramientas de BI.

- *Archival Zone*: zona de almacenamiento de datos fríos, inspirada en el *archival data pond* propuesto en la arquitectura de *Data Ponds* de Bill Inmon. Las zonas *Raw*, *Trusted* y *Refined* pueden enviar datos poco utilizados a la *Archival Zone*, y se registran metadatos que indican de qué zona provino. Para volver a acceder a estos datos, se vuelven a enviar de la *Archival Zone* a su zona de origen. La *Sandbox* puede acceder en todo momento a los datos de esta zona, para lo cual solicita una copia de los mismos.

- *Sandbox*: zona que permite el relevamiento de datos de cualquiera de las otras zonas para realizar tareas de exploración y análisis de datos ad-hoc por Data Scientists. Los resultados de estos análisis pueden ser almacenados como datos adicionales en la Raw Zone.

- *Governance Metadata Zone*: zona que almacena el repositorio de metadatos. Estos metadatos se distinguen en dos tipos: *General Metadata*, que refiere los metadatos asociados a los datos almacenados y a los procesos

realizados en la arquitectura, y *DQ Metadata*, que refiere a los metadatos de calidad, como lo son la representación de modelos de calidad y los resultados de mediciones. Se profundizará sobre las características de esta zona y el modelado de los metadatos en la Sección 3.2.3.

3.2.2. Grupos de usuarios

La arquitectura define cinco grupos de usuarios o roles, que poseen distintos privilegios y restricciones sobre el acceso a los datos del DL:

- *Data Analysts*: usuarios cuyo rol es analizar patrones de datos, utilizando herramientas de cálculo y visualización sobre datos con un alto nivel de refinamiento. Realizan informes, visualizaciones y análisis descriptivos, y examinan datos para identificar patrones y tendencias. Tienen acceso únicamente la *Refined Zone*.
- *Data Scientists*: usuarios que llevan a cabo tareas de *Data Science*, como la creación de modelos predictivos utilizando como entrada los datos disponibles del DL. Son los usuarios de la zona *Sandbox*, la cual les provee un ambiente para realizar experimentos y desarrollar modelos. Además, tienen acceso a la *Governance Metadata Zone* y a la *Refined Zone*.
- *System Administrators*: usuarios administradores, responsables del mantenimiento del sistema. Tienen acceso a todas las zonas.
- *Data Quality Experts*: usuarios encargados de la gestión de la calidad de datos del sistema. Son responsables por la definición del modelo de calidad, así como la medición y evaluación de la calidad de los datos en las distintas etapas de su procesamiento.
- *Data Engineers*: usuarios responsables de la implementación de los procesos y transformaciones de las zonas *Landing*, *Trusted* y Refined. Tienen acceso a estas zonas, y adicionalmente también tienen acceso a la *Governance Metadata Zone*.

En la tabla 3.1 se presenta un resumen de los grupos de usuarios con acceso a cada zona.

3.2.3. Modelado de metadatos

Como se mencionó anteriormente, la arquitectura busca incorporar aspectos de Gobierno de datos y de Gestión de la Calidad de los Datos. Para esto, se hace uso de la *Governance Metadata Zone*, la cual almacena y gestiona metadatos referentes a los *datasets*, procesos, y entidades relacionadas a la calidad de los datos.

En la Figura 3.4 se muestra el modelado de la *General Metadata*, la cual representa las zonas de la arquitectura, los *datasets* almacenados en ellas, y los procesos que se ejectuan en la arquitectura.

Zona	Grupos de usuarios con acceso
Landing Zone	System administrator, Data Engineer, Data Quality Expert
Raw Zone	System administrator, Data Quality Expert
Archival Zone	System administrator
Trusted Zone	System administrator, Data Engineer, Data Quality Expert
Refined Zone	Todos los usuarios
Sandbox	Data Scientists, System administrator
Governance Metadata Zone	Data Scientists, System administrator, Data Engineer y Data Quality Expert

Tabla 3.1: Permisos de acceso de grupos de usuario por zona

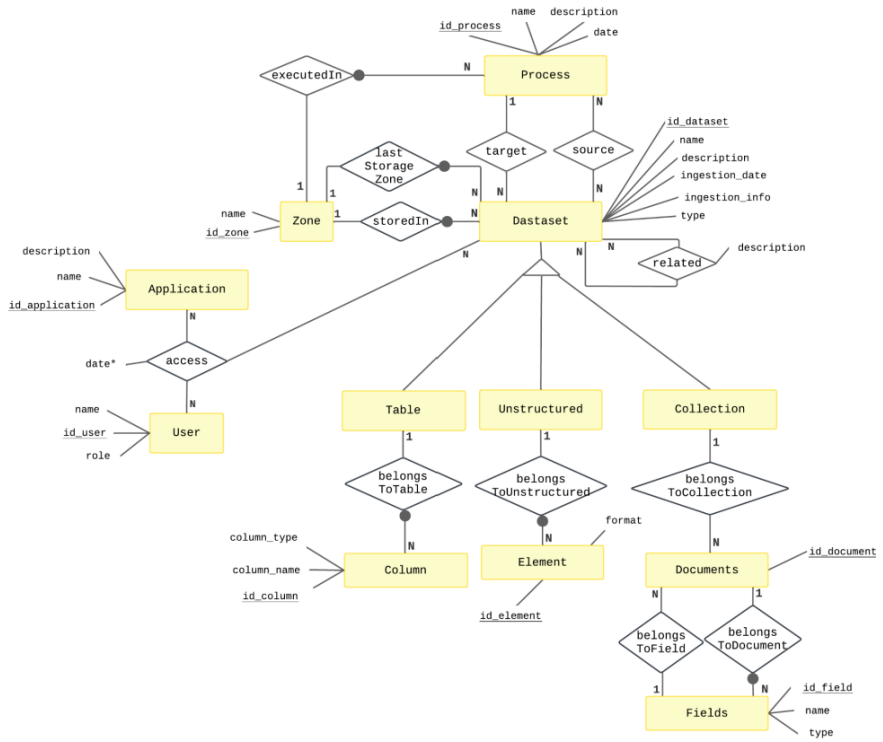

Figura 3.4: Modelo de metadatos de datos y procesos.
(Cortés, 2024)

Los metadatos de los *datasets* se categorizan en tres tipos según la estructura del *dataset*: *Table*, que representa a los datos estructurados, organizados en

columnas (*Column*); *Collection*, que representa a los datos semi-estructurados del tipo de bases de datos documentales, que son colecciones de documentos (*Document*), que a su vez tienen un conjunto de campos (*Field*); y *Unstructured*, que consiste de un conjunto de elementos (*Element*) de cualquier formato. Se modela también la relación de pertenencia de un *dataset* a una zona (*Zone*).

Los metadatos de los procesos se representan mediante la entidad *Process*, y almacena los datos de la ejecución de un proceso, como lo son el nombre del proceso, su descripción, la fecha de ejecución. Se representa también el vínculo del proceso con los *datasets* que utiliza tanto como entrada como salida, y adicionalmente la zona en la cual se ejecutó el proceso.

Por otro lado, en la Figura 3.5 se presenta el modelado de la *DQ Metadata*, la cual representa los metadatos relacionados a las medidas de calidad realizadas sobre los *datasets* de la arquitectura. Se modelan algunos de los conceptos de calidad presentados en la Sección 2.2.1: *Dimension*, *Factor*, *Metric* y *Measure*. Notar que el concepto *Measure* se categoriza según la granularidad de la métrica a la cual pertenece.

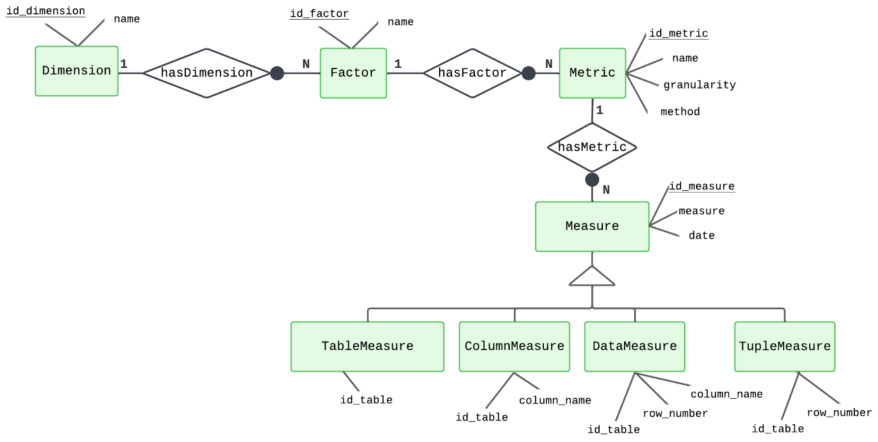

Figura 3.5: Modelo de metadatos de calidad.
(Cortés, 2024)

La Figura 3.6 muestra el modelado completo de los metadatos en la arquitectura, que combina los modelados de la *General Metadata* y *DQ Metadata*, agregando las relaciones entre las medidas de calidad y los componentes cuya calidad miden. La categorización del concepto *Measure* se reemplaza por relaciones entre *Measure* y las distintas categorizaciones del concepto *Dataset*, y los conceptos relacionados con ellas.

En la implementación de la arquitectura, se opta por utilizar una base de datos de grafos para el modelado de los metadatos, al presentar una variedad de ventajas sobre bases de datos relacionales, como la flexibilidad de los esquemas y la posibilidad de realizar consultas complejas, como *pattern matching* y

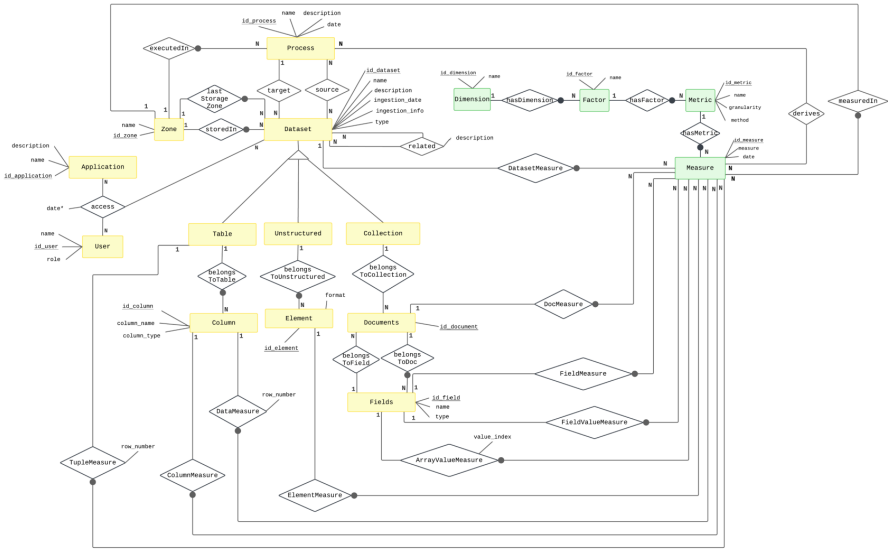

Figura 3.6: Modelo de metadatos de la arquitectura.
(Cortés, 2024)

consultas de longitud de caminos, que son útiles para consultas sobre la traza de transformaciones de los datos, por ejemplo. En las Figuras 3.7, 3.8 y 3.9 se presenta el modelado de grafo para los metadatos de los *datasets* estructurados, semi-estructurados y no-estructurados respectivamente. Se muestran los esquemas de esos grafos, donde los cada óvalo representa un tipo de nodo, mientras que las flechas representan las relaciones entre ellos.

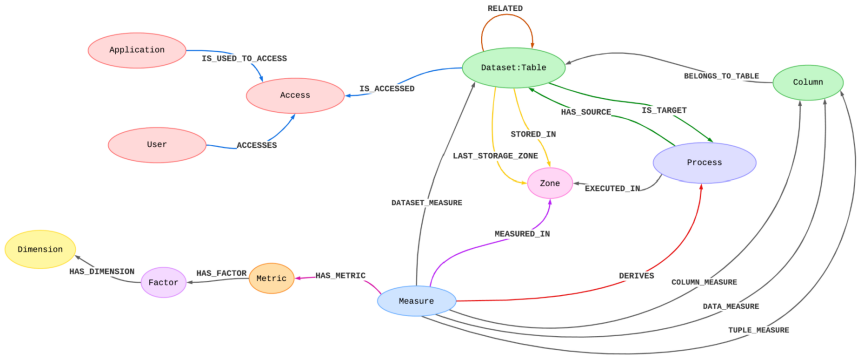

Figura 3.7: Modelo de metadatos para *dataset* estructurado.
(Cortés, 2024)

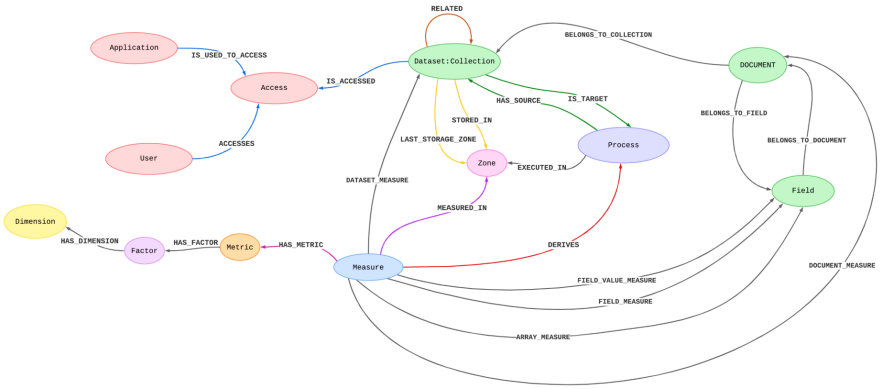

Figura 3.8: Modelo de metadatos para *dataset* semi-estructurado. (Cortés, 2024)

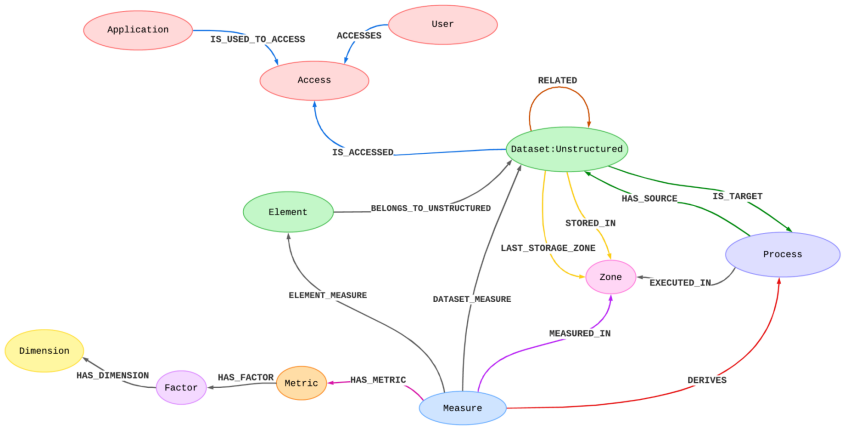

Figura 3.9: Modelo de metadatos para *dataset* no-estructurado. (Cortés, 2024)

3.2.4. Gestión de calidad de los datos

En la Figura 3.10 se presenta un diagrama del diseño de la gestión de la calidad de los datos de la arquitectura. En este diagrama se distinguen tres tipos de procesos: los coloreados en amarillo son los procesos de transformación de los datos, los rosados son las ejecuciones de las mediciones de calidad, y los azules son los procesos de generación de metadatos a partir de estas mediciones.

Esta arquitectura implementa la medición de la calidad de los datos en dos etapas en cada zona: se realiza una primera medición antes de ejecutar los procesos de transformación y otra después. Esto permite analizar cómo afectan

los procesos implementados a la calidad de los datos, y consecuentemente diseñar mejoras para estos procesos. Se destaca que en la *Raw Zone* no se realizan transformaciones ni se ejecutan mediciones de calidad.

Como se mencionó en la Sección 3.2.1, las medidas obtenidas generan metadatos de calidad, los cuales son almacenados en la Governance Metadata Zone como parte de la *DQ Metadata*. Estas medidas son evaluadas por los *Data Quality Experts*, y consecuentemente se derivan mejoras a los procesos en base a la evaluación.

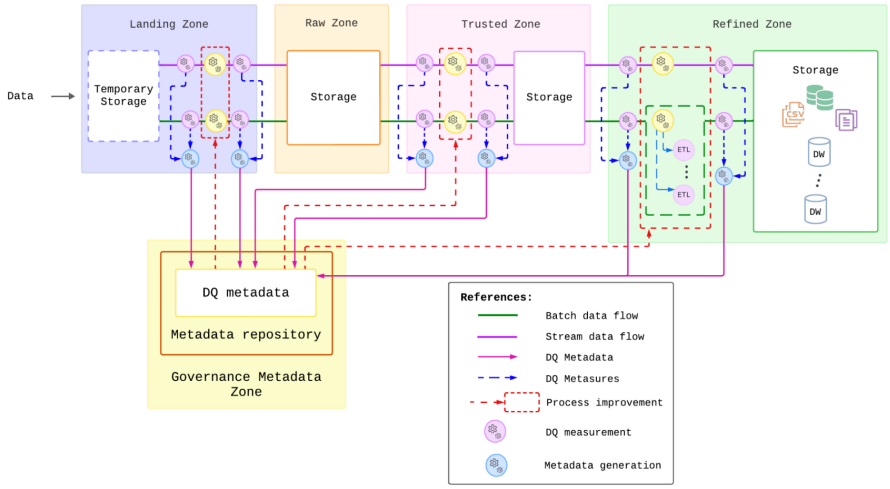

Figura 3.10: Proceso de gestión de calidad en la arquitectura. (Cortés, 2024)

Capítulo 4

Propuesta de calidad de datos basada en contexto en *Data Lake*

En este capítulo se presenta una propuesta para incorporar a la arquitectura de *Big Data*, presentada en la Sección 3.2 (en adelante, arquitectura de DL), la capacidad de instanciación de modelos de contexto y modelos de calidad de datos que respondan a ellos. Para esto es necesario adaptar y unificar ambos trabajos discutidos en el Capítulo 3 ((Cortés, 2024) y (Serra, 2024)).

En la arquitectura de DL se cuenta con la posibilidad de definir dimensiones, factores y métricas de calidad, que se almacenan como parte de los metadatos de calidad en la *Governance Metadata Zone*. En este trabajo, se busca incorporar el uso de modelos de contexto como parte de la gestión de calidad de los datos de la arquitectura. Por este motivo, se propone una extensión al metamodelo de contexto que responde a las nuevas consideraciones que surgen en el DL. A su vez, se propone una serie de modificaciones al modelo de metadatos de la arquitectura, con el objetivo de poder almacenar en la misma los modelos de contexto, y representar sus relaciones con los otros metadatos en el DL.

4.1. Influencia del contexto en las zonas del *Data Lake*

Con el objetivo de definir un lineamiento general para el uso e instanciación de modelos de contexto en el DL, en primer lugar se realiza un análisis de la presencia e influencia de los diferentes componentes de contexto sobre los *datasets* alojados en las diferentes zonas del DL. Para esto, se destacan dos consideraciones generales:

- Un modelo de contexto refiere a un único *dataset*, pero pueden existir

múltiples modelos de contexto para un mismo *dataset*.

- Se considerará como "usuarios de los datos", que se corresponde con el concepto de *User* del metamodelo de calidad de datos basada en el contexto (ver la Sección 3.1.2), únicamente a los usuarios del DL del tipo *Data Analyst*, que son aquellos que acceden y utilizan los *datasets* disponibles en la *Refined Zone*. Esto se debe a que, si bien los otros grupos de usuarios mencionados en la Sección 3.2.2 también tienen acceso a los datos, son los *Data Analyst* los que realmente utilizan el contenido de los datos para realizar sus tareas de negocio.

4.1.1. Presencia de componentes de contexto por zona

Para este análisis, solo se contemplarán las zonas *Landing*, *Raw*, *Trusted* y *Refined*, ya que son las zonas que participan en el flujo de los datos que se transforman y se adaptan para el caso de uso final de los usuarios de los datos. No se consideran las zonas *Sandbox* ni *Archival*, ya que toman los datos directamente de las distintas zonas sin transformarlos, por lo que la calidad de los mismos será igual a la calidad en sus zonas de origen (ver Figura 3.3).

A continuación se presenta un análisis de los componentes de contexto que participarán en los modelos de contexto definidos para cada una de las zonas consideradas.

Landing Zone

Como se menciona en la Sección 3.2.1, en la *Landing Zone* se realiza la ingestión inicial de los datos desde las fuentes originales al DL. En caso de estar disponibles, también se realiza una extracción de los metadatos de las fuentes, que puede ser utilizada como el componente de contexto *Metadata*.

En esta zona se realizan únicamente chequeos de calidad básicos y de baja complejidad para los datos. Estos chequeos no responden a necesidades particulares de los casos de uso de los usuarios finales de los datos, sino a lineamientos generales de negocio, por lo que responden a necesides impuestas por el *Application Domain* y sus *Business Rules* asociadas. Adicionalmente, los chequeos pueden surgir de limitaciones del sistema vinculadas a la implementación del DL, por lo que se presentarán *System Requirements*.

Como no se cuenta con la presencia de usuarios finales que accedan a estos datos, no se cuenta con *User Types* ni *Task at Hand* para contextualizar a los *datasets* que ingresan a esta zona. Consecuentemente, tampoco se podrán instanciar requerimientos del tipo *DQ Requirement* ni *Data Filtering*, ya que se corresponden con la presencia de *User Types* y *Tasks at Hand* respectivamente.

Por lo tanto, los modelos de contexto definidos para contextualizar los *datasets* ingeridos en la *Landing Zone*, contarán únicamente con los componentes: *Application Domain*, *Business Rules*, *Metadata* y *System Requirements*.

Raw Zone

Como se mencionó en la Sección 3.2.4, en la *Raw Zone* no se realizan mediciones de calidad de los datos. Por lo tanto, no se definirán modelos de calidad de datos para los *datasets* almacenados en esta zona, y consecuentemente no se definirán modelos de contexto.

Trusted Zone

En la *Trusted Zone*, se comienzan a realizar transformaciones sobre los datos que buscan cumplir con políticas más refinadas de negocio, confidencialidad o calidad. Además, en esta zona se suele resolver la integración de datos provenientes de distintas fuentes. Nuevamente, los controles y mejoras de calidad realizadas sobre los datos de esta zona se corresponderán con exigencias provenientes del *Application Domain* y sus *Business Rules*. Sin embargo, en esta zona se comienzan a realizar controles de calidad más sofisticados, que introducen el uso de referenciales, que comúnmente se utilizan para evaluar la calidad según las dimensiones *Accuracy* y *Completeness*. Estos referenciales se consideran parte del componente de contexto *Other Data*.

Al igual que en la *Landing Zone*, no se cuenta con la presencia de un usuario final que tenga acceso a los datos. Por lo tanto, no se cuenta con el componente de contexto *User Type*, y tampoco se cuenta con una *Task at Hand*, *DQ Requirements*, ni *Data Filtering*.

En resumen, los componentes de contexto que participan de modelos de contexto definidos sobre *datasets* de la *Trusted Zone* serán del tipo *Application Domain*, *Business Rules*, *System Requirements*, *Metadata* y *Other Data*.

Refined Zone

La *Refined Zone* es la zona que más se asemeja al caso de uso original para el cual fue pensado el metamodelo de contexto. En esta zona se almacenan *datasets* con un alto grado de procesamiento, con esquemas adaptados específicamente para un caso de uso particular, cuyo propósito es ser consumidos por usuarios finales del DL. Por lo tanto, en un modelo de contexto para un *dataset* de esta zona puede presentarse cualquiera de los componentes de contexto presentados en (Serra, 2024).

Los *User Types* serán los perfiles de los *Data Analysts*, y sus *Tasks at Hand* serán las tareas que desempeñan.

Al ser la única zona del DL a la cual tienen acceso los usuarios finales de los datos, esta es la única zona cuyos modelos de contexto presentarán los componentes de contexto *User Type* y *Task at Hand*, y consecuentemente también será la única zona que presente los componentes que dependen de la existencia de un usuario y una tarea llevada a cabo por el mismo, que es el caso de los componentes *Data Filtering* y *DQ Requirements*.

En la tabla 4.1 se muestra un resumen de la presencia de los componentes de contexto en los modelos de contexto para cada zona del DL.

Como parte del resultado del análisis, se observa que los componentes de contexto que participan en los modelos de contexto de una zona son un subconjunto de los que participan en las zonas siguientes (a excepción de la *Raw Zone*, ya que no se definen modelos de contexto sobre sus *datasets*). Este resultado es razonable, ya que el proceso de refinamiento de los datos en el DL implica una creciente incorporación de significado, estructura y validación. A medida que los datos avanzan desde la *Landing Zone* hacia la *Refined Zone*, se incorporan más restricciones sobre la calidad de los mismos, lo que se corresponde con una mayor influencia del dominio y con la definición de reglas de negocio más específicas. Por tanto, es esperable que los modelos de contexto se expandan progresivamente conforme los datos alcanzan niveles más altos de procesamiento y cercanía al caso de uso final.

	Landing Zone	**Raw Zone**	**Trusted Zone**	**Refined Zone**
Application Domain	✓	x	✓	✓
Business Rules	✓	x	✓	✓
Metadata	✓	x	✓	✓
Other Data	x	x	✓	✓
System Requirements	✓	x	✓	✓
Data Filtering	x	x	x	✓
DQ Requirements	x	x	x	✓
User Types	x	x	x	✓
Task at Hand	x	x	x	✓

Tabla 4.1: Presencia de los componentes de contexto por zona del *Data Lake*

4.1.2. Propagación hacia atrás de los componentes de contexto

Existen escenarios en los DL en donde un componente de contexto de una zona puede causar la existencia de un componente de contexto de una zona anterior en el flujo del DL. Esto se debe a que existen problemas de calidad que se propagan "hacia adelante" y se detectan al final del flujo.

Para modelar esta situación, se necesitaría instanciar un *DQ Requirement* que participe en un modelo de contexto para un *dataset* de la *Trusted Zone*.

Sin embargo, esto es incompatible con lo relevado del análisis realizado en la Sección 4.1.1, ya que no existen usuarios finales que accedan directamente a los datos alojados en esta zona, y por lo tanto no puede haber *DQ Requirements* impuestos por ellos.

Para permitir el modelado de estas situaciones, se considera que los *DQ Requirements* pertenecientes a modelos de contexto de *datasets* de la *Refined Zone* pueden derivar tanto en *System Requirements* como en *Business Rules* en modelos de contexto para *datasets* de la zona *Trusted*. A su vez, los *System Requirements* que participan en modelos de contexto para *datasets* de la *Trusted Zone* pueden derivar en otros *System Requirements* en modelos de contexto para *datasets* de la *Landing Zone*. Como criterio general, en los casos donde las características del sistema, como la capacidad de procesamiento, el uso de memoria y el rendimiento, sean influyentes sobre el requisito que se propaga, se definirán *System Requirements*. En otro caso, se propagarán como *Business Rules*. En la Figura 4.1 se presenta un diagrama de la propagación hacia atrás de los componentes de contexto discutidos.

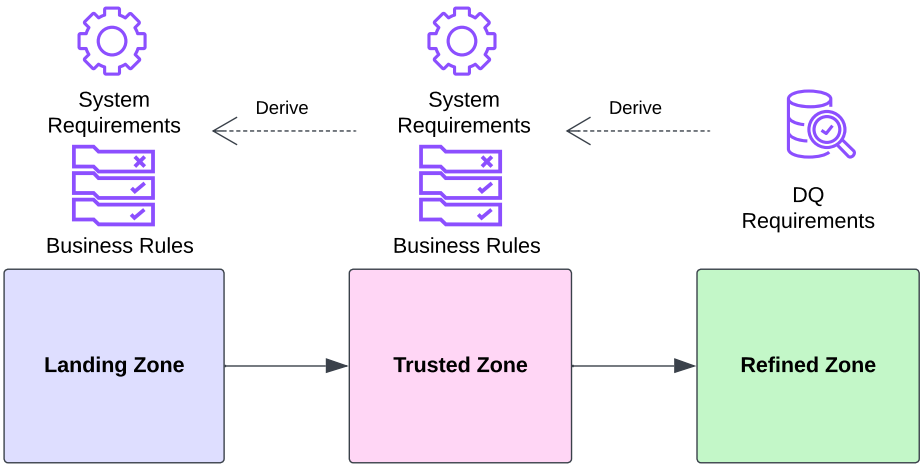

Figura 4.1: Propagación hacia atrás de los componentes de contexto.

Dos dimensiones que presentan este tipo de situaciones son la dimensión *Freshness* y la dimensión *Completeness* (en particular, el factor *Coverage*).

Ejemplo (*Freshness*): supongamos que existe un *DQ Requirement DQR1* impuesto por un *User Type UT1* que agrupa *Users* que acceden a los datos en la *Refined Zone*, cuyo *statement* es "Los datos deben actualizarse diariamente y estar disponibles a las 9am". Este requisito necesariamente impondrá requisitos sobre los *datasets* anteriores en el flujo, ya que deberán ser procesados en una hora anterior a las 9am, teniendo en cuenta la latencia de los procesos involucrados. Si se supone que los procesos entre la *Trusted Zone* y la *Refined Zone* tienen una duración de 30 minutos, entonces es de interés evaluar si los datos están disponibles en la *Trusted Zone* a las 8:30am. A partir de *DQR1* se relevaría un nuevo *System Requirement SR1* cuyo *statement* es "Los datos deben

actualizarse diariamente y estar disponibles a las 8:30am", que se utilizará en modelos de contexto para *datasets* de la *Trusted Zone*. Suponiendo una latencia de 30 minutos adicionales entre las zonas *Landing* y *Trusted*, se derivaría consecuentemente de *SR1* otro *System Requirement SR2* cuyo statement será "Los datos deben ingerirse diariamente y estar disponibles a las 8am". En la Figura 4.2 se ilustra el ejemplo con un diagrama.

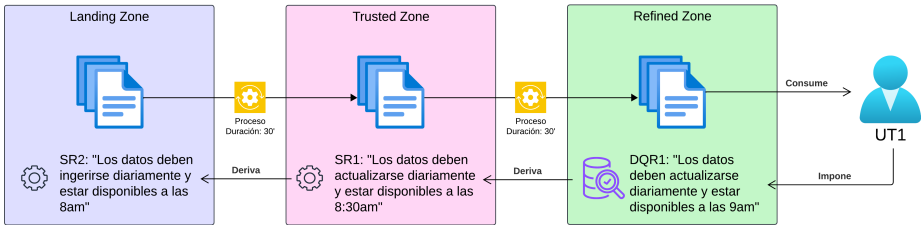

Figura 4.2: Ejemplo de propagación hacia atrás de un *DQ Requirement* temporal.

4.2. Extensión del metamodelo de contexto de (Serra, 2024)

En esta sección se presentan extensiones realizadas al metamodelo de contexto para incluir nuevas relaciones y componentes de contexto, que son de relevancia y utilidad en los DL.

4.2.1. Data Lineage

En la arquitectura, los datos pueden sufrir una gran variedad de transformaciones desde que son ingeridos inicialmente en la *Landing Zone* hasta que son disponibilizados para los usuarios finales en la *Refined Zone*. Los procesos que ejecutan dichas transformaciones tienen el potencial tanto de mejorar la calidad de los datos, como de introducir problemas que resulten en un deterioro de la calidad de los mismos. Por este motivo, el linaje de los datos toma un rol importante para determinar aspectos de calidad a evaluar sobre los datos, así como para detectar causas de problemas de calidad.

Ejemplo: supongamos que se tiene un *dataset* en la *Trusted Zone* que contiene información de transacciones financieras, y se conoce a través del linaje que dicho *dataset* fue construido a partir de un *dataset* que cuando fue ingerido en la *Landing Zone* presentaba un 30 % de valores faltantes en una determinada columna. Además, el linaje muestra que posteriormente fue sometido a un proceso de limpieza automática donde los valores faltantes son estimados mediante una lógica basada en promedios históricos. Se puede inferir que existe un riesgo potencial en la exactitud de algunos campos derivados. En este caso, el

conocimiento del linaje de los datos conduce a la definición de una métrica en la *Trusted Zone*, que de otro modo podría no haberse contemplado.

Se decide incorporar al metamodelo de contexto un nuevo componente de contexto: **Data Lineage**. El *Data Lineage* se incorpora al metamodelo de contexto como una nueva categorización de *Other Metadata*. Se destaca que esta categorización no es total, ya que se sigue aceptando la existencia de otros metadatos del tipo *Other Metadata* que no se relacionen con el *Data Lineage*. En la Figura 4.3 se ilustra la incorporación del componente *Data Lineage* (resaltado en color azul oscuro) al metamodelo de contexto.

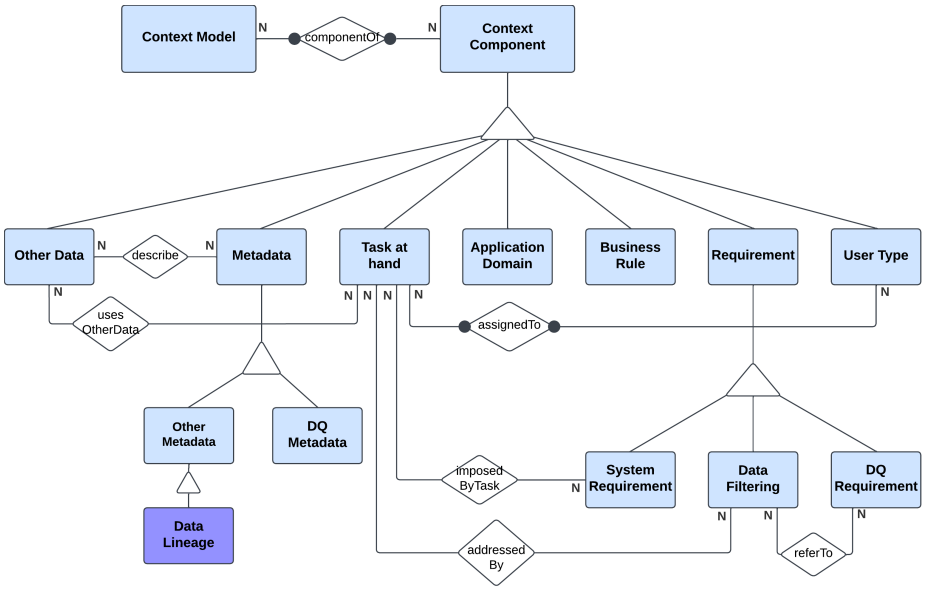

Figura 4.3: *Data Lineage* en el metamodelo de contexto

En la tabla 4.2 se presenta la especificación del componente *Data Lineage*. Para los *datasets* en la *Landing Zone*, el campo *sources* refiere a la fuente de la cual fueron ingeridos los datos, mientras que para los *datasets* en el resto de las zonas representa la unión de los *sources* de todos los *datasets* incluidos en el linaje. Este campo permite mantener un nivel mayor de trazabilidad y conocer el origen de los datos. El campo *Lineage* consiste de un String en formato *markup* (por ejemplo, *XML* o *JSON*) que contiene la información de todas las fuentes, *datasets* y procesos que participaron en la creación del *dataset* cuyo linaje se está describiendo. En esta propuesta se implementa el campo *Lineage* como un String *JSON*. Por último, las relaciones *includes dataset* y *includes process* referencian a los *datasets* y procesos del DL que participan en el linaje. Se mantienen estas relaciones para permitir una exploración más libre de los metadatos asociados a estos *datasets* y procesos, en caso de ser necesario.

El linaje de un *dataset* puede ser calculado a partir de los metadatos de los *datasets* y procesos, mediante una consulta al grafo de metadatos del DL.

Data Lineage	
Sources	Las fuentes originales de los datos, antes de ser insertados en el DL.
Lineage	El linaje de los datos en formato *JSON*.
Includes dataset	Referencia a los *datasets* en el DL incluidos en el linaje.
Includes process	Referencia a los procesos en el DL incluidos en el linaje.

Tabla 4.2: Especificación del componente *Data Lineage*

A partir de esta consulta, se puede generar una instancia del componente de contexto *Data Lineage*. Alternativamente, el *Data Lineage* puede construirse a partir del conocimiento de los procesos del *Data Lake*.

4.2.2. Nuevas relaciones entre componentes de contexto

Como se mencionó en la Sección 4.1.2, para modelar la propagación "hacia atrás"de los componentes de contexto, se extiende el metamodelo de contexto con la posibilidad de que *DQ Requirements* deriven *System Requirement* y *Business Rules*, y que a su vez los *System Requirements* puedan derivar otros *System Requirements*. Para modelar esto, se agregan tres relaciones al metamodelo de contexto:

- $BRDerivedFrom \subset BusinessRule \times DQRequirement$
- $derivedFromDQReq \subset SystemRequirement \times DQRequirement$
- $derivedFromSysReq \subset SystemRequirement \times SystemRequirement$

Todas las relaciones tienen cardinalidad *N:N*.
En la Figura 4.4 se presentan un diagrama de las relaciones agregadas.

4.3. Modificación del modelo de metadatos de la arquitectura de *Data Lake*

En esta sección se presentan las modificaciones realizadas al modelo de metadatos de la arquitectura de DL, para poder incluir el contexto en la gestión de la calidad de datos.

Dado el extenso tamaño del modelo de metadatos, se presenta divido por partes.

4.3.1. Modelo de metadatos de datasets y procesos

Al modelo de metadatos de *dataset* y procesos del DL (presentado en la Sección 3.2.3), se le realiza una modificación en la categorización de la entidad

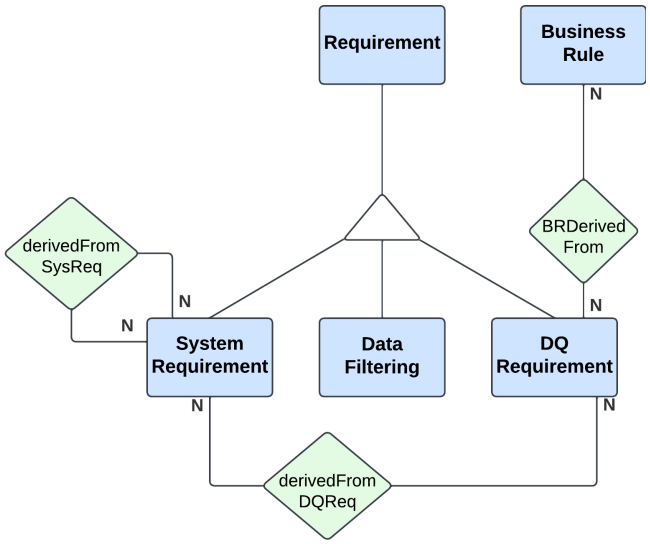

Figura 4.4: Relaciones agregadas al metamodelo de contexto para modelar la propagación

Dataset, donde se reemplaza la entidad *Table* por una nueva entidad *Database*. La entidad *Table* ya no es una categorización de *Dataset*, sino que se relaciona con la entidad *Database* mediante una relación *belongsToDB*, y posee los atributos *id_table*, *name*, y *description*. Este cambio se debe a que será de interés definir modelos de contexto y métricas de calidad que impliquen el uso de varias tablas a la vez, por lo que es de interés considerar a las bases de datos implementadas en la *Refined Zone* como un *dataset*. En el caso de contar con datos estructurados que no se relacionen con otros datos (por ejemplo, archivos individuales en formato tabular), se considerarán como elementos del tipo *Database* con una única *Table*.

En la Figura 4.5 se presenta el modelo de metadatos para datos y procesos de la arquitectura, con las modificaciones propuestas resaltadas en amarillo oscuro y verde.

4.3.2. Modelo de metadatos de calidad de datos

Como se presentó en la Sección 3.2.3, la arquitectura del DL modela los conceptos de calidad *Dimension*, *Factor*, *Metric* y *Measure*, contando con un modelado de metadatos de calidad que representa estas entidades. Sin embargo, no se cuenta con toda la jerarquía de entidades necesarias para instanciar un modelo de calidad de datos completo, según el metamodelo de calidad de datos presentado en la Sección 2.2.2 (especificado en (Serra, 2024)). Notablemente, no se cuenta con una entidad que permita agrupar las dimensiones en un modelo de calidad de datos, y no existe una distinción entre la aplicación de un método

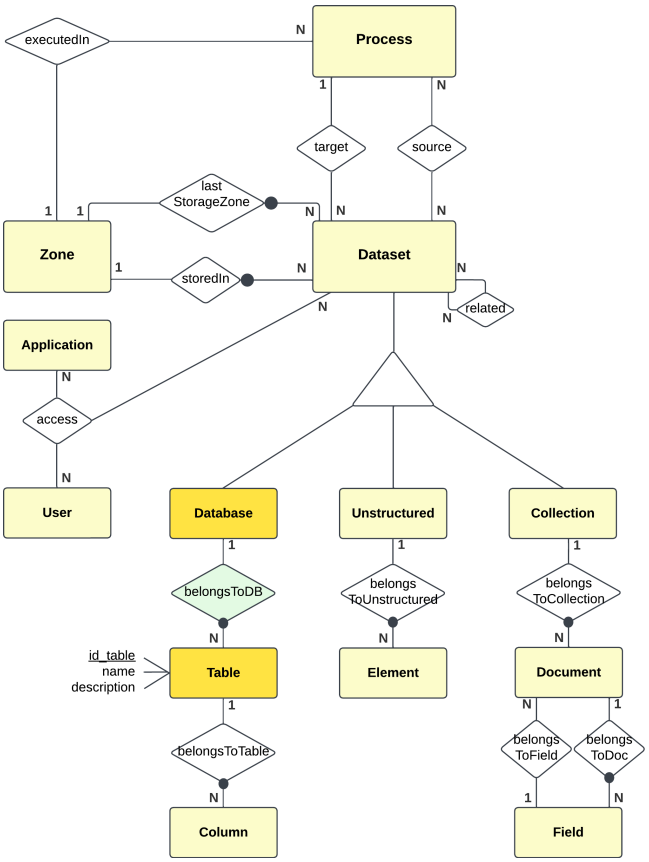

Figura 4.5: Modificación al modelo de metadatos de datos y procesos.

y el resultado de la ejecución del mismo, rol que ocupan los conceptos *Applied DQ Method* y *Execution Result* en el metamodelo de calidad de datos basada en el contexto (Serra, 2024).

En esta propuesta, se extiende el modelo de metadatos de calidad de la arquitectura de DL, con el objetivo de expandir la expresividad del mismo. Se toma como referencia el metamodelo de calidad presentado en (Serra, 2024).

En la Figura 4.6 se presenta el modelo de metadatos de calidad propuesto para la arquitectura de DL.

A continuación se describen las modificaciones realizadas al modelo de metadatos de calidad original de la arquitectura de DL. Del metamodelo de calidad de datos de (Serra, 2024) se relevan directamente las entidades *DQ Dimension*, *DQ Factor*, *DQ Metric*, *DQ Method* y *Applied DQ Method* con todos sus atributos, junto con las relaciones *dIn*, *facetOf*, *measures*, *implements* y *associated*.

Se utiliza la entidad *DQ Model* para agrupar *DQ Dimensions*, con el propósito de definir en los metadatos una instancia de un modelo de calidad. Se reem-

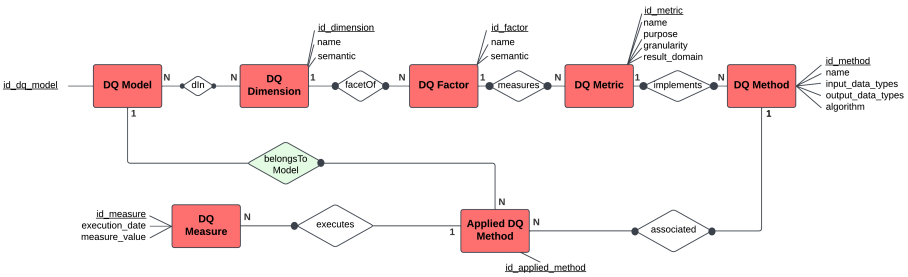

Figura 4.6: Modelo de metadatos de calidad de datos propuesto.

plaza el atributo *version* por un atributo identificador *id_dq_model*. Además, se incluye la entidad *DQ Measure*, que almacena la fecha y resultado de una ejecución de una medición.

Adicionalmente, se incluye una relación *1:N* entre *DQ Model* y *Applied DQ Method* llamada *belongsToModel*. Esto surge a partir de un requisito presente en el proyecto de grado (Castro y cols., 2025) (mencionado en la Sección 1.2), con el cual debíamos ser compatibles, y que presentaba la necesidad de tener una mejor navegabilidad entre las medidas de calidad y los modelos de calidad definidos.

4.3.3. Modelo de metadatos de contexto

Con el objetivo de incorporar el contexto como parte de la gestión de calidad de datos del DL, se propone un modelo de metadatos de contexto, que tiene como objetivo el almacenamiento completo de modelos de contexto y todos sus componentes en la *Governance Metadata Zone* del DL. Para esto, se representan todas las entidades del metamodelo de contexto, tomando en cuenta las extensiones propuestas en la Sección 4.2.

En la Figura 4.7 se presenta el modelo de metadatos de contexto que serán almacenados en la *Governance Metadata Zone*. Se resalta en color azul oscuro la inclusión del nuevo componente de contexto *Data Lineage* presentado en la Sección 4.2.1, y en verde las relaciones nuevas propuestas en la Sección 4.2.2. El componente *Data Lineage* presenta los atributos planteados anteriormente, mientras que el resto de los componentes de contexto conservan sus atributos presentados en (Serra, 2024).

4.3.4. Modelo completo de metadatos en la arquitectura de DL

En esta última sección se presentan las relaciones entre los modelos de metadatos de cada tipo, para definir el modelo de metadatos definitivo para la arquitectura.

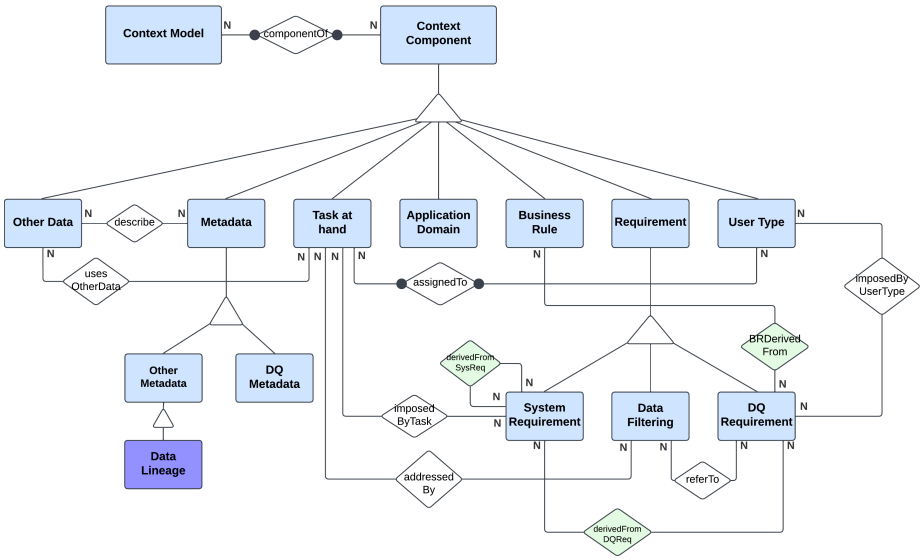

Figura 4.7: Modelo de metadatos de contexto en la arquitectura de DL

Modelo de metadatos de datos y procesos + Modelo de metadatos de calidad

Para relacionar el modelo de metadatos de datos y procesos (Figura 4.5) con el modelo de metadatos de calidad (Figura 4.6), en primer lugar se crea una relación *modelsDQFor* entre *Dataset* y *DQ Model*, que expresa sobre qué *dataset* es definido un modelo de contexto. Adicionalmente, de manera análoga a las relaciones que existían en la arquitectura entre la entidad *Measure* y los metadatos de los datos (ver Figura 3.6), se definen una serie de relaciones entre la entidad *Applied DQ Method* y las entidades *Dataset*, sus categorizaciones, y los elementos relacionados a las mismas:

- $appliedToDataset \subset AppliedDQMethod \times Dataset$
- $appliedToTuple \subset AppliedDQMethod \times Table$
- $appliedToData \subset AppliedDQMethod \times Column$
- $appliedToColumn \subset AppliedDQMethod \times Column$
- $appliedToElement \subset AppliedDQMethod \times Element$
- $appliedToArrayValue \subset AppliedDQMethod \times Field$
- $appliedToFieldValue \subset AppliedDQMethod \times Field$
- $appliedToField \subset AppliedDQMethod \times Field$
- $appliedToDoc \subset AppliedDQMethod \times Document$

En la Figura 4.8 se muestra la integración entre el modelo de metadatos de datos y procesos y el modelo de metadatos de calidad.

Modelo de metadatos de datos y procesos + Modelo de metadatos de contexto

Para la integración entre el modelo de metadatos de datos y procesos (Figura 4.5) y el modelo de metadatos de contexto (Figura 4.7), se define una relación *contextualizedDataset* entre las entidades *Dataset* y *Context Model*, que indica qué *Dataset* es considerado como el *Data At Hand* para el modelo de contexto instanicado. Además, se define la relación *group* entre *User* y *User Type*, que permite mantener la información de qué usuarios de la *Refined Zone* son agrupados por cuáles *User Types*. Por último, se agregan tres relaciones relativas al componente *Data Lineage*: *includesDataset* y *includesProcess*, definidas en la especificación del componente en la tabla 4.2, y la relación *hasLineage* entre un *Dataset* y un *Data Lineage*, que indica a de qué *dataset* describe el linaje la instancia de *Data Lineage*.

En la Figura 4.9 se muestra la integración entre el modelo de metadatos de datos y procesos y el modelo de metadatos de contexto.

Modelo de metadatos de calidad de datos + Modelo de metadatos de contexto

Para unificar los metadatos de calidad y los metadatos de contexto, se tomará como base la definición del metamodelo de calidad de datos basada en el contexto presentado en (Serra, 2024) (Figura 3.1) para relevar las relaciones entre los metadatos de calidad (Figura 4.6) y los metadatos de contexto (Figura 4.7). Por un lado, se incluyen las relaciones *suggestedBy*, *represents*, *influencedBy* y *uses* entre la entidad *Context Components* y las entidades *DQ Dimension*, *DQ Factor*, *DQ Metric* y *DQ Method* respectivamente. Además, se incluye una relación *contextualizesDQModel* entre *Context Model* y *DQ Model*, que representa el vínculo entre un modelo de contexto y los modelos de calidad que contextualiza. Se destaca que esta relación no presenta totalidad, ya que en la arquitectura se mantendrá la posibilidad de instanciar modelos de calidad que no respondan a ningún modelo de contexto.

En la Figura 4.10 se muestra la integración entre el modelo de metadatos de calidad y el modelo de metadatos de contexto.

Modelo de metadatos completo

Por último, en la Figura 4.11 se muestra el modelo de metadatos completo propuesto. Las entidades y relaciones propuestas en este capítulo se destacan en color verde.

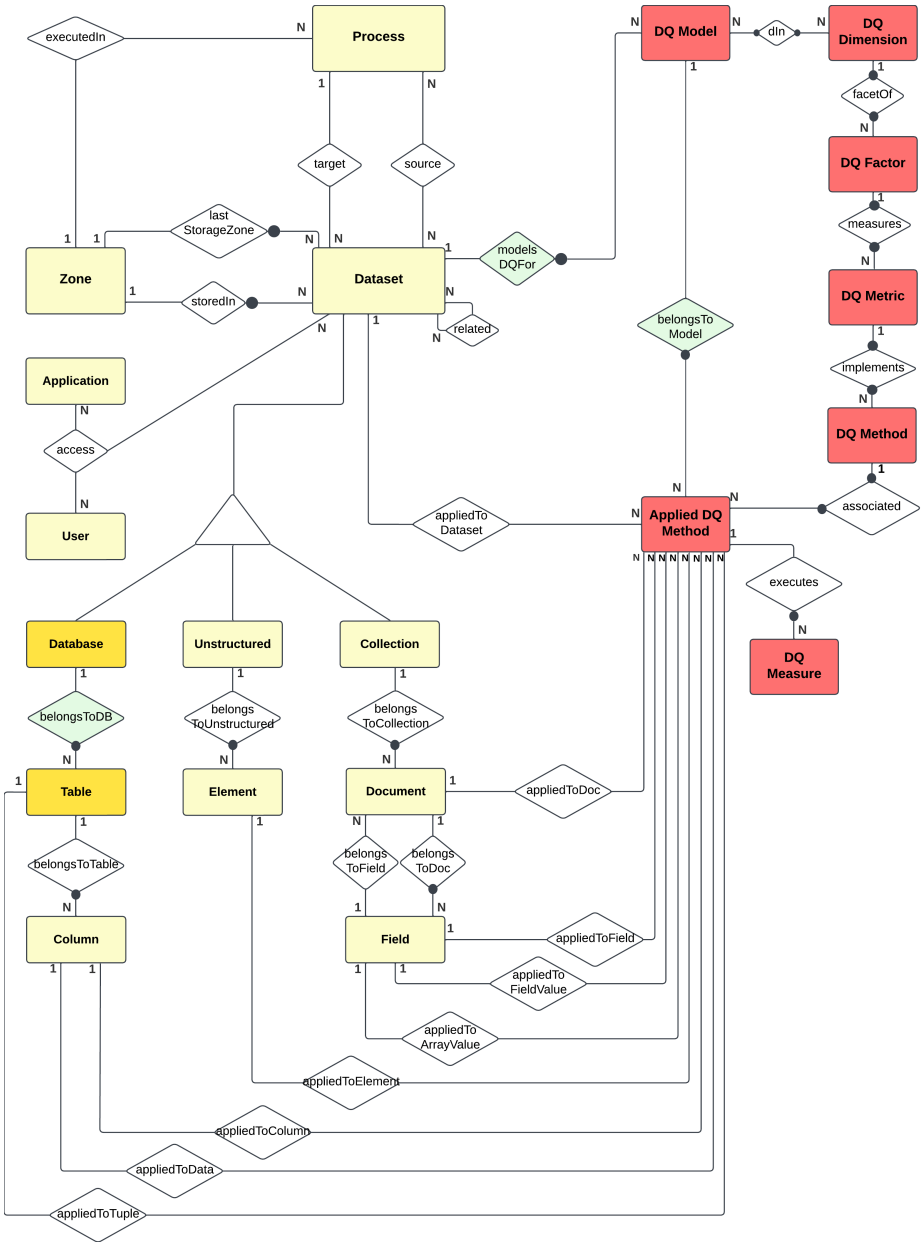

Figura 4.8: Relaciones entre el modelo de metadatos de datos y procesos y el modelo de metadatos de calidad de datos

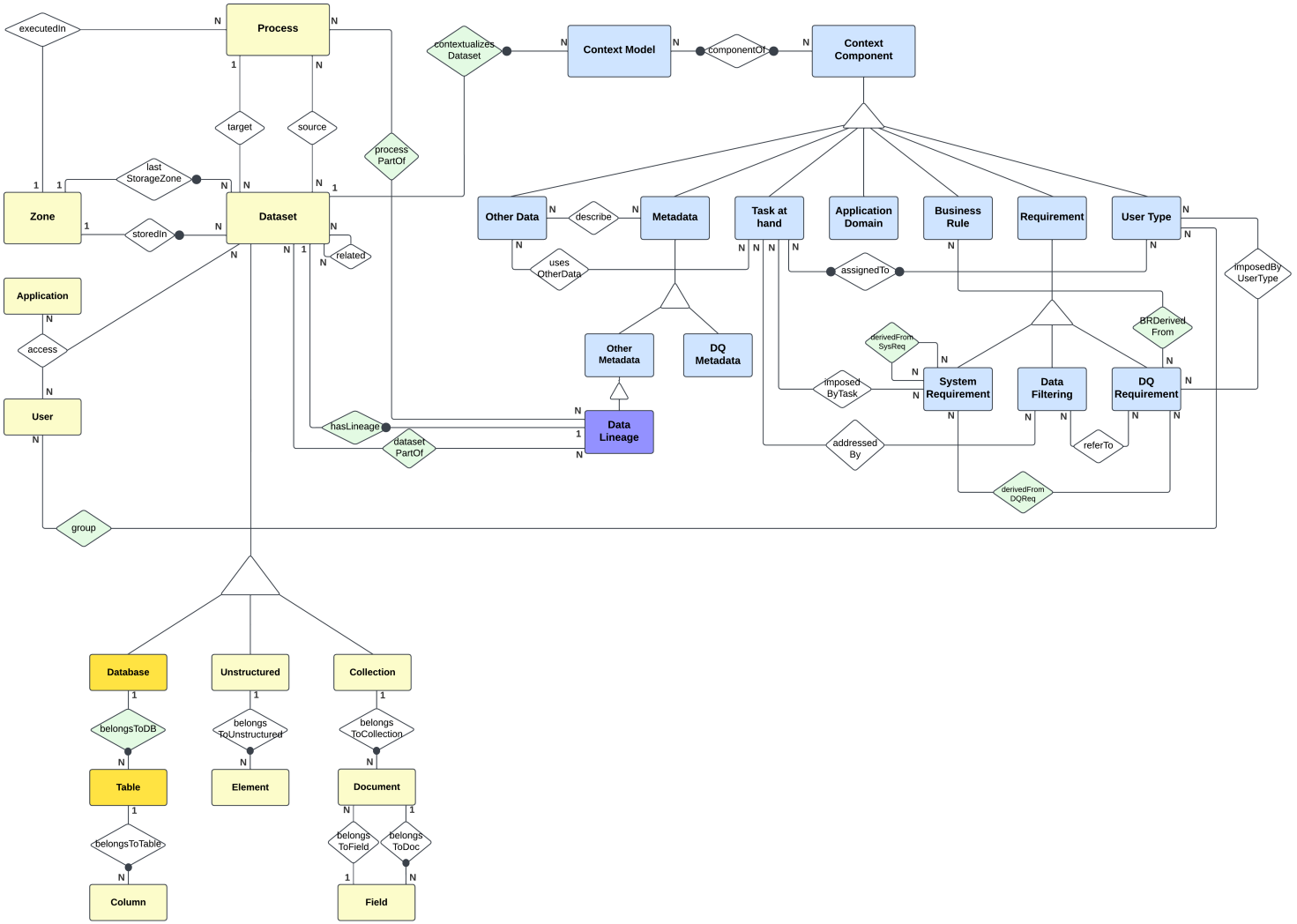

Figura 4.9: Relaciones entre el modelo de metadatos de datos y procesos y el modelo de metadatos de contexto

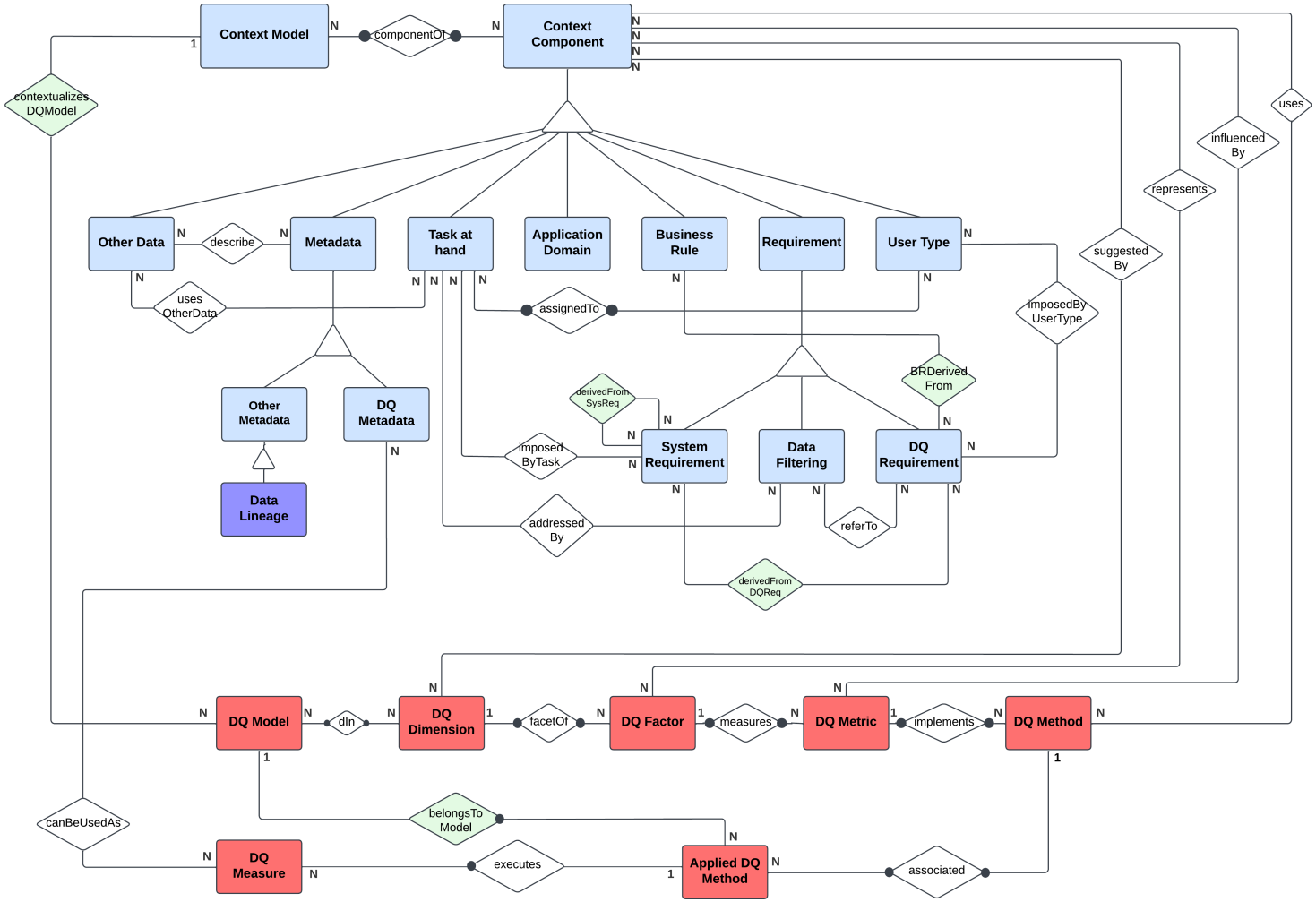

Figura 4.10: Relaciones entre el modelo de metadatos de contexto y el modelo de metadatos de calidad

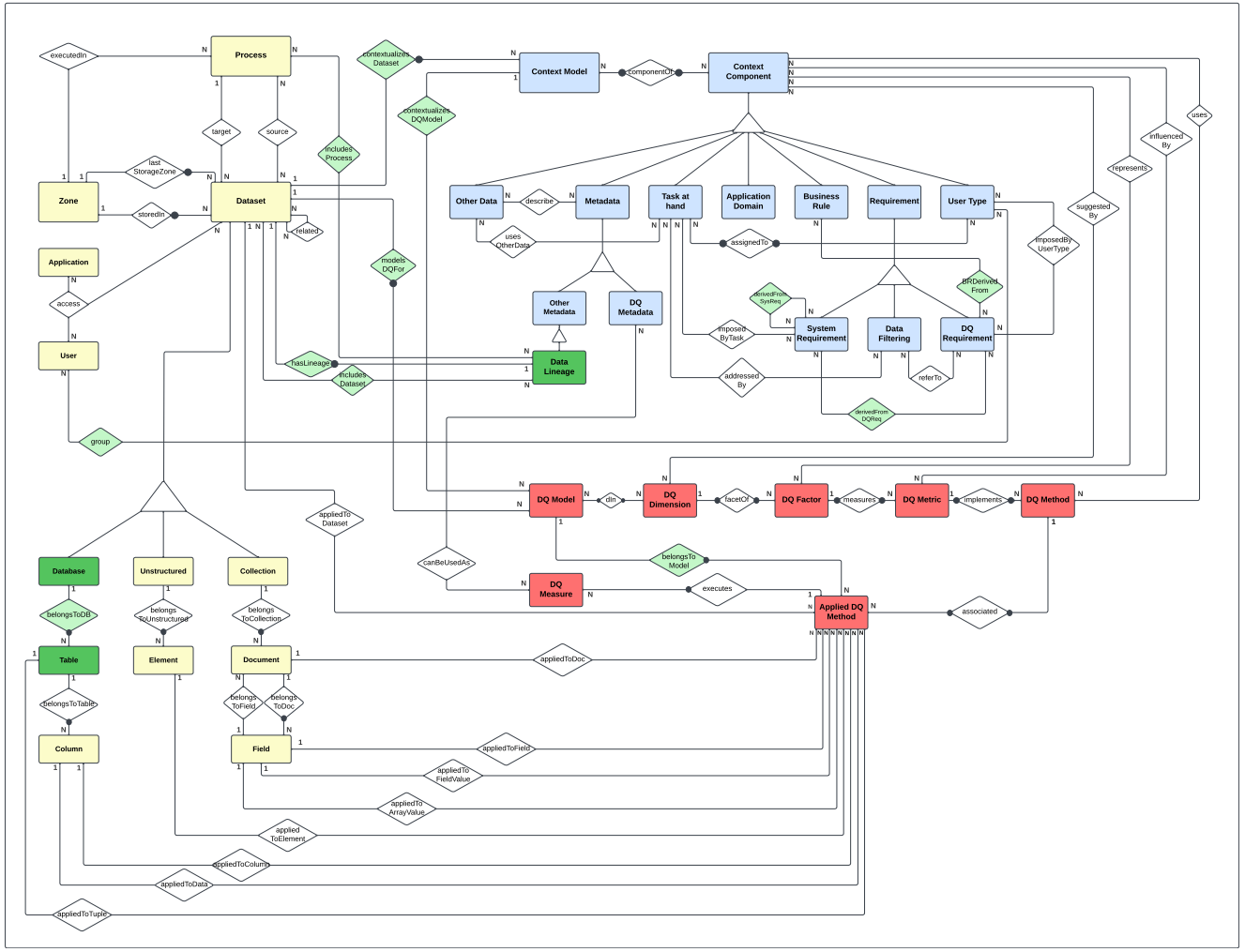

Figura 4.11: Modelo de metadatos completo.

Capítulo 5

Experimentación

En este capítulo se realiza un caso de estudio para validar la incorporación de modelos de contexto en la gestión de la calidad de los datos de la arquitectura, siguiendo la propuesta presentada en el Capítulo 4 (Propuesta de calidad de datos basada en contexto en *Data Lake*). El caso de estudio presentado es una adaptación del caso desarrollado en (Castro, 2025).

5.1. Descripción de la realidad y datos del *Data Lake*

El caso de estudio consiste de una empresa que se dedica al análisis y estudio del rendimiento de películas, tanto estrenadas como próximas a estrenar. Para ello, se implementa el DL como principal almacenamiento, ya que será necesario procesar grandes volúmenes de datos heterogéneos recopilados de distintas fuentes, que deberán ser procesados para distintos casos de uso y para llevar a cabo una variedad de tareas.

La empresa ofrece el servicio de asesorar a cadenas de cines interesadas en optimizar su programación. Estas cadenas buscan tomar decisiones informadas sobre qué películas proyectar, cómo distribuirlas en sus distintas salas, y qué estrategias de promoción aplicar, adaptándose al comportamiento y preferencias del mercado local.

Por lo tanto, los usuarios del DL tendrán tareas tanto descriptivas, donde analizan el desempeño de películas que se estrenaron en los últimos años, así como predictivas, donde se intenta anticipar el desempeño de nuevas películas en la región, ajustando su estrategia para maximizar la rentabilidad.

5.1.1. Fuentes de datos

El DL consume datos periódicamente de las plataformas *Kaggle* (Kaggle, s.f.) y *IMDb* (IMDb, s.f.), que publican datos de forma abierta. Estos datos pueden

ser consumidos a través de *APIs* o descargados de sus sitios web. A continuación se describen los *datasets* de interés, que son consumidos en la *Landing Zone*.

TMDB_movie_dataset_v11.csv

Este conjunto de datos está disponible públicamente en la plataforma *Kaggle*, donde es actualizado diariamente por su autor. El contenido proviene de la plataforma *TMDB* (*The Movie Database*), una de las principales fuentes de información sobre películas y series, junto con *IMDb*. Ambas plataformas ofrecen datos detallados sobre distintos tipos de shows, especialmente películas y series. Además de la información sobre los títulos, incluyen datos biográficos resumidos de los miembros del equipo involucrado en la producción, como el director y el elenco. El *dataset* se encuentra en formato *CSV*. En la tabla 5.1 se presentan los atributos más relevantes.

Atributo	**Descripción**
id	Identificador del registro en el *dataset*.
title	Título de la película
vote_average	Promedio de puntuación.
vote_count	Total de votos.
status	Estado de la película.
release_date	Fecha de estreno.
revenue	Recaudación total.
runtime	Duración en minutos.
adult	Indica si es solo para adultos (booleano).
budget	Costo de producción.
imdb_id	ID según el catálogo de IMDb.
original_language	Idioma original.
overview	Resumen de la película.
popularity	Puntaje de popularidad.
tagline	Frase publicitaria.
genres	Lista de géneros.
production_countries	Países de producción.
keywords	Palabras clave.
production_companies	Compañías productoras.

Tabla 5.1: Descripción de los atributos del *dataset TMDB_movie_dataset_v11.csv*

title.ratings.tsv

Este conjunto de datos es extraído del portal para desarrolladores *IMDb Developer*, y contiene información sobre la cantidad de votos y puntuación promedio de películas, según usuarios de la plataforma. Se encuentra en formato *TSV*. En la tabla 5.2 se presentan los atributos y sus descripciones.

Atributo	Descripción breve
tconst	Identificador único alfanumérico del título (string). Corresponde al ID usado en IMDb.
averageRating	Promedio ponderado de las calificaciones de los usuarios (float). Calculado en base a todos los votos registrados.
numVotes	Cantidad total de votos recibidos por el título (int). Refleja el número de usuarios que evaluaron el título.

Tabla 5.2: Descripción de los atributos del *dataset title.ratings.tsv*

title.crew.tsv

Al igual que el *dataset title.ratings.tsv*, este *dataset* es extraído de la plataforma *IMDb Developer*. Contiene información sobre los directores y guionistas que participaron en la producción del título. En la tabla 5.3 se presentan los atributos de este *dataset*.

Atributo	Descripción breve
tconst	Identificador único alfanumérico del título (string), usado también en otros archivos de IMDb.
directors	Lista de identificadores `nconst` del/los director(es) del título. Si hay más de uno, están separados por comas. Puede estar vacío si no hay datos disponibles.
writers	Lista de identificadores `nconst` del/los guionista(s) del título. También separados por comas; puede estar vacío.

Tabla 5.3: Descripción de los atributos del *dataset title.crew.tsv*

name.basics.tsv

Este *dataset* también es extraído de la plataforma *IMDb Developer*. Contiene información sobre personas en la industria del cine. En la tabla 5.4 se presentan los atributos de este *dataset*.

Atributo	Descripción breve
nconst	Identificador único alfanumérico de la persona (string), utilizado en toda la base de datos de IMDb.
primaryName	Nombre principal o más habitual con el que la persona aparece acreditada (string).
birthYear	Año de nacimiento en formato YYYY. Puede ser '\N' si no se conoce.
deathYear	Año de fallecimiento en formato YYYY si aplica; '\N' si no ha fallecido o no hay datos.
primaryProfession	Lista de hasta tres profesiones principales de la persona (array de strings separados por comas). Ej.: `actor,producer,writer`.
knownForTitles	Lista de identificadores `tconst` de los títulos más conocidos de la persona (array separados por comas).

Tabla 5.4: Descripción de los atributos del *dataset name.basics.tsv*

5.1.2. Usuarios y requisitos

Para incluir en este caso de estudio cuestiones relativas a usuarios y requerimientos de éstos, se realizan algunas suposiciones que se presentan a continuación.

- Dos tipos de usuarios: *Movie Warehouse Analyst* y *Movie Performance Specialist*. A ambos tipos de usuario se les concede el rol de *Data Analyst* en el DL, por lo que tienen acceso a los datos en la *Refined Zone*.

- Las fichas técnicas de cada tipo de usuario se detallan en las tablas 5.5 y 5.6.

- Las reglas de negocio comunes a ambos tipos de usuarios se presentan transversalmente en los procesos de la empresa:

  - Las películas se clasifican según su estado de producción, que puede ser "estrenada", "en producción", "en postproducción", "planificada", "rumoreada", o "cancelada".
  - Las fechas se manipulan con el formato de fecha ISO 8601 (*YYYY-MM-DD*).
  - Se toma como convención general que si una película no está estrenada, en los sistemas se le coloca la fecha 01/01/1800.
  - Las calificaciones de las películas se miden de 0 a 10, permitiendo números decimales.

Tipo de usuario	Movie Warehouse Analyst
Descripción de tareas	▪ Visualización de DW de películas, agrupados según las dimensiones Título, Director, País, Género, Fecha y Lenguaje. ▪ Generación de reportes y estadísticas en base a los datos de las películas disponibles en el sistema.
Requisitos específicos	▪ Ninguna de las dimensiones puede presentar valores vacíos. En el caso de no tener el dato, debe recurrirse a un valor por defecto, pero nunca puede ser vacío ni *null*. ▪ En el DW, deben respetarse en un 100 % las relaciones de clave foránea entre la tabla de hechos y las tablas de dimensión.

Tabla 5.5: Ficha técnica para los usuarios del tipo *Movie Warehouse Analyst*

Tipo de usuario	Movie Performance Specialists
Descripción de tareas	▪ Predicción de las calificaciones de películas sin estrenar en función de características como popularidad, género y duración. ▪ Identificación de tendencias en las fechas de estreno de películas. ▪ Análisis de la relación entre cantidad de votos, valoración promedio y popularidad para determinar los factores que contribuyen al éxito de una película. ▪ Identificación de compañías de producción exitosas según los votos que recibieron las películas que produjeron y analizar sus estrategias. El análisis varía dependiendo de si la película es para adultos o apta para todo público.
Requisitos específicos	▪ Los datos deben estar disponibles en su versión más actualizada todos los días a las 9am (GMT-3), ya que a esa hora comienza la ejecución programada del modelo predictivo de la aplicación del usuario. ▪ Los datos que representen nombres de países, productores, idiomas, o películas no deben presentar faltas de ortografía, y deben hacer uso correcto de letras mayúsculas. ▪ Los valores de las celdas no pueden ser vacíos ni nulos. Sí se permiten valores que representen explícitamente la falta de datos, como "Sin Datos" o fechas especiales. ▪ Deben estar presentes al menos 90 % de las películas estrenadas entre el año 2000 y 2024, tomando como referencia el catálogo de IMDb.

Tabla 5.6: Ficha técnica para los usuarios del tipo *Movie Performance Specialist*

5.1.3. Transformaciones

Para satisfacer los requerimientos de los usuarios, los *Data Engineers* del DL definen una serie de transformaciones sobre los datos:

- Los *datasets* son consumidos a través de la *Landing Zone*, donde no sufren ninguna modificación.
- Son almacenados en la *Raw Zone* en su formato original.
- Son procesados en las zonas *Trusted* y *Refined*.

A continuación se presentan las transformaciones por zona y por orden de ejecución.

Trusted Zone

Transformación 1: *tr:CleanTMDB*

Input: *dataset TMDB_movie_dataset_v11.csv* de la *Raw Zone*.
Acciones:

- Se seleccionan los atributos *id, title, imdb_id, genres, release_date, runtime, vote_average, vote_count, popularity, production_companies* y *status*.
- Se eliminan las filas que presenten valor nulo o vacío en el campo *imdb_id*.
- Se eliminan las filas duplicadas según todos sus atributos.
- Se realiza un *unnest* de las tuplas según el campo *directors* lo que genera una fila por cada director presente en la lista. Se renombra la columna a *director*.

Output: *TMDB_movies_clean.csv*, que se almacena en la *Trusted Zone*.

Transformación 2: *tr:MergeTMDBandRatings*

Input: *dataset TMDB_movies_clean.csv* (*Trusted Zone*) y el *dataset title.ratings.tsv* (*Raw Zone*)
Acciones:

- Se realiza un *LEFT JOIN* según los atributos *imdb_id* y *tconst*.
- Se eliminan las columnas *vote_average* y *vote_count*.
- Se renombran las columnas *averageRating* y *numVotes* a *vote_average* y *vote_count*, respectivamente.

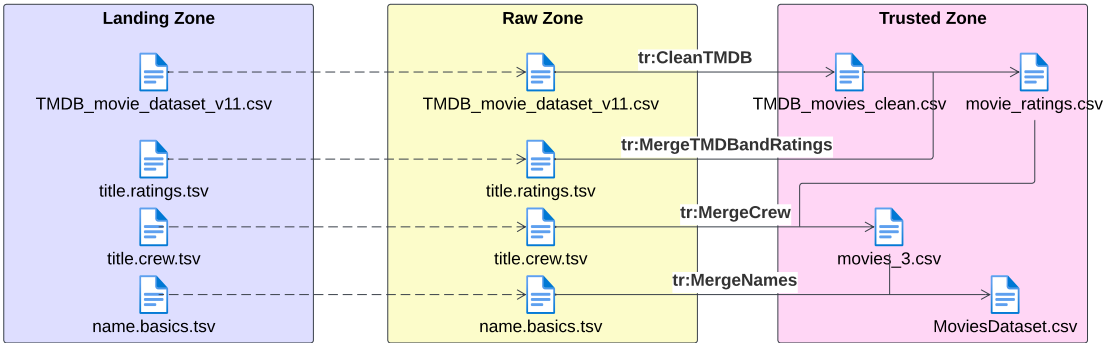

Figura 5.1: Transformaciones de la *Trusted Zone*

Output: *dataset movie_ratings.csv*, que se almacena en la *Trusted Zone*.

Como dato adicional, los *Data Engineers* del DL reportan que el tiempo combinado de las transformaciones *tr:MergeTMDBandRatings* y *tr:MergeCrew* es de aproximadamente 15 minutos para el volumen de datos esperado, en el peor caso.

Transformación 3: *tr:MergeCrew*

Input: *movie_ratings.csv* (*Trusted Zone*) y el *dataset title.crew.tsv* (*Raw Zone*).

Acciones:

- Se realiza un *LEFT JOIN* según los atributos *imdb_id* y *tconst*.
- Se elimina la columna *writers*.

Output: *dataset movies_3.csv*, que se almacena en la *Trusted Zone*.

Transformación 4: *tr:MergeNames*

Input: *movies_3.csv* (*Trusted Zone*) y el *dataset name.basics.tsv* (*Raw Zone*).

Acciones:

- Se realiza un *LEFT JOIN* según los atributos *director* y *nconst*.

Output: *MoviesDataset.csv*, que se almacena en la *Trusted Zone*.

En la Figura 5.1 se muestra un diagrama del procesamiento de los datos desde la *Landing Zone* hasta la *Trusted Zone*.

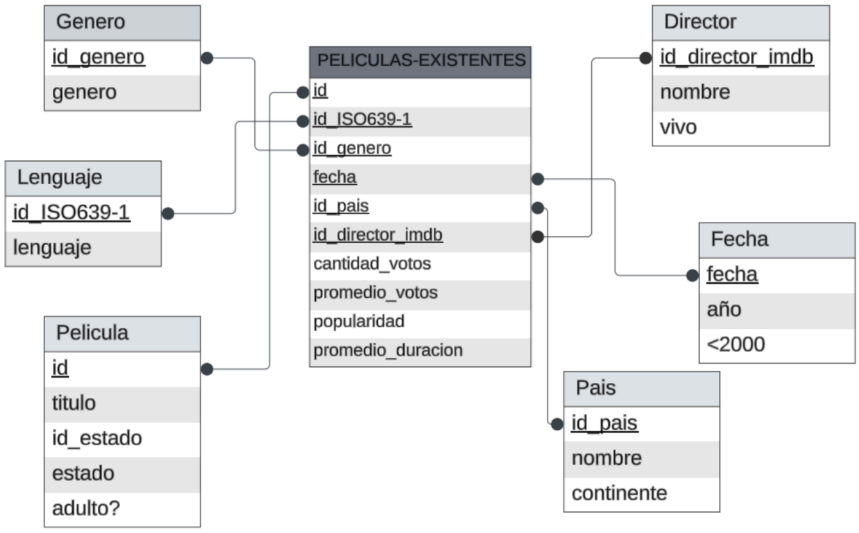

Figura 5.2: Modelado dimensional para *Movies DW* (Castro, 2025)

Refined Zone

En la *Refined Zone* se realizan dos transformaciones para satisfacer los requerimientos de los *Movie Performance Specialists* y *Movie Warehouse Analyst*, según se indicaron en sus respectivas fichas técnicas, presentadas en la Sección 5.1.2.

En primer lugar, para satisfacer el requerimiento de los usuarios *Movie Warehouse Analyst*, se define el *Data Warehouse "Movies DW"* en la *Refined Zone*. En la Figura 5.2, tomada de (Castro, 2025), se presenta el modelado dimensional realizado para el el *Data Warehouse "Movies DW"*. El DW es poblado por la transformación ***tr:CreateWarehouse***.

Transfomación 5: *tr:CreateWarehouse*

Input: *dataset MoviesDataset.csv* (*Trusted Zone*).
Acciones:

- Se generan las tablas de hechos (*fact_table.csv*), y las tablas de dimensión (*dim_genero.csv*, *dim_lenguaje.csv*, *dim_pelicula.csv*, *dim_pais.csv*, *dim_fecha.csv*, *dim_director.csv*).

Output: La tabla de hechos (*fact_table.csv*), y las tablas de dimensión (*dim_genero.csv*, *dim_lenguaje.csv*, *dim_pelicula.csv*, *dim_pais.csv*, *dim_fecha.csv*, *dim_director.csv*), que se almacenan en la *Refined Zone*.

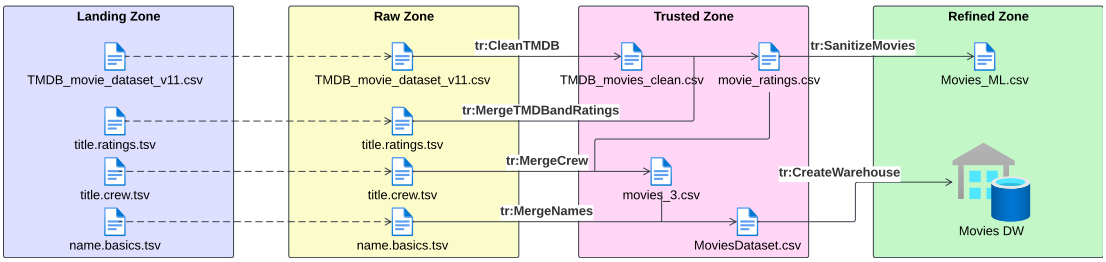

Figura 5.3: Transformaciones del *Data Lake*

Por último, para satisfacer las necesidades del usuario del tipo *Movie Performance Specialist*, se implementa la transformación ***tr:SanitizeMovies***.

Transfomación 6: *tr:SanitizeMovies*

Input: *dataset movie_ratings.csv* (*Trusted Zone*).
Acciones:

- Se sustituyen los valores nulos por el valor "Sin Datos" para todas las columnas a excepción de *release_date*.
- Se sustituyen los valores nulos en la columna *release_date* por el valor "1800-01-01"

Output: *Movies_ML.csv*, que se almacena en la *Refined Zone*.

Se destaca que la transformación *tr:SanitizeMovies* utiliza el *dataset movie_ratings.csv* como *input* porque el *Movie Performance Specialist* no hace uso de los datos de los directores para sus análisis. Como dato adicional, los *Data Engineers* del DL reportan que el tiempo de ejecución de esta transformación es de aproximadamente 15 minutos en el peor caso, para el volumen de datos esperado.

En la Figura 5.3 se muestran todas las transformaciones detalladas en esta sección.

5.2. Modelos de contexto

En base a las fichas técnicas relevadas en la Sección 5.1.2, se definen modelos de contexto para los *datasets*. Este análisis se centra en el caso de uso de los usuarios del tipo *Movie Performance Specialist*, por lo que se definen modelos

de contexto para *datasets* involucrados en este caso de uso. Adicionalmente, para mostrar la convivencia de modelos de contexto de distintos *datasets* en una misma zona del DL, se definen también modelos de contexto para el DW en la *Refined Zone* utilizado por los usuarios *Movie Warehouse Analyst*, pero no se estudian los *datasets* correspondientes a este caso en las zonas anteriores.

Para la definición de los modelos de contexto, en cada zona se contemplan los componentes indicados en la tabla 4.1. Además, se destaca la inclusión del componente *Data Lineage* en todos los modelos, donde se referencian todos los procesos que participan en el proceso de creación de los *Data at Hand*, como se indicaron en la Sección 5.1.3. Los archivos *JSON* correspondientes a los linajes se detallan en el Anexo A.

Refined Zone

En la tabla 5.7 se presenta el modelo de contexto para el DW instanciado en la *Refined Zone* para satisfacer el caso de uso de los tipos de usuarios *Movie Warehouse Analyst*, que incluye la tabla de hechos y las tablas de dimensión.

Tabla 5.7: Modelo de contexto para el DW

Data at hand	[fact_table.csv, dim_genero.csv, dim_lenguaje.csv, dim_pelicula.csv, dim_pais.csv, dim_fecha.csv, dim_director.csv]
Context Model	
ctxm1	
Application Domain	
appDom	Name: Cinematografía
Business Rules	
BR1	Statement: Estado $\in \{Released, InProduction, PostProduction, Planned, Rumored, Canceled\}$
	Semantic: Las películas se categorizan según su estado de producción, y se dividen en las categorías 'Released', 'In Production', 'Post Production', 'Planned', 'Rumored' y 'Canceled'.
BR2	Statement: Estado $\neq Released \rightarrow release\_date = 1800-01-01$
	Semantic: Para las películas cuyo estado no sea "Released", el campo *release_date* se debe setear al valor 1800-01-01
BR3	Statement: *format(release_date)* = "YYYY-MM-DD"
	Semantic: Las fechas se representan en formato "YYYY-MM-DD"
User Types	

Continued on next page

Tabla 5.7: Modelo de contexto para el DW (Continued)

UT1	Name: Movie Warehouse Analyst
	Characteristics: Data Analyst
Tasks at Hand	
T1	Name: Visualización de datos según un modelado dimensional
	Purpose: Visualizar datos de las películas, agrupados según las dimensiones: Película, Director, País, Género, Fecha y Lenguaje
	Assigned to: UT1
	Uses: -
T2	Name: Generación de reportes y estadísticas
	Purpose: Generar reportes y estadísticas en base a los datos de las películas disponibles en el sistema.
	Assigned to: UT1
	Uses: -
DQ Requirements	
DQR1	Statement: Ninguno de los datos puede presentar valores vacíos.
	Description: Ningún dato puede tomar el valor “ .º null.
	Imposed by: UT1
	Refer to: -
DQR2	Statement: Las claves de la tabla de hechos deben existir en las tablas de dimensión.
	Description: Todos los valores de los campos id, id_ISO639-1, id_genero, fecha, id_pais e id_director_imdb del archivo fact_table.csv deben existir como claves en los archivos dim_pelicula.csv, dim_lenguaje.csv, dim_genero.csv, dim_fecha.csv, dim_pais.csv y dim_director.csv respectivamente.
	Imposed by: UT1
	Refer to: -
Data filtering	
-	
System Requirements	
SR1	Statement: Modelado dimensional
	Description: El *dataset* debe presentar un modelado dimensional, y ser compatible con herramientas de análisis OLAP

Continued on next page

Tabla 5.7: Modelo de contexto para el DW (Continued)

	Imposed by: T1
DQ Metadata	
-	
Other Metadata	
-	
Data Lineage	
DL1	Sources: [“kaggle”, “IMDb”]
	Lineage: DW_lineage.json
	includes datasets: [MoviesDataset.csv, movies_3.csv, movie_ratings.csv, TMDB_movies_clean.csv, TMDB_movie_dataset_v11.csv, title.ratings.tsv, title.crew.tsv, name.basics.tsv]
	includes processes: [tr:CreateWarehouse, tr:MergeNames, tr:MergeCrew, tr:MergeTMDBandRatings, tr:CleanTMDB]
Other Data	
-	

En la tabla 5.8 se presenta el modelo de contexto para el *dataset MoviesML.csv* en la *Refined Zone*, que satisface el caso de uso de los tipo de usuarios *Movie Performance Specialist*.

Tabla 5.8: Modelo de contexto para MoviesML.csv

Data at hand:	[MoviesML.csv]
Context Model	
ctxm2	
Application Domain	
appDom	Cinematografía
Business Rules	
BR1	Statement: Estado $\in \{Released, InProduction, PostProduction, Planned, Rumored, Canceled\}$
	Semantic: Las películas se categorizan según su estado de producción, y se dividen en las categorías ‘Released’, ‘In Production’, ‘Post Production’, ‘Planned’, ‘Rumored’ y ‘Canceled’.

Continued on next page

Tabla 5.8: Modelo de contexto para MoviesML.csv (Continued)

BR2	Statement: Estado $\neq$ *Released* $\rightarrow$ *release_date* $= 1800-01-01$
	Semantic: Para las películas cuyo estado no sea "Released", el release_date se debe setear al valor 1800-01-01
BR3	Statement: *format(release_date)* = "YYYY-MM-DD"
	Semantic: Las fechas se representan en formato "YYYY-MM-DD"
BR4	Statement: Rating $\in [0, 10{,}0]$
	Semantic: El rating de una película se representa con un número real entre 0 y 10
User Types	
UT2	Name: Movie Performance Specialist
	Characteristics: Data Analyst
Tasks at Hand	
T3	Name: Predicción de calificaciones
	Purpose: Predecir las calificaciones de películas en función de características como popularidad, género y duración.
	Assigned to: UT2
	Uses: -
T4	Name: Identificación de tendencias de fechas de estreno
	Purpose: Identificar tendencias en las fechas de estreno de películas.
	Assigned to: UT2
	Uses: -
T5	Name: Identificación de factores de éxito
	Purpose: Analizar la relación entre cantidad de votos, valoración promedio y popularidad para determinar los factores que contribuyen al éxito de una película.
	Assigned to: UT2
	Uses: -
T6	Name: Análisis de estrategias efectivas de marketing para películas de adultos y niños
	Purpose: Identificar compañías de producción exitosas según los votos que recibieron las películas que produjeron y analizar sus estrategias. Se debe hacer un análisis distinto dependiendo de si la película es para adultos o apta para todo público.

Continued on next page

Tabla 5.8: Modelo de contexto para MoviesML.csv (Continued)

	Assigned to: UT2
	Uses: -
DQ Requirements	
DQR3	Statement: Los datos del día anterior deben estar actualizados y disponibles todos los días a las 9am (GMT-3).
	Description: Los datos deben estar disponibles en su versión más actualizada todos los días a las 9am (GMT-3), ya que a esa hora comienza la ejecución programada del modelo predictivo de la aplicación del usuario.
	Imposed by: UT2
	Refer to: -
DQR4	Statement: Los datos que representen nombres deben estar bien escritos
	Description: Los campos del tipo *String* que representen nombres de países, nombres de productores, nombres de idiomas, o nombres de películas no deben presentar faltas de ortografía, y deben hacer uso correcto de letras mayúsulas
	Imposed by: UT2
	Refer to: -
DQR5	Statement: Ningún dato puede tomar el valor “”ni null.
	Description: Los valores de las celdas no pueden tener valores vacíos ni nulos. Sí se permiten valores que representen explícitamente la falta de datos, como “Sin Datos” o fechas especiales
	Imposed by: UT2
	Refer to: -
DQR6	Statement: Deben estar presentes al menos 90 % de las películas estranadas entre el año 2000 y 2024.
	Description: Tomando como referencia el catálogo de IMDb, se debe contar con una cobertura mayor o igual a 90 % para las películas estrenadas entre el año 2000 y 2024
	Imposed by: UT2
	Refer to: -

Continued on next page

Tabla 5.8: Modelo de contexto para MoviesML.csv (Continued)

Data filtering	
DF1	Statement: SELECT * FROM movies WHERE adult = “True”
	Description: Filtrar registros que indiquen que la película es para adultos
	Addressed by: T6
DF2	Statement: SELECT * FROM movies WHERE adult = “False”
	Description: Filtrar registros que indiquen que la película es apta para todo público
	Addressed by: T6
System Requirements	
-	
DQ Metadata	
-	
Other Metadata	
-	
Data Lineage	
DL2	Sources: [“kaggle”, “IMDb”]
	Lineage: Movies_ML_lineage.json
	includes datasets: [Movies_ML_merged_clean.csv, Movies_ML_merged.csv, TMDB_Movies_clean.csv, title.crew.tsv, TMDB_movie_dataset_v11.csv]
	includes processes: [CleanTMDB, MergeTMDBandIMDb, CleanMerge, SanitizeMovies]
Other Data	
-	

Trusted Zone

Como se mencionó en la Sección 4.1.1, en los modelos de contexto de la *Trusted Zone* no participan:

- *User Types*
- *Tasks at Hand*
- *DQ Requirements*

En la tabla 5.9 se presenta el modelo de contexto para el *dataset movie_ratings.csv*. En este modelo, se incorporan componentes de contexto que

son el resultado de la propagación hacia atrás de los componentes (ver Sección 4.1.2). En particular, se observa que la *Business Rule* BR5 y el *System Requirement* SR2 se derivan de *DQ Requirements* presentes en el modelo de contexto del *dataset MoviesML.csv* (tabla 5.8), cuya definición utiliza el conocimiento de la latencia de los procesos posteriores, mencionados en la Sección 5.1.3.

Tabla 5.9: Modelo de contexto para movie_ratings.csv

Data at hand:	[movie_ratings.csv]
Context Model	
ctxm3	
Application Domain	
appDom	Cinematografía
Business Rules	
BR1	Statement: Estado $\in \{Released, InProduction, PostProduction, Planned, Rumored, Canceled\}$
	Semantic: Las películas se categorizan según su estado de producción, y se dividen en las categorías 'Released', 'In Production', 'Post Production', 'Planned', 'Rumored' y 'Canceled'.
BR3	Statement: format(release_date) = "YYYY-MM-DD"
	Semantic: Las fechas se representan en formato "YYYY-MM-DD"
BR4	Statement: Rating $\in [0, 10{,}0]$
	Semantic: El rating de una película se representa con un número real entre 0 y 10
BR5	Statement: Deben estar presentes al menos 90 % de las películas estranadas entre el año 2000 y 2024. La fuente de verdad es el catálogo online de IMDb
	Semantic: Tomando como fuente de verdad el catálogo de IMDb, se debe contar con una cobertura mayor o igual a 90 % para las películas estrenadas entre el año 2000 y 2024
	Derived from: DQR6
System Requirements	
SR2	Statement: Los datos del día anterior anterior deben estar actualizados y disponibles todos los días a las 8:45am (GMT-3).
	Description: Los datos deben estar disponibles en su versión más actualizada todos los días a las 8:45am (GMT-3)

Continued on next page

Tabla 5.9: Modelo de contexto para movie_ratings.csv (Continued)

	Derived from DQ Requirement: DQR3
DQ Metadata	
-	
Other Metadata	
-	
Data Lineage	
DL3	Sources: ["kaggle", "IMDb"]
	Lineage: movie_ratings_lineage.json
	includes datasets: [TMDB_movies_clean.csv, TMDB_movie_dataset_v11.csv, title.ratings.tsv]
	includes processes: [tr:MergeTMDBandRatings, tr:CleanTMDB]
Other Data	
-	

Landing Zone

Por último, se definen modelos de contexto para los *datasets* que ingresan a la *Landing Zone*, y que participan en el linaje del *dataset Movies_ML.csv*.

Como se mencionó en la Sección 4.1.1, los modelos de contexto de *datasets* de esta zona no incluyen componentes del tipo *Other Data*.

En la tabla 5.10 se presenta el modelo de contexto para el *dataset* proveniente de TMDB, *TMDB_movie_dataset_v11.csv*, mientras que en la tabla 5.11 se presenta el modelo de contexto para el *dataset* consumido de *IMDb*, *title.ratings.tsv*.

En esta zona, se presenta un *System Requirement* derivado de un *System Requirement* de modelos de contexto de la zona *Trusted*. Para definir este requerimiento, se utiliza el conocimiento de la latencia de los procesos posteriores, mencionados en la Sección 5.1.3. Además, se presenta un *System Requirement* que está dado por la arquitectura, el cual exige la codificación de los datos de texto en el formato *UTF-8*.

Tabla 5.10: Modelo de contexto para TMDB_movie_dataset_v11.csv

Data at hand:	[TMDB_movie_dataset_v11.csv]
Context Model	
ctxm4	

Continued on next page

Tabla 5.10: Modelo de contexto para TMDB_movie_dataset_v11.csv (Continued)

Application Domain	
appDom	Cinematografía
Business Rules	
BR1	Statement: Estado $\in \{Released, InProduction, PostProduction, Planned, Rumored, Canceled\}$
	Semantic: Las películas se categorizan según su estado de producción, y se dividen en las categorías 'Released', 'In Production', 'Post Production', 'Planned', 'Rumored' y 'Canceled'.
BR3	Statement: *format(release_date)* = "YYYY-MM-DD"
	Semantic: Las fechas se representan en formato "YYYY-MM-DD"
BR5	Statement: Deben estar presentes al menos 90 % de las películas estrenadas entre el año 2000 y 2024. La fuente de verdad es el catálogo online de IMDb
	Semantic: Tomando como fuente de verdad el catálogo de IMDb, se debe contar con una cobertura mayor o igual a 90 % para las películas estrenadas entre el año 2000 y 2024
	Derived from: DQR6
System Requirements	
SR3	Los datos nuevos deben ser ingeridos diariamente y estar disponibles 8:30am (GMT-3).
	Description: Los datos deben estar disponibles en su versión más actualizada todos los días a las 8:30am (GMT-3)
	Derived from Sys Requirement: SR2
SR4	Statement: Soporte UTF-8
	Description: Todos los caracteres de los datos deben estar encodeados con caracteres compatibles con UTF-8
DQ Metadata	
-	
Other Metadata	
OM1	Path: https://www.kaggle.com/datasets/asaniczka/tmdb-movies-dataset-2023-930k-movies?select=TMDB_movie_dataset_v11.csv
	Description: Metadata available at the dataset's origial publication
	Author: asaniczka

Continued on next page

Tabla 5.10: Modelo de contexto para TMDB_movie_dataset_v11.csv (Continued)

	Last update: -
Data Lineage	
DL4	Sources: ["kaggle"]
	Lineage: -
	includes datasets: -
	includes processes: -

Tabla 5.11: Modelo de contexto para title.ratings.tsv

Data at hand:	[title.ratings.tsv]
Context Model	
ctxm5	
Application Domain	
appDom	Cinematografía
Business Rules	
BR4	Statement: Rating $\in [0, 10{,}0]$
	Semantic: El rating de una película se representa con un número real entre 0 y 10
System Requirements	
SR3	Statement: Los datos nuevos deben ser ingeridos diariamente y estar disponibles 7am (GMT-3).
	Description: Los datos deben estar disponibles en su versión más actualizada todos los días a las 7am (GMT-3)
	Derived from Sys Requirement: SR2
SR4	Soporte UTF-8
	Description: Todos los caracteres de los datos deben estar encodeados con caracteres compatibles con UTF-8
DQ Metadata	
-	

Tabla 5.11: Modelo de contexto para title.ratings.tsv (Continued)

Other Metadata	
OM2	Path: https://developer.imdb.com/non-commercial-datasets/#titleratingstsvgz
	Description: Dataset and column description available from IMDb
	Author: IMDb
	Last update: -
Data Lineage	
DL5	Sources: ["IMDb"]
	Lineage: -
	includes datasets: -
	includes processes: -

Se observa que los modelos de contexto correspondientes a las zonas de menor procesamiento del DL, tienden a ser más reducidos. Este resultado es razonable, ya que al tratarse de *datasets* cuya intención es retener generalidad, para poder ser posteriormente adaptados a varios casos de uso, presentan contextos menos específicos. Por lo tanto, sus modelos de contexto presentan una menor cantidad de componentes.

5.3. Modelos de calidad de datos

A partir de los modelos de contexto ligados a los *datasets* que participan en el caso de uso de los tipos de usuarios *Movie Performance Specialist*, se definen modelos de calidad de datos (en adelante, modelos de CD) asociados, los cuales se presentan en las Figuras 5.4, 5.5 y 5.6.

Estos modelos de CD son basados en el contexto y se especifica, para cada elemento, su relación con el modelo de contexto del *dataset*, si aplica. Se destaca que no se exige que todos los elementos del modelo de CD deban relacionarse con el modelo de contexto, como es el caso de la dimensión *Cantidad de Datos* presente en el modelo de CD 5.6.

Comparando los modelos de CD 5.4, 5.5 y 5.6, se observa que, al igual que los modelos de contexto, los modelos de calidad de datos basada en el contexto tienden a ser más reducidos en las zonas de menor procesamiento del DL.

DQ Dimension	DQ Factor	DQ Metric	DQ Method	Applied DQ Method
ID: D1 Name: Exactitud Semantic: Concierne la correctitud y la precisión con que los datos del mundo real son representados en un sistema de información Suggested by: {BR3, T4, DQR4}	ID: D1F1 Name: Exactitud Sintáctica Semantic: Indica qué tan libre de errores sintácticos están los datos Represents: {BR3, DQR4}	ID: M1 Name: FormatoFecha Purpose: Mide si el formato de un dato que representa una fecha es correcto, relativo a un formato especificado. No se mide para datos faltantes. Influenced by: {BR3} Granularity: celda Result domain: {0,1}	ID: ME1 Name: verificarFormatoFecha Uses: {BR3} Input data types: String, String Output data types: Boolean Algorithm: verificarFormatoFecha(dato, formato)	ID: AME1 Applied to: Movies_ML.csv, “release_date”
		ID: M2 Name: NombreCorrecto Purpose: Mide si un dato que representa un nombre está bien escrito, sin imporar las mayúsculas Influenced by: {DQR4} Granularity: celda Result domain: {0,1}	ID: ME2 Name: verificarNombre Uses: {DQR4} Input data types: String, Dataset Output data types: Boolean Algorithm: compararNombreConReferencial(dato, referencial)	ID: AME2 Applied to: Movies_ML.csv, “title”
				ID: AME3 Applied to: Movies_ML.csv, “production_countries”
	ID: D1F2 Name: Exactitud Semántica Semantic: Indica qué tan correctos son los datos, relativos a la realidad Represents: {T4}	ID: M3 Name: FechaCorrecta Purpose: Mide si una fecha es correcta, dado un referencial Influenced by: {T4} Granularity: celda Result domain: {0,1}	ID: ME3 Name: verificarFechaCorrecta Uses: {T4} Input data types: Date, String, dataset Output data types: Boolean Algorithm: fechaCorrecta(fecha, titulo, referencial)	ID: AME4 Applied to: Movies_ML.csv, “release_date”
ID: D2 Name: Completitud Semantic: Concierne la proporción del mundo real que está representado en el SI Suggested by: {DQR5, DQR6, DL2, T3, T5, T6}	ID: D2F1 Name: Densidad Semantic: Indica qué cantidad de registros con datos faltantes hay Represents: {DQR5, DL2, T3, T5, T6}	ID: M4 Name: RatioNoVacios Purpose: Mide el grado de valores no faltantes en una columna. Lo que se considera faltante depende del método. Influenced by: {DQR5, DL2, T3, T5, T6} Granularity: columna Result domain: [0..1]	ID: ME4 Name: ratioNoNulos Uses: {DQR5, DL2} Input data types: Object[] Output data types: Float Algorithm: ratioNoNulos(columna)	ID: AME5 Applied to: Movies_ML.csv, “title”
				ID: AME6 Applied to: Movies_ML.csv, “genres”
				ID: AME7 Applied to: Movies_ML.csv, “release_date”
				ID: AME8 Applied to: Movies_ML.csv, “vote_average”
				ID: AME9 Applied to: Movies_ML.csv, “vote_count”
				ID: AME10 Applied to: Movies_ML.csv, “production_countries”
			ID: ME5 Name: ratioNoNulosNiSD Uses: {T3, T5, T6} Input data types: Object[], String[] Output data types: Float Algorithm: ratioNoNulosNiSD(columna, ref)	ID: AME11 Applied to: Movies_ML.csv, “genres”
				ID: AME12 Applied to: Movies_ML.csv, “release_date”
				ID: AME13 Applied to: Movies_ML.csv, “vote_average”
				ID: AME14 Applied to: Movies_ML.csv, “vote_count”
				ID: AME15 Applied to: Movies_ML.csv, “production_countries”
	ID: D2F2 Name: Cobertura Semantic: Indica qué cantidad de datos de la realidad están representados en el sistema Represents: {DQR6}	ID: M5 Name: PorcentajeCobertura Purpose: Mide el porcentaje de valores que se encuentran representados por algún registro del dataset, relativo a un referencial Influenced by: {DQR6} Granularity: dataset Result domain: [0..1]	ID: ME6 Name: porcentajeCoberturaRef Uses: {DQR6} Input data types: Object[], String[] Output data types: Float Algorithm: porcentajeCoberturaRef(columna, ref)	ID: AME16 Applied to: Movies_ML.csv
ID: D3 Name: Consistencia Semantic: Concierne la consistencia entre los registros del SI Suggested by: {BR1, BR2, BR4, T3, DL2}	ID: D3F1 Name: Integridad de Dominio Semantic: Indica que los datos representados estén en el dominio, dada su semántica y el dominio de aplicación Represents: {BR1, BR4, T3, DL2}	ID: M6 Name: DatoEnDominioString Purpose: Mide si un dato de tipo texto es posible en el dominio, dado un referencial Influenced by: {BR1} Granularity: celda Result domain: {0,1}	ID: ME7 Name: StatusEnDominio Uses: {BR1} Input data types: String, String[] Output data types: Boolean Algorithm: verificarStatus(status, posiblesStatus)	ID: AME17 Applied to: Movies_ML.csv, “status”
		ID: M7 Name: DatoEnRango Purpose: Mide si un dato numérico es positivo Influenced by: {BR4, T3, DL2} Granularity: celda Result domain: {0,1}	ID: ME8 Name: DatoEnRango Uses: {BR4, T3, DL2} Input data types: Float, Float, Float Output data types: Boolean Algorithm: verificarNumeroEnRango(dato, min, max)	ID: AME18 Applied to: Movies_ML.csv, “vote_average”
				ID: AME19 Applied to: Movies_ML.csv, “vote_count”
	ID: D3F2 Name: Integridad Intrarrelación Semantic: Indica que los valores de dos o más atributos de un registro sean consistentes entre sí Represents: {BR2}	ID: M8 Name: IntegridadFechas Purpose: Mide si una fecha es válida, dado otro dato Influenced by: {BR2} Granularity: par de celdas Result domain: {0,1}	ID: ME9 Name: IntegridadFechasReleased Uses: {BR2} Input data types: Date, String Output data types: Boolean Algorithm: fechaEsValidaSegunStatus(fecha, status)	ID: AME20 Applied to: Movies_ML.csv, “release_date”, “status”
ID: D4 Name: Unicidad Semantic: Concierne el grado de repetición de los registros del SI Suggested by: {DL2}	ID: D4F1 Name: No-Duplicación Semantic: Mide el grado de filas que son únicas Represents: {DL2}	ID: M9 Name: RatioClavesUnicas Purpose: Mide el grado de filas de un dataset que son únicas, según una fila identificadora Influenced by: {DL2} Granularity: dataset Result domain: [0..1]	ID: ME10 Name: RatioClavesUnicas Uses: {DL2} Input data types: Object[][], Object[] Output data types: Float Algorithm: ratioNoDupFilas(dataset, columna)	ID: AME21 Applied to: Movies_ML.csv, “id”
ID: D5 Name: Frescura Semantic: Concierne el grado en el que la edad de los datos afecta su uso Suggested by: {DQR3}	ID: D5F1 Name: Oportunidad Semantic: Mide qué tan oportunos son los datos a la hora de su lectura Represents: {DQR3}	ID: M10 Name: OportunidadPuntaje Purpose: Calcula la diferencia en minutos entre la hora en la que se necesitan los datos y la hora en la que se actualizan Influenced by: {DQR3} Granularity: dataset Result domain: [0..+inf)	ID: ME11 Name: Oportunidad9am Uses: {DQR3} Input data types: Object[][] Output data types: Integer Algorithm: timelinessScore(dataset)	ID: AME22 Applied to: Movies_ML.csv
		ID: M11 Name: Edad Purpose: Calcula el tiempo desde la última actualización de los datos, en una unidad de tiempo dada Influenced by: {DQR3} Granularity: dataset Result domain: [0..+inf)	ID: ME12 Name: EdadMinutos Uses: {DQR3} Input data types: Object[][] Output data types: Integer Algorithm: staleness(dataset)	ID: AME23 Applied to: Movies_ML.csv

Figura 5.4: Modelo de calidad de datos basada en el contexto para MoviesML.csv

DQ Dimension	DQ Factor	DQ Metric	DQ Method	Applied DQ Method
ID: D1 Name: Exactitud Semantic: Concierne la correctitud y la precisión con que los datos del mundo real son representados en un sistema de información Suggested by: {BR3}	ID: D1F1 Name: Exactitud Sintáctica Semantic: Indica qué tan libre de errores sintácticos están los datos Represents: {BR3}	ID: M1 Name: FormatoFecha Purpose: Mide si el formato de un dato que representa una fecha es correcto, relativo a un formato especificado. No se mide para datos faltantes. Influenced by: {BR3} Granularity: celda Result domain: {0,1}	ID: ME1 Name: verificarFormatoFecha Uses: {BR3} Input data types: String, String Output data types: Boolean Algorithm: verificarFormatoFecha(dato, formato)	ID: AME24 Applied to: movie_ratings.csv, “release_date”
ID: D2 Name: Completitud Semantic: Concierne la proporción del mundo real que está representado en el SI Suggested by: {BR5, DL3}	ID: D2F1 Name: Densidad Semantic: Indica qué cantidad de registros con datos faltantes hay Represents: {DL3}	ID: M4 Name: RatioNoVacios Purpose: Mide el grado de valores no faltantes en una columna. Lo que se considera faltante depende del método. Influenced by: {DL3} Granularity: columna Result domain: [0..1]	ID: ME4 Name: ratioNoNulos Uses: {DL3} Input data types: Object[] Output data types: Float Algorithm: ratioNoNulos(columna)	ID: AME25 Applied to: movie_ratings.csv, “title”
				ID: AME26 Applied to: movie_ratings.csv, “genres”
				ID: AME27 Applied to: movie_ratings.csv, “release_date”
				ID: AME28 Applied to: movie_ratings.csv, “vote_average”
				ID: AME29 Applied to: movie_ratings.csv, “vote_count”
				ID: AME30 Applied to: movie_ratings.csv, “production_countries”
	ID: D2F2 Name: Cobertura Semantic: Indica qué cantidad de datos de la realidad están representados en el sistema Represents: {BR5}	ID: M5 Name: PorcentajeCobertura Purpose: Mide el porcentaje de valores que se encuentran representados por algún registro del dataset, relativo a un referencial Influenced by: {BR5} Granularity: dataset Result domain: [0..1]	ID: ME6 Name: porcentajeCoberturaRef Uses: {BR5} Input data types: Object[], String[] Output data types: Float Algorithm: porcentajeCoberturaRef(columna, ref)	ID: AME31 Applied to: movie_ratings.csv
ID: D3 Name: Consistencia Semantic: Concierne la consistencia entre los registros del SI Suggested by: {BR1, BR4, DL3}	ID: D3F1 Name: Integridad de Dominio Semantic: Indica que los datos representados estén en el dominio, dada su semántica y el dominio de aplicación Represents: {BR1, BR4, DL3}	ID: M6 Name: DatoEnDominioString Semantic: Mide si un dato de tipo texto es posible en el dominio, dado un referencial Influenced by: {BR1} Granularity: celda Result domain: {0,1}	ID: ME7 Name: StatusEnDominio Semantic: Dado un dato y un referencial, verifica que el dato esté en el referencial Uses: {BR1} Input data types: String, String[] Output data types: Boolean Algorithm: verificarStatus(status, posiblesStatus)	ID: AME32 Applied to: movie_ratings.csv, “status”
		ID: M7 Name: DatoEnRango Semantic: Mide si un dato numérico es positivo Influenced by: {BR4, DL3} Granularity: celda Result domain: {0,1}	ID: ME8 Name: DatoEnRango Semantic: Dado un dato numérico, verifica si se encuentra en un rango dado Uses: {BR4, DL3} Input data types: Float, Float, Float Output data types: Boolean Algorithm: verificarNumeroEnRango(dato)	ID: AME33 Applied to: movie_ratings.csv, “vote_average”
				ID: AME34 Applied to: movie_ratings.csv, “vote_count”
ID: D4 Name: Unicidad Semantic: Concierne el grado de repetición de los registros del SI Suggested by: {DL3}	ID: D4F1 Name: No-Duplicación Semantic: Mide el grado de filas que son únicas Arises from = {DL3}	ID: M9 Name: RatioClavesUnicas Semantic: Mide el grado de filas de un dataset que son únicas, según una fila identificadora Depends on = {DL3} Granularity: dataset Result domain: [0..1]	ID: ME10 Name: RatioClavesUnicas Uses: {DL3} Input data types: Object[][], Object[] Output data types: Float Algorithm: ratioNoDupFilas(dataset, columna)	ID: AME35 Applied to: movie_ratings.csv, “id”
ID: D5 Name: Frescura Semantic: Concierne el grado en el que la edad de los datos afecta su uso Suggested by: {SR2}	ID: D5F1 Name: Oportunidad Semantic: Mide qué tan oportunos son los datos a la hora de su lectura Represents: {SR2}	ID: M10 Name: OportunidadPuntaje Semantic: Calcula la diferencia en minutos entre la hora en la que se necesitan los datos y la hora en la que se actualizan Influenced by: {SR2} Granularity: dataset Result domain: [0..+inf)	ID: ME13 Name: Oportunidad8:45am Uses: {SR2} Input data types: Object[][] Output data types: Integer Algorithm: timelinessScore(dataset)	ID: AME36 Applied to: movie_ratings.csv
		ID: M11 Name: Edad Semantic: Calcula los minutos desde la última actualización de los datos Influenced by: {SR2} Granularity: dataset Result domain: [0..+inf)	ID: ME12 Name: EdadMinutos Uses: {SR2} Input data types: Object[][] Output data types: Integer Algorithm: staleness(dataset)	ID: AME37 Applied to: movie_ratings.csv

Figura 5.5: Modelo de calidad de datos basada en el contexto para movie_ratings.csv

DQ Dimension	DQ Factor	DQ Metric	DQ Method	Applied DQ Method
ID: D1 Name: Exactitud Semantic: Concierne la correctitud y la precisión con que los datos del mundo real son representados en un sistema de información Suggested by: {BR3}	ID: D1F1 Name: Exactitud Sintáctica Semantic: Indica qué tan libre de errores sintácticos están los datos Represents: {BR3}	ID: M1 Name: FormatoFecha Purpose: Mide si el formato de un dato que representa una fecha es correcto, relativo a un formato especificado. No se mide para datos faltantes. Influenced by: {BR3} Granularity: celda Result domain: {0,1}	ID: ME1 Name: verificarFormatoFecha Uses: {BR3} Input data types: String, String Output data types: Boolean Algorithm: verificarFormatoFecha(dato, formato)	ID: AME38 Applied to: TMDB_movie_dataset_v11.csv, “release_date”
ID: D2 Name: Completitud Semantic: Concierne la proporción del mundo real que está representado en el SI Suggested by: {OM1}	ID: D2F1 Name: Densidad Semantic: Indica qué cantidad de registros con datos faltantes hay Represents: {OM1}	ID: M4 Name: RatioNoVacios Purpose: Mide el grado de valores no faltantes en una columna. Lo que se considera faltante depende del método. Influenced by: {OM1} Granularity: columna Result domain: [0..1]	ID: ME4 Name: ratioNoNulos Uses: {OM1} Input data types: Object[] Output data types: Float Algorithm: ratioNoNulos(columna)	ID: AME39 Applied to: TMDB_movie_dataset_v11.csv, “title”
				ID: AME40 Applied to: TMDB_movie_dataset_v11.csv, “genres”
				ID: AME41 Applied to: TMDB_movie_dataset_v11.csv, “release_date”
				ID: AME42 Applied to: TMDB_movie_dataset_v11.csv, “vote_average”
				ID: AME43 Applied to: TMDB_movie_dataset_v11.csv, “vote_count”
				ID: AME44 Applied to: TMDB_movie_dataset_v11.csv, “production_countries”
ID: D3 Name: Consistencia Semantic: Concierne la consistencia entre los registros del SI Suggested by: {BR1}	ID: D3F1 Name: Integridad de Dominio Semantic: Indica que los datos representados estén en el dominio, dada su semántica y el dominio de aplicación Represents: {BR1}	ID: M6 Name: DatoEnDominioString Purpose: Mide si un dato de tipo texto es posible en el dominio, dado un referencial Influenced by: {BR1} Granularity: celda Result domain: {0,1}	ID: ME7 Name: StatusEnDominio Semantic: Dado un dato y un referencial, verifica que el dato esté en el referencial Uses: {BR1} Input data types: String, String[] Output data types: Boolean Algorithm: verificarStatus(status, posiblesStatus)	ID: AME45 Applied to: TMDB_movie_dataset_v11.csv, “status”
ID: D5 Name: Frescura Semantic: Concierne el grado en el que la edad de los datos afecta su uso Suggested by: {SR3}	ID: D5F1 Name: Oportunidad Semantic: Mide qué tan oportunos son los datos a la hora de su lectura Represents: {SR3}	ID: M10 Name: OportunidadPuntaje Purpose: Calcula la diferencia en minutos entre la hora en la que se necesitan los datos y la hora en la que se actualizan Influenced by: {DQR3} Granularity: dataset Result domain: [0..+inf)	ID: ME19 Name: Oportunidad8:30am Uses: {SR3} Input data types: Object[][] Output data types: Integer Algorithm: timelinessScore(dataset)	ID: AME46 Applied to: TMDB_movie_dataset_v11.csv
		ID: M11 Name: Edad Purpose: Calcula el tiempo desde la última actualización de los datos, en una unidad de tiempo dada Influenced by: {DQR3} Granularity: dataset Result domain: [0..+inf)	ID: ME12 Name: EdadMinutos Uses: {DQR6} Input data types: Object[][] Output data types: Integer Algorithm: staleness(dataset)	ID: AME47 Applied to: TMDB_movie_dataset_v11.csv
ID: D6 Name: Cantidad de Datos Semantic: Concierne la cantidad y el tamaño de los datos almacenados en el SI Suggested by: {}	ID: D5F1 Name: Volumen Semantic: Mide la cantidad de datos almacenados Represents: {}	ID: M12 Name: CantidadDeFilas Semantic: Calcula la cantidad de filas en dataset Influenced by: {} Granularity: dataset Result domain: [0..+inf)	ID: ME20 Name: countFilas Uses: {} Input data types: Object[][] Output data types: Integer Algorithm: countRows(dataset)	ID: AME48 Applied to: TMDB_movie_dataset_v11.csv
		ID: M13 Name: Tamaño Semantic: Calcula el tamaño del dataset en alguna unidad especificada Influenced by: {} Granularity: dataset Result domain: [0..+inf)	ID: ME21 Name: byteSize Uses: {} Input data types: Object[][] Output data types: Integer Algorithm: sizeInBytes(dataset)	ID: AME49 Applied to: TMDB_movie_dataset_v11.csv

Figura 5.6: Modelo de calidad de datos basada en el contexto para TMDB_movie_dataset_v11.csv

5.4. Implementación y resultados obtenidos

Para realizar una prueba de la medición de calidad de datos basada en el contexto, teniendo en cuenta los modelos de contexto y calidad propuestos, se realiza la implementación de las transformaciones de los *datasets* para el caso de uso del usuario *Movie Performance Specialist*, en el lenguaje *Python* (Python Software Foundation, 2025).

Para el almacenamiento de los metadatos del DL, se instancia una base de datos de grafos, utilizando la tecnología *Neo4j* (Neo4j, Inc., 2025).

Neo4j es un sistema de gestión de bases de datos (*DBMS*, por sus siglas en inglés) que utiliza el lenguaje de consulta *Cypher*. Como se especifica en la arquitectura, los metadatos de los datos y procesos se generarán al mismo tiempo que se ejecuten los procesos.

5.4.1. Preparación de metadatos de calidad de datos y contexto

Antes de consumir los *datasets* en la *Landing Zone*, se implementan en la base de datos de *Neo4j* los metadatos asociados a los modelos de contexto y de calidad de datos, que serán aplicados a los datos procesados. Para los modelos de contexto, se crea un nodo por cada modelo, y un nodo por cada instancia de componentes de contexto. En la Figura 5.7 se muestra la implementación del modelo de contexto *ctxm2*, que corresponde al *dataset MoviesML.csv* de la *Refined Zone*.

Como aún no se cuenta con los metadatos de datos y procesos, los nodos del tipo *Data Lineage* no contarán con relaciones hacia estos metadatos. Sin embargo, sí se contará con una versión *"placeholder"* del campo *Lineage*. Los procesos que generen los metadatos de datos y procesos también serán responsables de generar el *JSON* de *Lineage* y las relaciones entre los nodos del tipo *Data Lineage* y los metadatos de *Dataset* y *Process*.

En el caso de los modelos de CD, se genera un nodo para el *DQ Model*, y se generan nodos para todas las *DQ Dimension*, *DQ Factor*, *DQ Metric* y *DQ Method*, y *Applied DQ Method*. Además, se incluyen relaciones con los nodos del modelo de contexto.

Como aún no se cuenta con los metadatos de datos y procesos, no se crean las relaciones entre *Applied DQ Method* y el esquema sobre el que se aplica el modelo de CD. Por lo tanto, los procesos que generen los metadatos de datos y procesos, también serán responsables por crear las relaciones entre los nodos del tipo *Applied DQ Method* y sus respectivos esquemas.

En la Figura 5.8 se muestra la implementación del modelo de CD (*dqm1*) correspondiente al *dataset Movies_ML.csv* (Figura 5.4). En esta figura se desarrolla la dimensión Exactitud (por simplicidad, se omiten los factores, métricas, métodos y métodos aplicados asociados a las otras dimensiones).

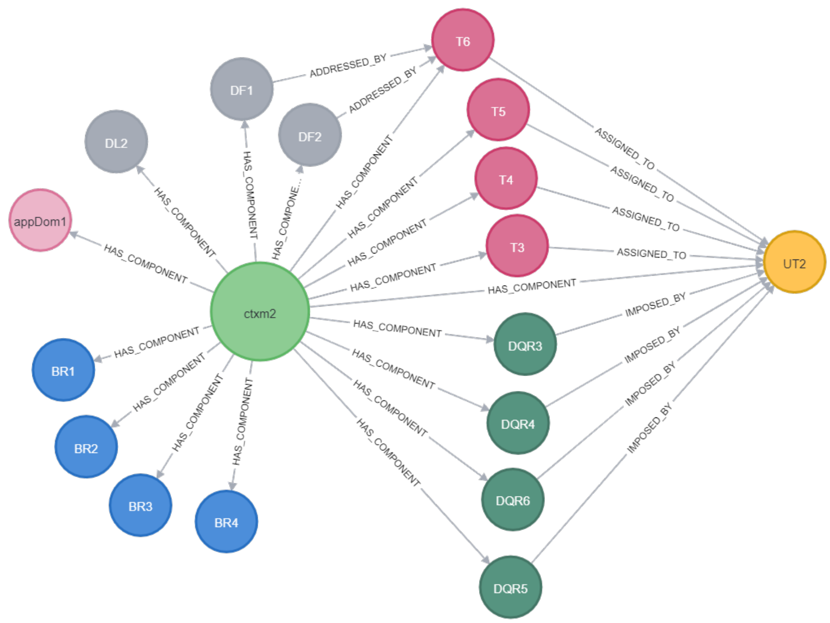

Figura 5.7: Implementación del modelo de contexto *ctxm2* en *Neo4j*

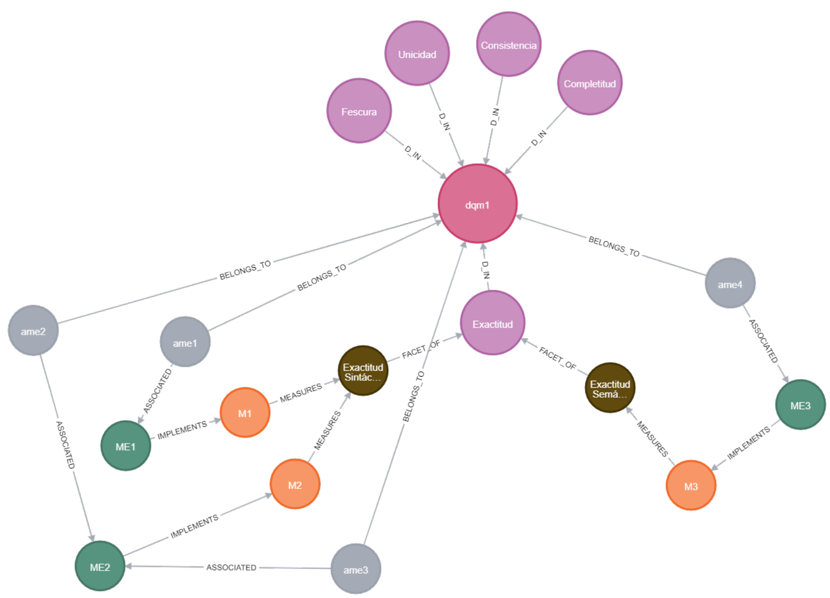

Figura 5.8: Implementación del modelo de calidad *dqm1* en *Neo4j*, expandiendo la dimensión Exactitud

Una vez implementados todos los modelos de contexto y de CD, es posible realizar consultas sobre el grafo de metadatos para estudiar las relaciones entre ellos. En la Figura 5.9 se muestra el grafo con las relaciones entre el modelo de contexto *ctxm2* y el modelo de CD *dqm1*. Por claridad visual, nuevamente solo se expande la dimensión Exactitud, hasta sus factores. Observar que existe la relación *CONTEXTUALIZES* entre *ctxm2* y *dqm1*. Tal como se indica en el modelo de CD, la dimensión Exactitud es sugerida por los componentes *T4*, *BR3* y *DQR4*. El factor Exactitud Semántica representa a la tarea *T4*, y el factor Exactitud Sintáctica representa a *BR3* y *DQR4*.

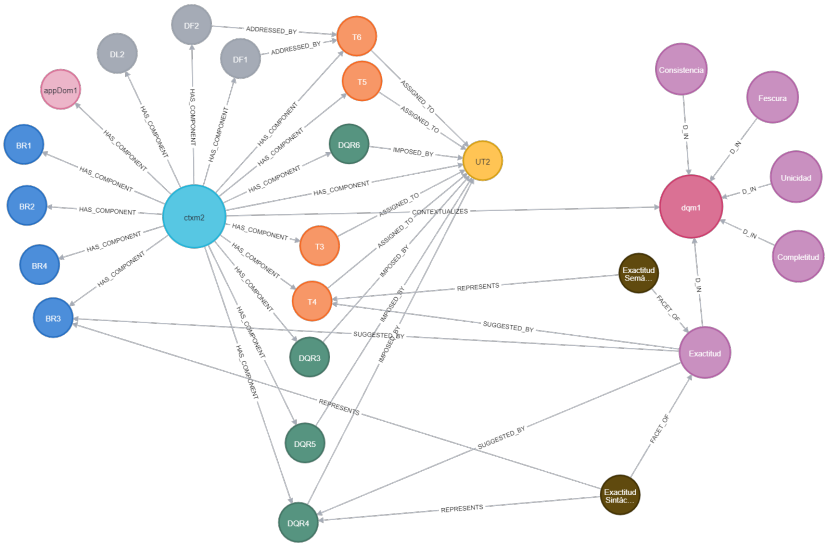

Figura 5.9: Relaciones entre el modelo de contexto *ctxm2* y el modelo de calidad de datos *dqm1*

Adicionalmente, se pueden realizar consultas más específicas.

Ejemplo 1: para obtener todos los componentes de contexto que sugieren a la dimensión Completitud, es posible realizar la consulta:

```
MATCH (D2:Dimension {name: 'Completitud'})-[:SUGGESTED_BY]->(related)
RETURN D2, related;
```

Ejemplo 2: para obtener todos los modelos de contexto en los que participa el componente BR2, a qué *dataset* contextualizan esos modelos, y a qué zona pertenecen esos *datasets*, es posible realizar la consulta:

```
MATCH (cm:ContextModel)-[:HAS_COMPONENT]->(br:BusinessRule {id: "BR2"})
MATCH (cm)-[:CONTEXTUALIZES]->(ds)
MATCH (ds)-[:STORED_IN]->(z:Zone)
RETURN cm, ds, z, br
```

En las Figuras 5.10 y 5.11 se muestra el resultado de las consultas de los Ejemplos 1 y 2, respectivamente.

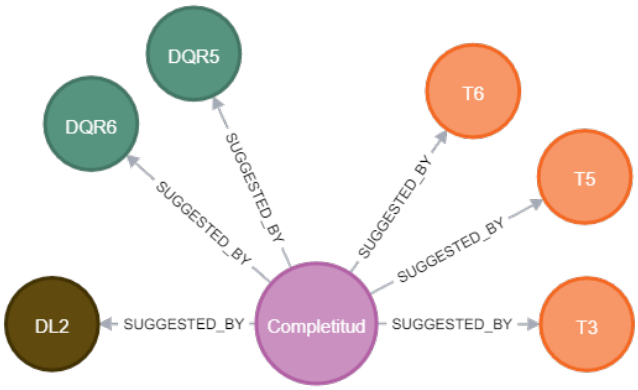

Figura 5.10: Resultado de la consulta del Ejemplo 1.

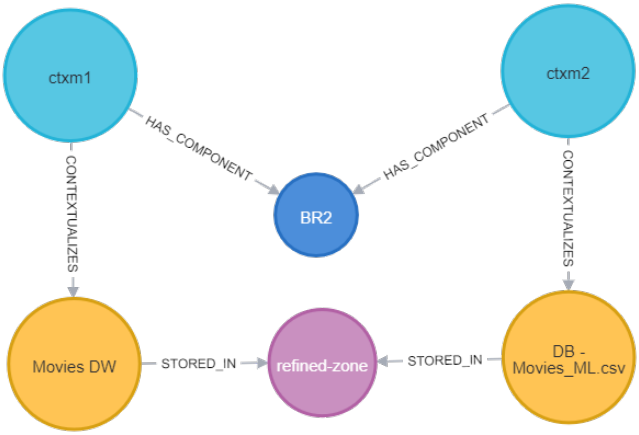

Figura 5.11: Resultado de la consulta del Ejemplo 2.

5.4.2. Ejecución de procesos del *Data Lake* y medidas de calidad

Para implementar las transformaciones llevadas a cabo en el DL, se utilizan los *frameworks* *Pandas* (Pandas, 2025) y *NumPy* (NumPy team, 2025) pa-

ra *Python*, los cuales permiten realizar operaciones eficientes de manipulación, limpieza y análisis de datos estructurados de manera flexible y escalable. Para simular las zonas del DL, simplemente se utilizan directorios con los nombres de las mismas en el *file system* local.

Antes de su ejecución, los procesos son configurados para conocer el modelo de CD que deben aplicar. Luego, las mediciones de calidad de datos se realizan una vez completada la transformación. Los procesos incluyen la lógica necesaria para vincular automáticamente los metadatos correspondientes al momento de ejecución, incluyendo la información de contexto, calidad, *datasets* y los propios procesos, dentro del grafo de metadatos almacenado en *Neo4j*. Esto se logra a través de una conexión a la base de datos, mediante la biblioteca oficial de *Python neo4j*. Además, después de realizar cada una de las transformaciones, los procesos generan los metadatos correspondientes a los *datasets* y procesos involucrados. En este momento, se genera el *JSON* correspondiente al *Data Lineage* del *dataset* generado.

Para este ejemplo, se realiza la implementación y se ejecutan las mediciones para la métrica *M4*, *RatioNoVacíos*, que participa en los tres modelos de calidad de datos definidos.

En las Figuras 5.12, 5.13 y 5.14 se presentan gráficas que muestran la medida de la métrica *RatioNoVacios* para los *datasets* *TMDB_movies_dataset_v11.csv*, *movie_ratings.csv* y *Movies_ML.csv*, respectivamente. Notar que solo se aplican los métodos sobre las columnas detalladas en los modelos de contexto.

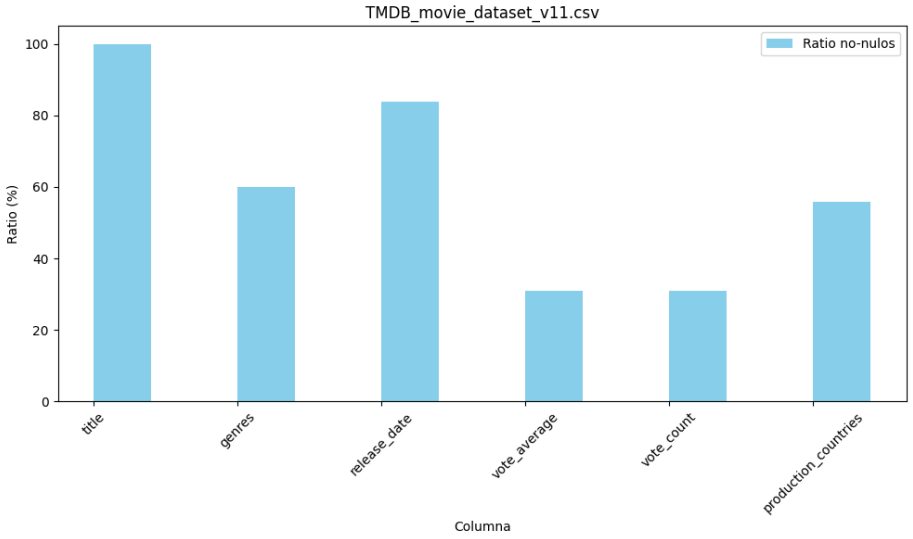

Figura 5.12: *RatioNoVacios* para *TMDB_movies_dataset_v11.csv*

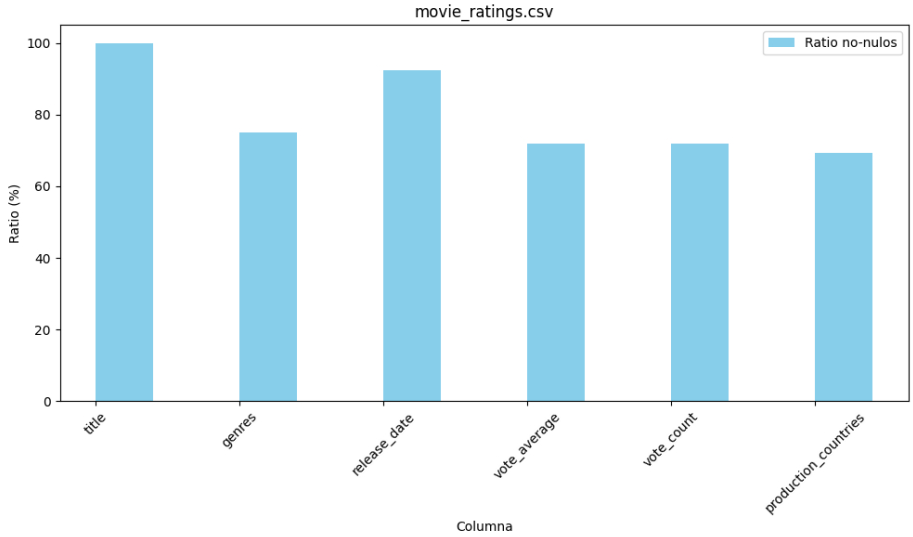

Figura 5.13: *RatioNoVacios* para *movie_ratings.csv*

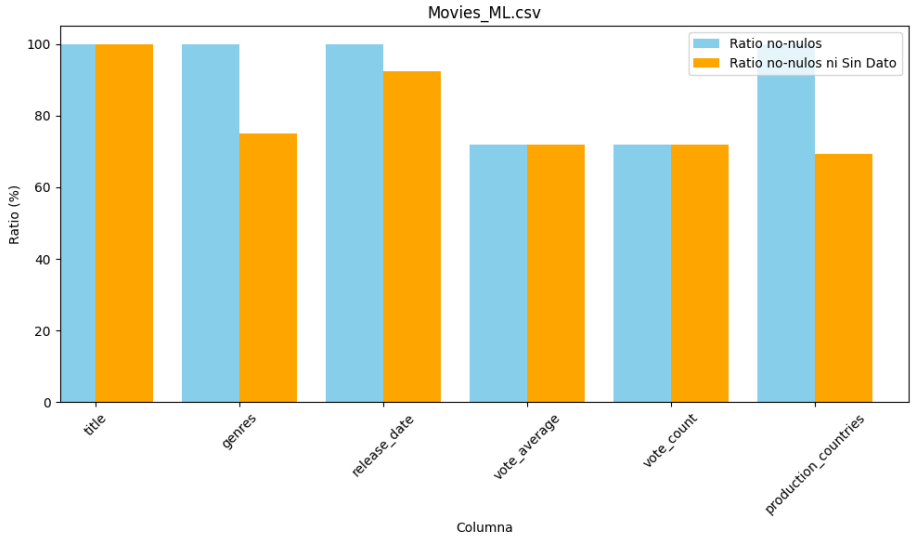

Figura 5.14: *RatioNoVacios* para *Movies_ML.csv*

Se observa que la cantidad de registros no-nulos mejora drásticamente entre *TMDB_movies_dataset_v11.csv* (*Landing Zone*) y *movie_ratings.csv* (*Trusted Zone*) en los campos *vote_count* y *vote_average*, lo que es esperable, dado que se incorporan datos de *IMDb*.

En el caso de *Movies_ML.csv*, se ejecuta la métrica aplicando dos métodos diferentes, uno que mide la proporción de valores nulos, y otro que mide la proporción de valores nulos o con un dato que represente el valor "Sin Datos". En este caso, se observa que, para los campos *genres*, *release_date* y *production_countries*, el 100 % de las columnas toman un valor no-nulo, pero sus ratios de valores no nulos ni "Sin Dato", coincide con el valor de datos no-nulo de las columnas correspondientes en el *dataset movie_ratings.csv*.

Finalmente, los resultados de las mediciones se almacenan en la base de datos de grafos, creando nodos del tipo *DQ Measure* asociados a sus respectivos *Applied DQ Method*, y vinculados a los metadatos de los campos cuya calidad es medida. En la Figura 5.15 se muestra la relación entre los metadatos del modelo de contexto *dqm1* y los metadatos que representan al *dataset Movies_ML.csv*. Esta relación está representada por las relaciones *APPLIED_TO* entre los nodos *Applied DQ Method* y los nodos *Column*. Por claridad, solo se muestra la dimensión Completitud, con el factor Densidad, su métrica *RatioNoVacios* (*M4*) y el método *RatioNoNulos* (*ME4*). Los círculos verdes claros, pequeños y con valores numéricos son los nodos del tipo *DQ Measure*, que almacenan las medidas.

Con estos metadatos, es posible realizar diferentes tipos de consultas.

Por un lado, se puede consultar por todas las métricas que hayan sido aplicadas y ejecutadas sobre una columna determinada. En la Figura 5.16 se muestra el resultado de consultar las medidas disponibles para la columna *production_countries* del *dataset Movies_ML.csv*, y los componentes de contexto del modelo *ctxm2* (representados en azul) que justifican su medición. Se observa que la medida es considerablemente peor si se evalúa según el método *ME5*.

Por otro lado, se puede utilizar el grafo de metadatos para evaluar la evolución de la calidad de los datos a lo largo del procesamiento de los *datasets*. En la Figura 5.17 se muestra el resultado de consultar los valores de las medidas relacionadas a la dimensión Densidad para la columna *vote_average*, que se encuentra presente en los tres *datasets* analizados. Como se observa en la figura, el método *ME4* (*ratioNoNulos*) es aplicado a la columna *vote_average* de los *datasets Movies_ML.csv* (*Refined Zone*), *movie_ratings.csv* (*Trusted Zone*) y *TMDB_movie_dataset_v11.csv* (Landing Zone), a través de los *Applied DQ Method ame8*, *ame28* y *ame42*, que pertenecen a los modelos de calidad de datos *dqm1*, *dqm2* y *dqm3*, respectivamente. Los modelos de calidad *dqm1*, *dqm2* y *dqm3* de definen considerando los modelos de contexto *ctxm2*, *ctxm3* y *ctxm4* respectivamente. Estos tres modelos de contexto contienen los componentes que sugieren el uso del método *ME4* en cada caso, siendo estos *DL2* y *DQR5* en el caso de *ctxm2*, *DL3* en el caso de *ctxm3*, y *OM1* en el caso de *ctxm4*.

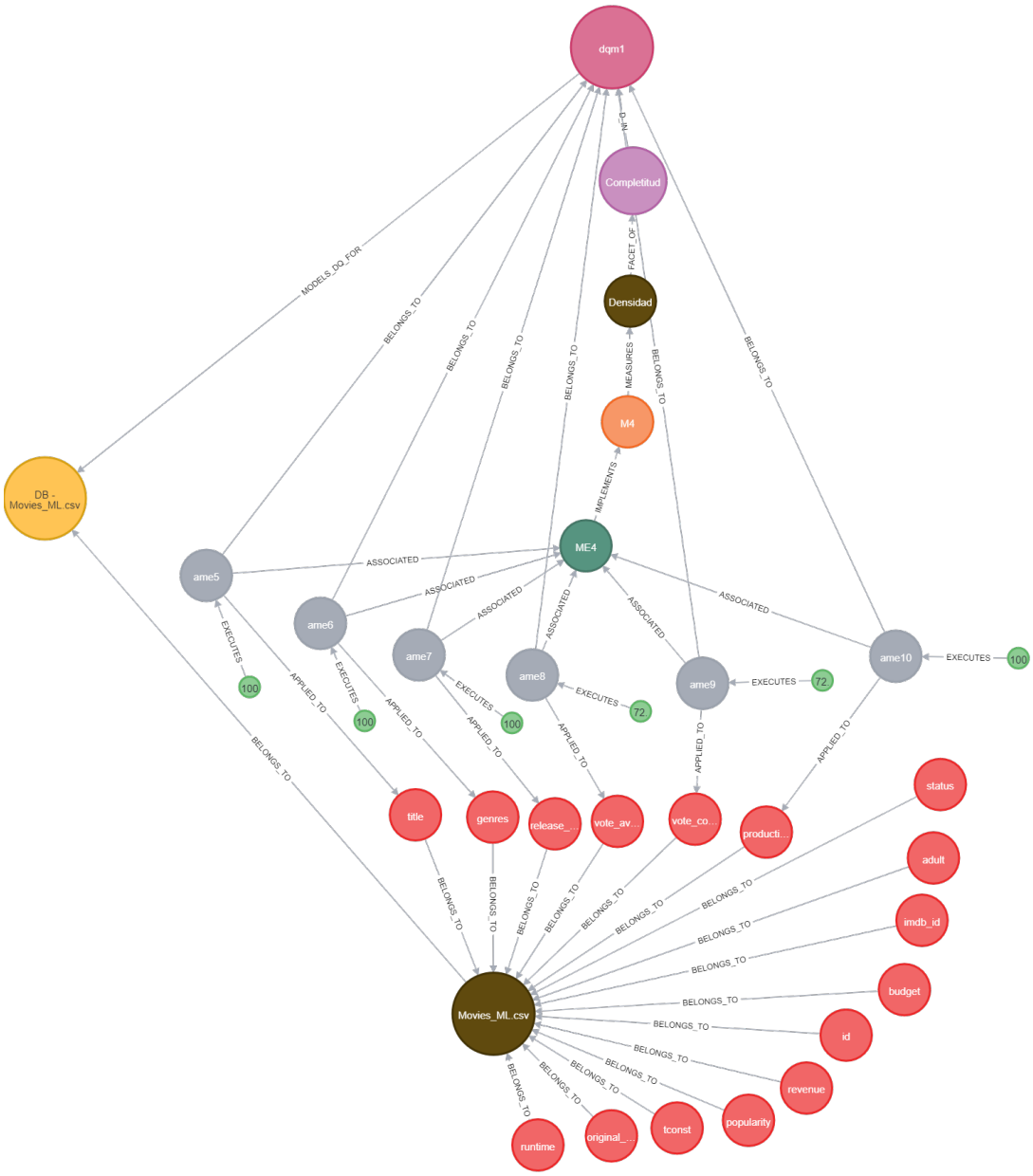

Figura 5.15: Relaciones entre los metadatos de *Movies_ML.csv* y *dqm1*

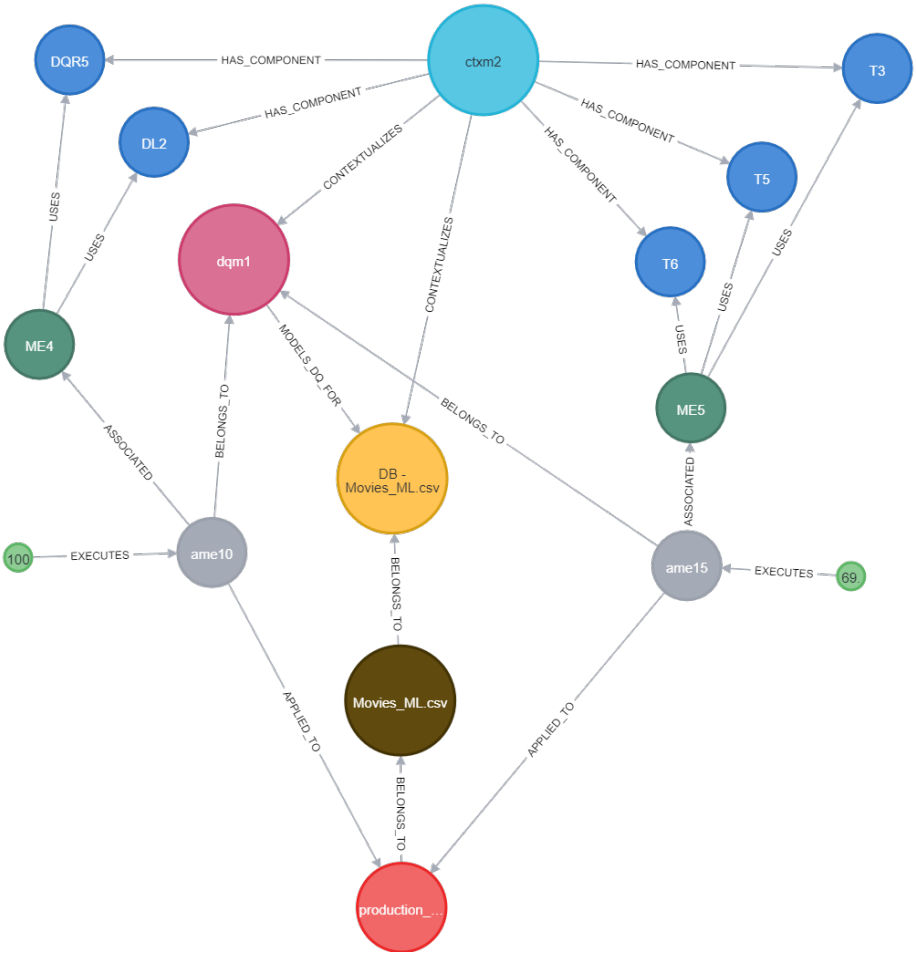

Figura 5.16: Metadatos de medidas relacionadas a la columna *production_countries*.

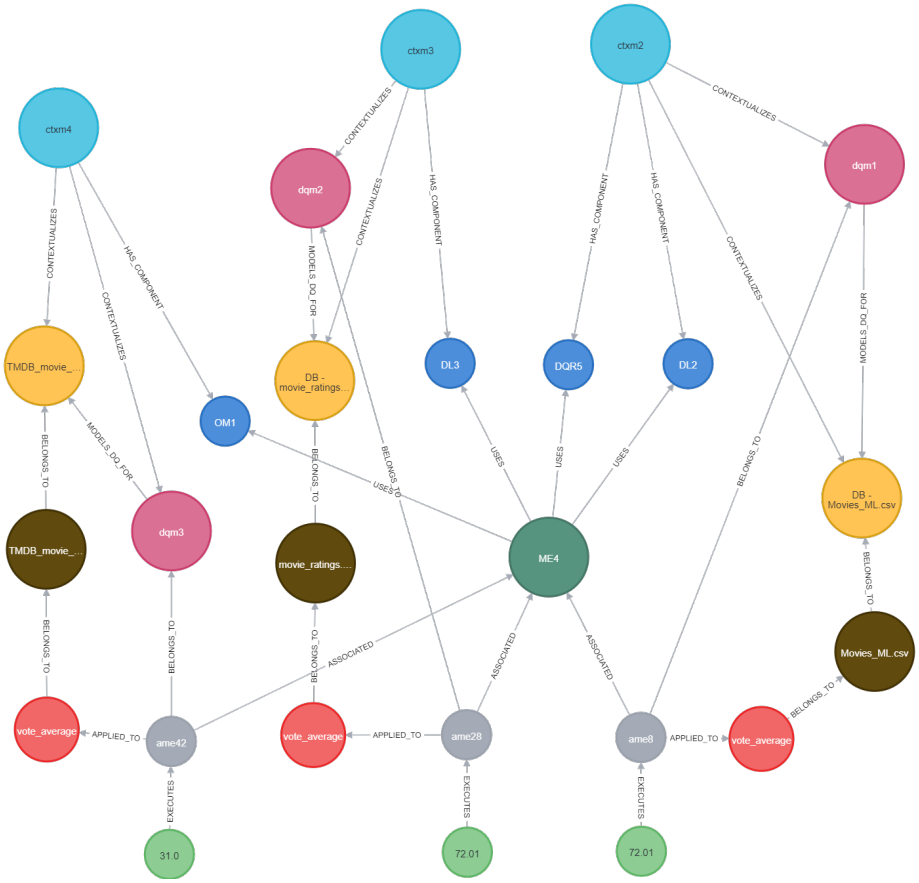

Figura 5.17: Evolución de la Densidad para la columna *vote_average*.

Observando los resultados obtenidos, se nota que la medida mejora drásticamente entre los valores dados por *ame42* y *ame28*, aumentando de 31.0 % a 72.01 %. En base a esto, se evalúa que el proceso de *JOIN* entre los *datasets* que se utilizan como *input* para la transformación resulta en una mejora de la calidad de los datos. Sin embargo, luego la medida retiene el mismo valor al ser evaluada por *ame8*, por lo que se concluye que los procesos intermedios no ayudan a la mejora de la calidad de los datos según esta métrica. Esto podría sugerir el desarrollo de una mejora de los procesos involucrados.

Capítulo 6

Conclusiones y trabajo futuro

En este capítulo se presentan las conclusiones, donde se realiza una reflexión sobre el trabajo realizado, y se evalúa el cumplimiento de los objetivos propuestos al inicio del proyecto. Además, se proponen posibles extensiones a este trabajo que, si bien no fueron implementadas por limitaciones de alcance de este proyecto, elevarían el nivel de la solución.

6.1. Conclusiones

En este proyecto, se definieron cinco objetivos principales. El primer objetivo planteaba el estudio de la bibliografía existente sobre arquitecturas de *Big Data*, particularmente profundizando en los *Data Lakes*. Para lograr esto, se realizó un análisis de la literatura existente en el área, en el cual se detallaron varias arquitecturas de *Big Data*, resaltando las ventajas de cada una y las diferencias entre las mismas.

El segundo objetivo proponía un estudio en profundidad de la arquitectura de *Big Data* genérica, definida en el proyecto de grado de (Cortés, 2024), el cual, a su vez, se utiliza en el proyecto del grupo Gema. Mediante el desarrollo del presente proyecto, se logró un nivel de entendimiento profundo de la arquitectura, y de la propuesta inicial de gestión de calidad de datos en la misma. En base a esto, se propusieron mejoras al proyecto de (Cortés, 2024), puntualmente al modelo de metadatos, representados en la *Governance Metadata Zone*.

Por otro lado, el tercer objetivo indicaba el estudio de la propuesta de gestión de calidad de datos basada en el contexto, que involucra la definición de modelos de contexto y modelos de calidad de datos basada en el contexto. Para incluir este enfoque en este proyecto, fue necesario estudiar la definición del metamodelo de contexto presentada en (Serra, 2024), comprendiendo la motivación y el diseño del mismo, incluyendo sus componentes y relaciones, y cómo se relacionan con los modelos de calidad de datos. Adicionalmente, se extendió la

propuesta, identificando un nuevo componente de contexto, denominado "*Data Lineage*", así como también todas las relaciones que involucran a dicho componente. Estas relaciones surgieron en base a las necesidades que se presentaron al buscar aplicar la propuesta de (Serra, 2024) a un sistema de *Big Data*, como lo son el relevamiento de medidas de calidad de datos a partir del linaje de los datos, y la propagación de los componentes de contexto entre modelos de *datasets* de distintas zonas.

Por último, el cuarto y quinto objetivo planteaban la propuesta e implementación de una extensión de la arquitectura genérica de *Big Data*, con el fin de incorporar modelos de contexto como parte de la gestión de la calidad de los datos. Para lograr esto, se realizó un análisis profundo de los dos trabajos previos, que eran el punto de partida, a partir del cual se logró presentar una propuesta concreta, diseñando un modelo de metadatos que satisficiera todas las necesidades identificadas. Con este diseño, se logró conceder a la arquitectura la capacidad de considerar el contexto como parte del proceso de medición de la calidad de los datos. La viabilidad de esta propuesta fue demostrada a través de la implementación de un caso de estudio.

Se considera que la propuesta elaborada en este proyecto aporta un valor positivo, tanto al marco conceptual de gestión de calidad de datos como a su aplicación práctica en entornos de *Big Data*. La incorporación de modelos de contexto a la gestión de calidad de datos presenta una ventaja considerable, ya que permite definir modelos de calidad de datos que se ajustan más precisamente a las necesidades de los usuarios. De esta forma, se reduce el riesgo de omitir métricas de calidad que podrían ser fundamentales para los mismos. Además, se limita la cantidad de mediciones irrelevantes que se ejecutan sobre los datos, lo que cobra especial importancia en *Big Data*, donde el diseño y ejecución de mediciones de calidad de datos supone costos más altos que en sistemas más reducidos.

6.2. Trabajo futuro

A pesar de que este trabajo demostró la viabilidad de la propuesta de gestión de calidad de datos basada en contexto en la arquitectura de *Big Data*, y se cumplieron todos los objetivos propuestos, se identificaron algunas mejoras a desarrollar en trabajos futuros.

- **Interfaz gráfica para la creación y modificación de los modelos en la base de datos de grafos.** La implementación de los modelos de contexto y de calidad de datos en la base de datos de grafos resulta ser una tarea tediosa, dado que con las herramientas actuales se depende de la generación de *scripts* en el lenguaje de consulta de la base utilizada (*Cypher* para *Neo4j*, en el caso de estudio presentado en el Capítulo 5, Experimentación), o bien de la ejecución manual de sentencias en el *DBMS*. Esta limitación hace que el proceso de definición de los metadatos de calidad y contexto sea propenso a errores, especialmente cuando se definen modelos

de mayor tamaño. Por lo tanto, sería beneficiosa la implementación de una herramienta gráfica que permita la gestión de los modelos, tanto de calidad de datos como de contexto, en la base de datos de grafos, con una interfaz de usuario clara.

- **Herramientas para la evaluación de la calidad de los datos a partir de las mediciones obtenidas.** Se podrían incorporar elementos a la propuesta que asistan en las fases del proceso de gestión de calidad de datos posteriores a la medición, como es la evaluación de la calidad de datos. Por ejemplo, se podría implementar la generación de alertas automáticas en los procesos de medición, que sean configurables por los *Data Quality Expert* y los notifiquen cuando ciertas medidas de calidad de datos estén por debajo o por encima de un umbral dado.

- **Mejora de procesos a partir de las medidas de calidad de datos obtenidas.** Como parte del caso de estudio, se presentó un ejemplo de la evolución de la calidad de los datos para una columna, según una métrica y método específico. Este tipo de análisis podría extenderse a un análisis más amplio y sistemático, que permitiera identificar tendencias o anomalías en la calidad de los datos, a medida que avanzan por el DL, con el fin de identificar procesos que empeoran la calidad de los datos (o que presentan una oportunidad para mejorarla pero no lo hacen), y en consecuencia proponer acciones correctivas o preventivas orientadas a la mejora continua de estos procesos.

Referencias

Amazon. (2024a). *Amazon redshift.* (https://aws.amazon.com/es/redshift/. Accedido: 2024-05-24)

Amazon. (2024b). *Amazon s3.* (https://aws.amazon.com/es/s3/. Accedido: 2024-05-25)

Batini, C., y Scannapieco, M. (2016). *Data and information quality: Dimensions, principles and techniques* (1st ed.). Springer Publishing Company, Incorporated.

Castro, F. (2025). *Caso de estudio para data lakehouse* (Módulo de Taller). Facultad de Ingeniería, Universidad de la República.

Castro, F., Gómez, P., y López, M. (2025). *Evaluación de exactitud sobre data lakes considerando el contexto* (Tesis de grado). Facultad de Ingeniería, Universidad de la República. (https://www.colibri.udelar.edu.uy/jspui/handle/20.500.12008/50091)

Cortés, C. (2024). *Gestión de calidad de datos en arquitecturas de big data* (Tesis de grado). Facultad de Ingeniería, Universidad de la República. (https://www.colibri.udelar.edu.uy/jspui/handle/20.500.12008/45647)

Fentaw, A. E. (2014). *Data vault modelling: An introductory guide* (Bachelor's thesis). Helsinki Metropolia University of Applied Sciences. (https://www.theseus.fi/bitstream/handle/10024/74895/thesis.pdf?sequence=1&isAllowed=y)

Google. (2024a). *Google bigquery.* (https://cloud.google.com/bigquery?hl=es-419. Accedido: 2024-05-25)

Google. (2024b). *Google cloud storage.* (https://cloud.google.com/products/storage?hl=es_419. Accedido: 2024-05-25)

Grupo Gema, InCo. (2023). *Calidad de datos en la preparación para el análisis de big data. proyecto csic.*

Harby, A. A., y Zulkernine, F. (2022). From data warehouse to lakehouse: A comparative review. En *2022 ieee international conference on big data (big data)* (p. 389-395). doi: 10.1109/BigData55660.2022.10020719

Hlupić, T., Oreščanin, D., Ružak, D., y Baranović, M. (2022). An overview of current data lake architecture models. En *2022 45th jubilee international convention on information, communication and electronic technology (mipro)* (p. 1082-1087). doi: 10.23919/MIPRO55190.2022.9803717

IBM. (2024a). 6 pillars of data quality and how to improve your data. *IBM Blog*. (https://www.ibm.com/blog/6-pillars-of-data-quality-and

-how-to-improve-your-data/. Accedido: 2024-09-16)
IBM. (2024b). *Ibm cloud object storage.* (https://www.ibm.com/products/cloud-object-storage. Accedido: 2024-05-25)
IBM. (2024c). *Ibm db2 warehouse.* (https://www.ibm.com/data-warehouse. Accedido: 2024-05-25)
IMDb. (s.f.). *IMDb.* (https://www.imdb.com/. Accedido: 2025-07-18)
Inmon, B. (2016). *Data lake architecture: Designing the data lake and avoiding the garbage dump.* Bradley Beach, NJ: Technics Publications.
Inmon, B., Levins, M., y Srivastava, R. (2021). *Building the data lakehouse.* Technics Publications.
Inmon, W. (2001). *Data mart does not equal data warehouse.* DM-Review.com. Descargado de https://web.archive.org/web/20110420134556/http://csis.bits-pilani.ac.in/faculty/goel/Data%20Warehousing/Articles/Data%20Marts/dataWarehouse_com%20%20Article_DM%20VS%20DW.htm (Accedido: 2024-05-30)
Inmon, W. H. (2002). *Building the data warehouse.* United States: John Wiley & Sons, Inc.
Kaggle. (s.f.). *Kaggle.* https://www.kaggle.com/. (Accedido: 2025-07-18)
Laney, D. (2001). *3D data management: Controlling data volume, velocity, and variety* (Inf. Téc.). META Group.
Marotta, A., y Serra, F. (2025). *Curso: Calidad de datos e información.* (Facultad de Ingeniería, Universidad de la República. https://eva.fing.edu.uy/course/view.php?id=1073)
Microsoft. (2024a). *Microsoft azure data lake storage.* (https://azure.microsoft.com/en-us/products/storage/data-lake-storage/. Accedido: 2024-05-25)
Microsoft. (2024b). *Microsoft azure synapse analytics.* (https://azure.microsoft.com/es-es/products/synapse-analytics/. Accedido: 2024-05-25)
Neo4j, Inc. (2025). *Neo4j graph database platform.* (https://neo4j.com/. Accedido: 2025-07-18)
NumPy team. (2025). *Numpy.* (https://numpy.org/. Accedido: 2025-07-18)
Pandas. (2025). *pandas: Python data analysis library.* (https://pandas.pydata.org/. Accedido: 2025-07-18)
Pentaho. (2025). *Pentaho data integration.* (https://pentaho.com/products/pentaho-data-integration/. Accedido: 2025-06-02)
Python Software Foundation. (2025). *Python programming language.* (https://www.python.org/. Accedido: 2025-07-18)
Ramchand, S., y Mahmood, T. (2022). Big data architectures for data lakes: A systematic literature review. En *2022 ieee 46th annual computers, software, and applications conference (compsac)* (p. 1141-1146). doi: 10.1109/COMPSAC54236.2022.00179
Schroeck, M., Shockley, R., Smart, J., Romero Morales, D., y Tufano, P. (2012). Analytics: the real-world use of big data: How innovative enterprises extract value from uncertain data, executive report.

Serra, F. (2024). *Context-aware data quality management* (Tesis Doctoral no publicada). Programa de Desarrollo de las Ciencias Básicas (PEDECIBA). Universidad de la Republica, Uruguay. Universidad de Tours, Francia. (https://www.colibri.udelar.edu.uy/jspui/handle/20.500.12008/47343)

Serra, F., Peralta, V., Marotta, A., y Marcel, P. (2022). Modeling context for data quality management. En J. Ralyté, S. Chakravarthy, M. Mohania, M. A. Jeusfeld, y K. Karlapalem (Eds.), *Conceptual modeling* (pp. 325–335). Cham: Springer International Publishing.

Sharma, B. (2018). *Architecting data lakes: Data management architectures for advanced business use cases*. O'Reilly Media, Inc.

Strong, D. M., Lee, Y. W., y Wang, R. Y. (1997). Data quality in context. *Commun. ACM*, *40*(5). doi: 10.1145/253769.253804

Yessad, L., y Labiod, A. (2016). Comparative study of data warehouses modeling approaches: Inmon, kimball and data vault (p.95-99).. doi: 10.1109/ICSRS.2016.7815845

Zaharia, M. A., Ghodsi, A., Xin, R., y Armbrust, M. (2021). Lakehouse: A new generation of open platforms that unify data warehousing and advanced analytics. En *Conference on innovative data systems research*. (https://api.semanticscholar.org/CorpusID:229576171)

Anexo A

Anexo 1

En este anexo se presentan los archivos *JSON* que representan los linajes de los datos correspondientes a los modelos de contexto presentados en el caso de estudio del capítulo 5.

Movies_ML_lineage.json

```
{
  "sources": [
        {
          "name": "kaggle"
        },
        {
          "name": "IMDb"
        }
  ],
  "datasets": [
    {
      "name": "TMDB_movie_dataset_v11.csv",
      "id": "ds1",
            "zone": "RAW",
            "source": "kaggle",
            "ingestion_date": "2024-11-08"
    },
    {
      "name": "title.ratings.tsv",
      "id": "ds4",
            "zone": "RAW",
            "source": "IMDb",
            "ingestion_date": "2024-11-08"
    },
          {
```

```
      "name": "TMDB_Movies_clean.csv",
      "id": "ds5",
            "zone": "TRUSTED"
    },
          {
      "name": "movie_ratings.csv",
      "id": "ds6",
            "zone": "TRUSTED"
    },
          {
            "name": "Movies_ML.csv",
            "id": "ds16",
            "zone": "REFINED"
          }
  ],
  "processes": [
    {
      "name": "tr:CleanTMDB",
      "id": "p1",
      "input": ["TMDB_movie_dataset_v11.csv"],
      "output": ["TMDB_Movies_clean.csv"],
      "description": "Se seleccionan los atributos
         relevantes del dataset TMDB Movie Dataset y
         se eliminan filas duplicadas."
    },
    {
      "name": "tr:MergeTMDBandRatings",
      "id": "p2",
      "input": ["title.ratings.tsv", "
         TMDB_Movies_clean.csv"],
      "output": ["movie_ratings.csv"],
      "description": "Realiza un merge entre
         TMDB_Movies_clean.csv y title.ratings.tsv,
         matcheando la clave imdb_id del dataset de
         TMDB con la clave tconst del dataset de IMDb.
          Se sustituyen los valores en las columnas
         vote_average y vote_count por los valores en
         averageRating y numVotes."
    },
    {
      "name": "tr:SanitizeMovies",
      "id": "p6",
      "input": ["movie_ratings.csv"],
      "output": ["Movies_ML.csv"],
      "description": "Se sanitizan los valores nulos
         de genre, release_date y production_companies
```

```
          , y se traslada a la zona Refined."
    }
  ]
}
```

DW_lineage.json

```
{
  "sources": [
        {
          "name": "kaggle"
        }
  ],
  "datasets": [
    {
      "name": "TMDB_movie_dataset_v11.csv",
      "id": "ds1",
          "zone": "RAW",
          "source": "kaggle",
          "ingestion_date": "2024-11-08"
    },
    {
      "name": "title.crew.tsv",
      "id": "ds2",
          "zone": "RAW",
          "source": "kaggle",
          "ingestion_date": "2024-11-08"
    },
    {
      "name": "name_basics.tsv",
      "id": "ds3",
          "zone": "RAW",
          "source": "kaggle",
          "ingestion_date": "2024-11-08"
    },
    {
      "name": "title.ratings.tsv",
      "id": "ds4",
          "zone": "RAW",
          "source": "kaggle",
          "ingestion_date": "2024-11-08"
    },
        {
      "name": "TMDB_Movies_Clean.csv",
      "id": "ds5",
```

```
            "zone": "TRUSTED"
    },
        {
      "name": "movie_ratings.csv",
      "id": "ds6",
            "zone": "TRUSTED"
    },
        {
      "name": "movies_3.csv",
      "id": "ds7",
            "zone": "TRUSTED"
    },
        {
      "name": "MoviesDataset.csv",
      "id": "ds8",
            "zone": "TRUSTED"
    },
        {
    "name": "dim_genero.csv",
    "id": "ds9",
    "zone": "REFINED"
    },
        {
          "name": "dim_director.csv",
          "id": "ds10",
          "zone": "REFINED"
        },
        {
          "name": "dim_lenguaje.csv",
          "id": "ds11",
          "zone": "REFINED"
        },
        {
          "name": "dim_pelicula.csv",
          "id": "ds12",
          "zone": "REFINED"
        },
        {
          "name": "dim_pais.csv",
          "id": "ds13",
          "zone": "REFINED"
        },
        {
          "name": "dim_fecha.csv",
          "id": "ds14",
          "zone": "REFINED"
```

```
        },
        {
          "name": "fact_table.csv",
          "id": "ds15",
          "zone": "REFINED"
        }
  ],
  "processes": [
    {
      "name": "tr:CleanTMDB",
      "id": "p1",
      "input": ["TMDB_movie_dataset_v11.csv"],
      "output": ["TMDB_Movies_clean.csv"],
      "description": "Se seleccionan los atributos
         relevantes del dataset TMDB Movie Dataset y
         se eliminan filas duplicadas."
    },
    {
      "name": "tr:MergeTMDBandRatings",
      "id": "p2",
      "input": ["title.ratings.tsv", "
         TMDB_Movies_clean.csv"],
      "output": ["movie_ratings.csv"],
      "description": "Realiza un merge entre
         TMDB_Movies_clean.csv y title.ratings.tsv,
         matcheando la clave imdb_id del dataset de
         TMDB con la clave tconst del dataset de IMDb.
          Se sustituyen los valores en las columnas
         vote_average y vote_count por los valores en
         averageRating y numVotes."
    },
    {
      "name": "tr:MergeCrew",
      "id": "p3",
      "input": ["movie_ratings.csv", "title.crew.tsv"]
         ,
      "output": ["movies_3.csv"],
      "description": "Se realiza un LEFT JOIN entre
         movie_ratings.csv y title.crew.tsv utilizando
          imdb_id y tconst respectivamente."
    },
    {
      "name": "tr:MergeNames",
      "id": "p4",
      "input": ["movies_3.csv", "name.basics.tsv"],
      "output": ["MoviesDataset.csv"],
```

```
      "description": "Se separan en multiples tuplas las peliculas que tengan una lista de directores en el atributo directors. Se dropean tconst y writers."
    },
    {
      "name": "tr:CreateWarehouse",
      "id": "p5",
      "input": ["MoviesDataset.csv"],
      "output": ["dim_genero.csv", "dim_lenguaje.csv", "dim_pelicula.csv", "dim_pais.csv", "dim_fecha.csv", "dim director.csv", "fact_table.csv"],
      "description": "Se separan los datos en tablas de dimension, generando identificadores para cada uno. Se crea la tabla de hechos."
    }
  ]
}
```

movie_ratings_lineage.json

```
    {
  "sources": [
        {
          "name": "kaggle"
        },
        {
          "name": "IMDb"
        }
  ],
  "datasets": [
    {
      "name": "TMDB_movie_dataset_v11.csv",
      "id": "ds1",
            "zone": "RAW",
            "source": "kaggle",
            "ingestion_date": "2024-11-08"
    },
    {
      "name": "title.ratings.tsv",
      "id": "ds4",
            "zone": "RAW",
            "source": "IMDb",
            "ingestion_date": "2024-11-08"
```

```
    },
          {
      "name": "TMDB_Movies_clean.csv",
      "id": "ds5",
            "zone": "TRUSTED"
    },
          {
      "name": "movie_ratings.csv",
      "id": "ds6",
            "zone": "TRUSTED"
    }
  ],
  "processes": [
    {
      "name": "tr:CleanTMDB",
      "id": "p1",
      "input": ["TMDB_movie_dataset_v11.csv"],
      "output": ["TMDB_Movies_clean.csv"],
      "description": "Se seleccionan los atributos
         relevantes del dataset TMDB Movie Dataset y
         se eliminan filas duplicadas."
    },
    {
      "name": "tr:MergeTMDBandRatings",
      "id": "p2",
      "input": ["title.ratings.tsv", "
         TMDB_Movies_clean.csv"],
      "output": ["movie_ratings.csv"],
      "description": "Realiza un merge entre
         TMDB_Movies_clean.csv y title.ratings.tsv,
         matcheando la clave imdb_id del dataset de
         TMDB con la clave tconst del dataset de IMDb.
          Se sustituyen los valores en las columnas
         vote_average y vote_count por los valores en
         averageRating y numVotes."
    }
  ]
}
```