UAB
Universitat Autònoma de Barcelona

Dipòsit digital de documents de la UAB

This is the **published version** of the bachelor thesis:

Gomàriz Mora, Carla. *Integració del Mòdul de Qualitat de Dades al Data Hub Corporatiu mitjançant Snowflake*. Treball de Final de Grau (Universitat Autònoma de Barcelona), 2026 (Grau en Enginyeria de Dades)

This version is available at https://ddd.uab.cat/record/337322

under the terms of the BY-NC-ND license.

Integració del Mòdul de Qualitat de Dades al Data Hub Corporatiu mitjançant Snowflake

Carla Gomàriz Mora

Resum– El mòdul de qualitat de dades (DQA) desenvolupat per Cetaqua existia com a script Python aïllat, inaccessible des del Data Hub corporatiu basat en Snowflake. S'ha integrat el mòdul DQA al Data Hub corporatiu mitjançant una arquitectura genèrica basada en Stage, UDTF (User Defined Table Function) i Stored Procedures, permetent avaluar dades segons sis dimensions de qualitat (*Completeness*, *Consistency*, *Timeliness*, *Adjusted Timeliness*, *Uniqueness* i *Validity*) sobre dos casos d'ús reals: *Agua Suministrada* i *Agua Registrada Devengada*. La implementació aconsegueix una millora de rendiment del 98% respecte a la solució inicial (de 32 minuts a 28 segons per execució en *Agua Suministrada*), i els resultats es visualitzen mitjançant una aplicació Streamlit que inclou funcionalitats d'alertes per a l'equip de negoci. La solució és escalable i reutilitzable: nous casos d'ús s'incorporen únicament mitjançant configuració JSON, sense modificar el motor DQA. El projecte ha estat desenvolupat seguint metodologia Agile Scrum i es troba actualment en entorn de pre-producció.

Paraules clau– Qualitat de dades, Data Hub, Snowflake, UDTF, Stored Procedures, Arquitectura Genèrica, Python, *Completeness*, *Consistency*, *Timeliness*, *Uniqueness*, *Validity*, Agile Scrum, Streamlit.

Abstract– The Data Quality Assessment (DQA) module developed by Cetaqua existed as an isolated Python script, inaccessible from the corporate Data Hub based on Snowflake. The DQA module has been integrated into the corporate Data Hub through a generic architecture based on Stage, UDTF (User Defined Table Function), and Stored Procedures, enabling data evaluation according to six quality dimensions (*Completeness*, *Consistency*, *Timeliness*, *Adjusted Timeliness*, *Uniqueness*, and *Validity*) on two real use cases: *Agua Suministrada* and *Agua Registrada Devengada*. The implementation achieves a 98% performance improvement over the initial solution (from 32 minutes to 28 seconds per execution for *Agua Suministrada*), and results are visualized through a Streamlit application that includes alert functionalities for the business team. The solution is scalable and reusable: new use cases are incorporated solely through JSON configuration, without modifying the DQA engine. The project has been developed following Agile Scrum methodology and is currently in pre-production environment.

Keywords– Data Quality, Data Hub, Snowflake, UDTF, Stored Procedures, Generic Architecture, Python, *Completeness*, *Consistency*, *Timeliness*, *Uniqueness*, *Validity*, Agile Scrum, Streamlit.

1 Introducció i Context del Treball

AQUEST projecte s'emmarca en una col·laboració entre Cetaqua, Centre Tecnològic de l'Aigua [5], i Synectic [12], empresa del grup Veolia [13], amb l'objectiu d'integrar un mòdul de qualitat de dades (DQA), ja existent, al Data Hub corporatiu basat en Snowflake [10]. La necessitat sorgeix de la importància crítica de garantir la qualitat de les dades en el sector de l'aigua, on decisions operacionals i estratègiques depenen directament de la fiabilitat de la informació disponible.

• E-mail de contacte: carla@gomariz.com

• Treball tutoritzat per: Cristina Fernández Córdoba (Departament d'Enginyeria de la Informació i de les Comunicacions)

• Curs 2025/26

1.1 Context i Motivació

Cetaqua és una fundació privada que impulsa la recerca, desenvolupament i innovació (R+D+i) per a la sostenibilitat del cicle integral de l'aigua. En el marc del projecte WATERSERVE [3], Cetaqua va desenvolupar un mòdul de

qualitat de dades implementat en Python que avaluava registres de fonts d'aigua segons cinc dimensions de qualitat. Aquest mòdul va ser creat com a solució local, executat manualment en entorns de desenvolupament de Cetaqua, sense integració amb plataformes cloud corporatives. El repte principal era que aquest mòdul DQA existia com a codi Python independent, sense connexió amb Snowflake, i calia executar-lo directament sobre les dades del Data Hub corporatiu de manera escalable i automatitzada. Tot i que el projecte WATERSERVE ja s'havia tancat, Synectic va identificar l'oportunitat de reutilitzar aquest mòdul dins del seu ecosistema de dades, transformant una solució local en una solució corporativa integrada. Aquesta iniciativa representa un benefici mutu: Synectic incorpora una eina de qualitat de dades al seu ecosistema corporatiu, mentre que Cetaqua adquireix coneixement pràctic sobre Snowflake, una tecnologia estratègica de cara al futur.

1.2 Les Sis Dimensions de Qualitat de Dades

La versió actual del mòdul DQA avalua conjunts de dades de sèries temporals segons sis dimensions de qualitat. Les primeres cinc dimensions (*Completeness*, *Consistency*, *Timeliness*, *Uniqueness* i *Validity*) van ser seleccionades en el projecte WATERSERVE per ser les més rellevants per al context del sector de l'aigua i les seves necessitats específiques identificades. Una sisena dimensió, *Adjusted Timeliness*, ha estat desenvolupada específicament per a aquest projecte amb l'objectiu d'ajustar el càlcul del *Timeliness* tradicional mitjançant una ponderació temporal, reduint l'impacte dels buits històrics i donant més rellevància a la disponibilitat recent de les dades.

- ***Completeness***: Avalua si hi ha dades per a tots els períodes esperats dins de la sèrie temporal. Un valor de 1.0 indica que no hi ha períodes mancants.

- ***Consistency***: Avalua si els valors estan dins dels rangs esperats i són coherents amb les regles de validació definides. Un valor de 1.0 indica que totes les dades compleixen les restriccions.

- ***Timeliness***: Avalua si les lectures arriben amb la periodicitat esperada. Mesura la regularitat temporal de les dades en relació amb la freqüència inferida o configurada.

- ***Adjusted Timeliness***: Una dimensió desenvolupada específicament per a aquest projecte que millora el *Timeliness* aplicant una ponderació exponencial per penalitzar més els buits recents que els antics. Per a l'equip de negoci és més crítica la qualitat de les dades recents que la de les dades històriques.

- ***Uniqueness***: Avalua si no hi ha duplicats en les dades. Un valor de 1.0 indica que totes les dades són úniques.

- ***Validity***: Avalua si els valors compleixen amb les regles de validació definides. Un valor de 1.0 indica que totes les dades són vàlides.

1.3 Estructura de Nivells d'Agregació i Casos d'Ús

El projecte implementa dos casos d'ús amb estructures de nivells d'agregació diferents. Els casos d'ús són *Agua Suministrada* i *Agua Registrada Devengada*, i cada un té diferents nivells d'agregació que representen la jerarquia de les dades: es comença amb la unitat mínima de mesura (actiu o punt de servei) i es van agregant progressivament a nivells superiors (activitats, explotacions). Aquesta estructura permet obtenir mètriques de qualitat a diferents nivells de granularitat, facilitant l'anàlisi tant a nivell detallat com a nivell estratègic.

Ara descriurem els dos casos d'ús implementats en el marc d'aquest projecte i els seus nivells d'agregació.

Agua Suministrada (aigua subministrada): Com es pot veure a la Taula 1, *Agua Suministrada* té una estructura de tres nivells d'agregació que van des de l'actiu, corresponent a un comptador d'aigua, fins a l'explotació, que representa el nivell més alt:

Nivell	Entitat	Descripció
1	Actiu	Unitat mínima de mesura
2	Activitat	Conjunt d'actius relacionats
3	Explotació	Agrupa diverses activitats

TAULA 1: ESTRUCTURA DE NIVELLS D'AGREGACIÓ PER A *Agua Suministrada*

Aquesta jerarquia permet obtenir mètriques de qualitat a cada nivell, facilitant tant l'anàlisi operacional (a nivell d'actiu) com l'anàlisi estratègica (a nivell d'explotació).

Agua Registrada Devengada (aigua registrada amb devengo): Aquest cas d'ús té una estructura de dos nivells d'agregació que van del punt de servei, associat a un contracte i des d'on es registren les dades d'aigua, fins a l'explotació:

Nivell	Entitat	Descripció
1	Punt de Servei	Unitat mínima de mesura
2	Explotació	Agrupa diversos punts de servei

TAULA 2: ESTRUCTURA DE NIVELLS D'AGREGACIÓ PER A *Agua Registrada Devengada*

En contrast, com es pot veure a la Taula 2, *Agua Registrada Devengada* presenta una estructura més simple amb només dos nivells d'agregació: punt de servei i explotació. Aquesta diferència en la jerarquia es deu a la naturalesa diferent dels dos casos d'ús: *Agua Suministrada* disposa d'un nivell intermedi (Activitat) que permet agrupar els actius abans d'arribar a l'explotació, mentre que *Agua Registrada Devengada* estableix una relació directa entre els punts de servei i les explotacions.

Una diferència important entre els dos casos d'ús és la periodicitat de les dades. Per a *Agua Suministrada*, cada actiu pot tenir una periodicitat diferent (horària, diària, mensual, etc.) que és inferida pel motor DQA a partir dels intervals temporals entre registres. En canvi, per a *Agua Registrada Devengada*, la periodicitat és sempre mensual (registres el dia 1 de cada mes), per la qual cosa no cal inferir-la.

A més, per al cas d'*Agua Registrada Devengada*, el cas d'ús ja disposava de dues variants de càlcul definides, sobre les quals s'aplica el motor DQA per obtenir les mètriques de qualitat per cada una:

- **Variant Lineal**: Utilitza la dada original, dividint el consum entre els dies del mes.
- **Variant Ajustada**: Té en compte el pes de la *Agua Suministrada* sobre aquell mes, amb una tendència proporcional a la suministrada. Aquesta variant permet comparar com els ajustos humans afecten les mètriques de qualitat.

1.4 Context Tècnic: Capacitats de Snowflake Rellevants

Snowflake ofereix múltiples mecanismes per executar lògica de dades dins la plataforma: queries SQL directes, Tasks programades, Snowpark, connectors externs, entre d'altres. Per a aquest projecte, que requereix integrar codi Python de manera escalable, les funcionalitats rellevants són:

Stored Procedure (SP): Un procediment emmagatzemat que permet encapsular i executar múltiples operacions com un únic procés, orquestrant fluxos de dades complexos.

UDTF (User Defined Table Function): Una funció personalitzada que retorna una taula com a resultat. A diferència de les funcions SQL estàndard, les UDTF permeten executar lògica complexa (com codi Python) dins Snowflake.

PARTITION BY: Una clàusula que, en el context de les UDTF, indica a Snowflake com dividir les dades d'entrada per alimentar la funció en paral·lel. Això permet distribuir el processament entre múltiples nodes de la infraestructura, aprofitant el caràcter distribuït de la plataforma. La combinació de UDTF + PARTITION BY és particularment important per a casos d'ús amb grans volums de dades.

Aquestes tres funcionalitats han permès arribar a una solució final que va més enllà de la simple integració del mòdul DQA dins de Snowflake. Al llarg del desenvolupament s'han anat identificant noves necessitats i limitacions tècniques que han requerit replantejar progressivament l'arquitectura de la solució. Per aquest motiu, el projecte ha tingut un marcat caràcter exploratori, en què les diferents decisions tècniques s'han anat prenent i validant de manera iterativa conjuntament amb l'equip de Synectic.

1.5 Objectius del Projecte

Els objectius específics del projecte són:

- **OE1**: Analitzar i comprendre el mòdul de qualitat de dades existent, estudiant el seu funcionament, estructura i les sis dimensions que avalua.
- **OE2**: Estudiar l'arquitectura del Data Hub corporatiu i les capacitats tècniques de Snowflake, identificant les opcions disponibles per a la integració del mòdul.
- **OE3**: Proposar i avaluar diferents alternatives d'integració en col·laboració amb l'equip tècnic de Synectic, seleccionant la solució més adequada.
- **OE4**: Implementar la solució escollida a Snowflake, adaptant el mòdul a les necessitats i característiques de la plataforma, i validar-ne el funcionament sobre les dades de *Agua Suministrada* i *Agua Registrada Devengada*.
- **OE5**: Desenvolupar una visualització dels resultats de qualitat de dades mitjançant Streamlit, que permeti representar les mètriques obtingudes per als diferents tipus de consum d'aigua i nivells d'agregació, facilitant la presentació dels resultats a l'equip de negoci. Aquest objectiu ha estat completat per a ambdós casos d'ús.
- **OE6**: Explorar possibles millores de rendiment aprofitant les capacitats natives de Snowflake, amb l'objectiu de reduir el cost computacional associat al processament de dades.

1.6 Organització del Document

La resta del document s'organitza de la següent manera: la Secció 2 presenta l'estat de l'art en qualitat de dades i treballs relacionats; la Secció 3 descriu la metodologia i el procés de desenvolupament; la Secció 4 presenta els resultats obtinguts i una discussió crítica; finalment, la Secció 5 exposa les conclusions i possibles extensions futures.

2 ESTAT DE L'ART

2.1 Qualitat de Dades: Conceptes i Estàndards

La qualitat de dades és un concepte multidimensional àmpliament estudiat en la literatura. L'estàndard ISO/IEC 25012:2008 [7] defineix un model de qualitat de dades amb quinze característiques organitzades en dues categories: qualitat inherent (independent del sistema que les utilitza) i qualitat dependent del sistema. Entre les dimensions habitualment considerades rellevants per a l'avaluació de dades operacionals es troben la completesa, la consistència, la puntualitat (*timeliness*), la unicitat i la validesa.

Batini i Scannapieco [4] ofereixen un marc teòric complet per a la gestió de la qualitat de dades, argumentant que les organitzacions han d'implementar processos sistemàtics d'avaluació adaptats al context específic de cada domini. Els autors emfatitzen que no existeix un conjunt universal de dimensions: la selecció ha de respondre a les necessitats operacionals i als processos de presa de decisions de cada organització. Aquesta premissa justifica, en el context d'aquest projecte, la selecció de les dimensions implementades, escollides inicialment en el marc del projecte WATERSERVE per la seva rellevància específica en el sector de l'aigua.

2.2 Qualitat de Dades en el Sector de l'Aigua

El sector de l'aigua presenta reptes específics per a la qualitat de dades derivats de la naturalesa de les seves fonts: sensors amb periodicitats variables, registres de consum mensuals, i sèries temporals amb buits històrics freqüents. Baena-Miret et al. [3] presenten un *framework* específic per a l'assegurament de la qualitat de dades en aquest sector,

identificant com a principals desafiaments la variabilitat de les freqüències de mesura, la presència de valors anòmals i la dificultat de comunicar resultats de qualitat a equips no tècnics. El seu treball proposa que les mètriques de qualitat s'han de dimensionar per a cada context concret i presentar-se de manera comprensible per als equips de negoci, aspectes que aquest projecte aborda mitjançant la configuració JSON parametritzable i la visualització amb Streamlit.

Un aspecte que no apareix habitualment tractat en els enfocaments revisats és el tractament diferenciat dels buits temporals recents versus els històrics. En sèries temporals de consum d'aigua, un buit recent té un impacte operacional molt superior a un buit de fa anys. Aquesta mancança motiva el desenvolupament de la dimensió *Adjusted Timeliness* en aquest projecte, que aplica una ponderació exponencial per penalitzar els buits recents de manera proporcional a la seva rellevància per als processos de negoci.

2.3 Integració de Qualitat de Dades en Plataformes Cloud

La integració de processos de qualitat de dades directament en plataformes d'emmagatzematge i anàlisi és una tendència creixent en l'arquitectura de dades moderna. Els *Data Hubs* corporatius basats en plataformes cloud permeten centralitzar tant les dades com la lògica de qualitat, reduint la dependència de processos externs i manuals.

Snowflake ofereix mecanismes natius per executar lògica personalitzada sobre les dades: les *Stored Procedures* permeten orquestrar fluxos complexos, mentre que les UDTF (*User Defined Table Functions*) permeten executar codi Python directament sobre les dades emmagatzemades. La clàusula `PARTITION BY`, aplicada sobre les UDTF, permet la utilització de particions i executar el processament de manera distribuïda, aprofitant la infraestructura de la plataforma. Aquesta combinació és especialment rellevant per a casos d'ús amb grans volums de dades com els que es treballen en aquest projecte, on l'alternativa seqüencial resultava inviable en termes de temps d'execució.

2.4 Metodologia Agile Scrum en Projectes Exploratoris

La metodologia Agile Scrum [9] és un framework iteratiu i incremental per a la gestió de projectes que ha demostrat ser especialment efectiva en projectes de desenvolupament de software i transformació digital. Els seus principis de col·laboració, adaptabilitat i entrega contínua de valor són especialment rellevants en projectes exploratoris com aquest, on les decisions tècniques es prenen de manera progressiva.

3 Metodologia i Procés de Desenvolupament

3.1 Metodologia de Desenvolupament

El projecte ha estat desenvolupat seguint la metodologia Agile Scrum, amb sprints de dues setmanes i revisions periòdiques amb l'equip de Synectic. Aquesta metodologia ha permès adaptar-se als canvis que han anat sorgint durant un projecte de caràcter exploratori com aquest, on les decisions tècniques s'han pres de manera progressiva a partir dels resultats obtinguts en cada fase.

El desenvolupament s'ha estructurat en diferents iteracions: inicialment es va implementar una primera solució funcional sobre el cas d'ús d'*Agua Suministrada*, posteriorment es van identificar limitacions de rendiment i escalabilitat, es van aplicar millores aprofitant les capacitats natives de Snowflake i finalment es va evolucionar cap a una arquitectura genèrica reutilitzable per a futurs casos d'ús.

La planificació temporal detallada es pot consultar al diagrama de Gantt de la Figura A.8 de l'Apèndix.

3.2 Evolució de la Solució

El procés de desenvolupament ha seguit una evolució clara en tres fases: una primera implementació funcional però amb limitacions de rendiment, una optimització mitjançant paral·lelització, i finalment una refactorització cap a una arquitectura genèrica. Aquesta evolució ha estat impulsada per les necessitats identificades en cada fase i per la validació contínua amb l'equip de Synectic.

3.2.1 Primera Implementació i Optimització del Processament

La primera implementació es va realitzar sobre el cas d'ús d'*Agua Suministrada*, ja que permetia validar el funcionament del motor DQA sobre dades reals abans d'abordar la generalització del sistema.

En aquesta primera versió, alguns paràmetres estaven definits directament dins de la implementació, com les taules d'origen, les columnes utilitzades, els nivells d'agregació i altres configuracions específiques del cas d'ús.

Aquesta aproximació permetia validar el funcionament del motor i obtenir les primeres mètriques de qualitat. Tanmateix, presentava una limitació important: cada nou cas d'ús que es volgués afegir, requeriria adaptar el codi existent, dificultant la reutilització i escalabilitat del sistema.

Un cop validada aquesta primera solució, es va analitzar el rendiment del processament. La implementació inicial executava el càlcul de manera seqüencial mitjançant una Stored Procedure que iterava sobre cada entitat mínima (actiu) dins d'un bucle: per a cada actiu, es cridava el motor DQA, s'esperava el resultat i es passava al següent. Això significava que els 3.346 actius d'Agua Suministrada es processaven un darrere l'altre, resultant en uns 32 minuts per execució completa. A més, cada nivell d'agregació disposava de la seva pròpia Stored Procedure —una per a actius, una per a activitats i una per a explotacions—, que s'havien d'executar seqüencialment i en ordre, ja que cada nivell depenia dels resultats del nivell inferior.

Per solucionar aquesta limitació es va implementar una UDTF amb PARTITION BY. Aquesta combinació permet que Snowflake distribueixi l'execució entre múltiples particions, executant el càlcul de manera paral·lela i aprofitant la infraestructura distribuïda de la plataforma.

En aquest projecte, cada partició correspon a l'entitat mínima d'anàlisi del cas d'ús: l'actiu per a *Agua Suministrada* i el punt de servei per a *Agua Registrada Devengada*.

3.2.2 Evolució cap a una Arquitectura Genèrica Configurable

Després d'aconseguir una implementació funcional i optimitzada, es va plantejar aplicar la solució al segon cas d'ús: *Agua Registrada Devengada*. En aquest punt es va identificar que l'arquitectura inicial no era adequada per escalar-se a nous casos d'ús.

Per aquest motiu, es va realitzar una refactorització de la solució amb l'objectiu de separar la lògica del motor DQA de les configuracions específiques de cada cas.

La refactorització consisteix en reestructurar el codi i l'arquitectura sense modificar la funcionalitat, però millorant la qualitat, mantenibilitat i reutilització. En aquest projecte, la solució inicial hardcodejada es va transformar en una arquitectura genèrica on els paràmetres es defineixen mitjançant fitxers de configuració JSON.

Amb aquesta nova arquitectura, la UDTF passa a ser completament genèrica i no necessita modificacions per incorporar nous casos d'ús. Cada cas d'ús disposa de la seva pròpia Stored Procedure, que s'encarrega d'orquestrar el flux d'execució, seleccionar les taules corresponents i enviar les dades necessàries al motor DQA. Això consolida en un únic procés orquestrat el que anteriorment requerien tres Stored Procedures independents executades manualment en ordre.

3.3 Arquitectura de la Solució

L'arquitectura final de la solució es basa en tres components principals que treballen de manera coordinada: el Stage de Snowflake per emmagatzemar el codi, la UDTF per executar la lògica en paral·lel, i les Stored Procedures per orquestrar tot el procés. A continuació es descriuen els detalls de configuració i arquitectura que fan possible aquesta integració.

3.3.1 Configuració Genèrica Basada en JSON

La configuració basada en JSON ha estat un pas crucial per a la productivització del projecte. La solució inicial estava hardcodejada, és a dir, els paràmetres (taules d'origen, columnes, nivells d'agregació, etc.) estaven fixats directament a la implementació. Això significava que per a cada nou cas d'ús calia modificar el codi i crear noves Stored Procedures específiques.

La refactorització va transformar aquesta solució en una arquitectura genèrica on tots els paràmetres es defineixen en fitxers de configuració JSON. Aquesta configuració es desa a una taula de configuració de Snowflake, permetent:

- **Extensibilitat**: Afegir nous casos d'ús sense modificar codi, només inserint una nova configuració JSON.
- **Mantenibilitat**: La lògica està centralitzada en la UDTF i les Stored Procedures, facilitant futurs canvis.
- **Reutilitzabilitat**: Els patrons desenvolupats són aplicables a futurs projectes de qualitat de dades.
- **Auditoria**: Cada execució queda registrada amb un snapshot de la configuració utilitzada, facilitant la traçabilitat.

Justificació de la decisió JSON: La configuració basada en JSON va ser escollida per múltiples raons. Primer, l'equip de negoci de Synectic estava acostumat a treballar amb JSON per a configuracions, facilitant la comunicació i els canvis futurs. Segon, Snowflake gestiona JSON de manera nativa mitjançant el tipus VARIANT, sense necessitat de dependències externes. Tercer, Python permet processar JSON sense llibreries addicionals. Finalment, JSON permet definir estructures flexibles i niuades, facilitant parametritzar casos d'ús complexos sense modificar l'esquema de la base de dades. Altres opcions com taules SQL eren més rígides, mentre que YAML i XML afegien complexitat sense benefici real en aquest context.

La configuració defineix elements com la taula d'origen, columnes de timestamp i valor, filtres, claus de partició, nivells d'agregació, mapes de relacions i taules de sortida.

3.3.2 Arquitectura de la Solució

La solució final es basa en tres components principals que treballen de manera coordinada:

- **Stage de Snowflake**: Emmagatzematge intern dels fitxers Python que componen el mòdul DQA. Els fitxers es pugen al stage '@STG_DQA' i es mantenen versionats.
- **UDTF (User Defined Table Function)**: Implementa el càlcul de les mètriques DQA per a cada partició. La UDTF importa el codi Python del stage i l'executa dins de Snowflake, aprofitant la capacitat de processament distribuït mitjançant la clàusula PARTITION BY.
- **Stored Procedures**: Orquestren l'execució end-to-end del pipeline DQA. Llegeixen la configuració, preparen les dades d'entrada, executen la UDTF amb PARTITION BY i agreguen els resultats als nivells superiors.

Flux de Dades i Relació entre Components: Per entendre com aquests components treballen junts, és útil pensarlos com a parts d'un sistema coordinat:

1. El **Stage** actua com un magatzem de codi: emmagatzema els fitxers Python del motor DQA de manera versionada.
2. La **UDTF** actua com el treballador que executa el càlcul: llegeix el codi Python del Stage i executa la lògica DQA per a cada partició (actiu o punt de servei).
3. La **Stored Procedure** actua com el director que coordina tot el procés: llegeix la configuració JSON, prepara les dades, invoca la UDTF amb PARTITION BY (que distribueix l'execució en paral·lel entre múltiples particions), i finalment agrega els resultats als nivells superiors.

La clau de la millora de rendiment és que la clàusula PARTITION BY dins la UDTF permet a Snowflake executar múltiples instàncies de la funció en paral·lel, una per a cada partició (actiu o punt de servei). Això contrasta amb la Stored Procedure seqüencial inicial, que processava les particions una a una dins d'un bucle.

Justificació de l'ús de UDTF amb PARTITION BY: La Stored Procedure seqüencial original processava les particions amb un bucle: agafava una partició, calculava les mètriques, passava a la següent... de forma seqüencial. Podria haver usat PARTITION BY dins d'una window function SQL, però això només serveix per a càlculs SQL senzills, no per executar lògica Python complexa en paral·lel.

La UDTF + PARTITION BY és l'únic mecanisme que permet a Snowflake distribuir l'execució d'una funció Python personalitzada entre múltiples particions simultàniament. Aquesta és una característica específica de les table functions, no aplicable a les Stored Procedures directament. Per tant, la UDTF no era una opció opcional sinó la solució tècnica necessària per aconseguir paral·lelisme en la lògica Python complexa.

A la Figura 1 es mostra el flux complet de dades de l'arquitectura end-to-end.

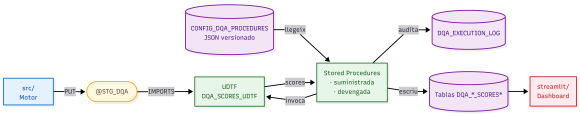


Fig. 1: Flux de dades de l'arquitectura DQA end-to-end: el Stage emmagatzema el codi Python, la UDTF l'executa en paral·lel per particions, i la Stored Procedure orquestra tot el procés

3.4 Funcionament del Motor DQA

Un cop definida l'arquitectura genèrica, el procés d'execució segueix un flux comú per a tots els casos d'ús. En aquesta secció es descriuen els detalls del càlcul de les mètriques en el nivell més baix d'agregació, la incorporació de la nova dimensió temporal, i finalment com s'agreguen els resultats als nivells superiors.

3.4.1 Càlcul de Mètriques del Nivell 1

Un cop definida l'arquitectura genèrica, el procés d'execució segueix un flux comú per a tots els casos d'ús.

La Stored Procedure corresponent a cada cas d'ús inicia el procés llegint la configuració JSON associada. Posteriorment, prepara les dades d'entrada i executa la UDTF, que calcula les mètriques de qualitat del nivell més baix d'agregació.

El nivell 1 (*actiu* per a *Agua Suministrada* o *punt de servei* per a *Agua Registrada Devengada*) es calcula mitjançant la UDTF, que:

1. Construeix la sèrie temporal per a cada partició.
2. Infereix o força la freqüència (individualment per a cada actiu en *Agua Suministrada*; fixa a mensual en *Agua Registrada Devengada*).
3. Aplica regles de validació i neteja.
4. Calcula les 6 mètriques DQA (*Completeness*, *Consistency*, *Timeliness*, *Adjusted Timeliness*, *Uniqueness*, *Validity*) de manera independent per a la sèrie temporal de cada partició.
5. Retorna una fila amb les mètriques i la freqüència inferida.

Per a *Agua Registrada Devengada*, s'ha implementat una millora important: en comptes d'agafar freqüències de 30 o 31 dies (que causava blocs fixes i resultats poc fiables), s'utilitza la funció de pandas 'MS' [8] (*month start*), que detecta automàticament el primer dia de cada mes. Això permet que el càlcul del *Timeliness* sigui correcte independentment del nombre de dies que tingui cada mes.

3.4.2 Incorporació de la Mètrica *Adjusted Timeliness*

Durant les primeres execucions del motor DQA ja integrat a SNowflake, es va introduir una nova dimensió de qualitat denominada *Adjusted Timeliness*. Aquesta mètrica es va dissenyar amb l'objectiu d'ajustar el càlcul tradicional de *Timeliness*, partint de la hipòtesi que els buits temporals recents tenen un impacte operacional superior als buits històrics i, per tant, haurien de penalitzar més el resultat final de la mètrica.

La mètrica està basada en una ponderació exponencial dels intervals temporals esperats, de manera que els buits més recents tenen un pes superior que els buits antics. Això permet donar més importància a la disponibilitat actual de les dades, que és habitualment més rellevant per als processos operatius i de negoci.

El càlcul es realitza primer resamplejant la sèrie temporal segons la freqüència esperada de cada actiu, generant tots els intervals temporals que haurien de contenir dades. Posteriorment, cada interval rep un pes calculat mitjançant:

$$w_t = e^{-\lambda \times d_t} \tag{1}$$

on d_t és el nombre de dies des de la data inicial (primera lectura disponible de l'actiu) fins a la data més recent disponible, i λ és un paràmetre configurable que controla la intensitat de la penalització. En la versió implementada està definida a $\lambda = 0.0005$, que equival a un pes de 0.7 als 2 anys d'antiguitat.

El valor final de la mètrica es calcula com:

$$\text{Adjusted Timeliness} = \frac{\sum(w_t \text{ amb dades})}{\sum(w_t \text{ totals})} \tag{2}$$

Aquesta mètrica permet mantenir el mateix rang de valors que *Timeliness*, facilitant la comparació entre ambdues dimensions. Els resultats obtinguts mostren que, en els casos d'ús analitzats, les diferències entre *Timeliness* i *Adjusted Timeliness* són mínimes, indicant que la ponderació temporal no modifica significativament l'avaluació de qualitat en aquests datasets. Això suggereix que el problema de qualitat temporal no és tant la distribució dels buits (antics vs recents) sinó l'absència general de registres en les sèries temporals.

3.4.3 Agregació per Nivells Superiors

Els nivells superiors es calculen dins de la lògica de les Stored Procedures, implementant una **mitjana ponderada pel consum**.

Recordem que pel cas d'ús *Agua Suministrada* farem l'agregació d'actiu a activitat, i després d'activitat a explotació. Per al cas d'*Agua Registrada Devengada* es fa només l'agregació de punt de servei a explotació.

Per a *Agua Suministrada* (Nivell 2: Actiu → Activitat):

Considerem una activitat A, amb actius relacionats $\{a_1, \ldots, a_n\}$, on cada actiu a_j té consum consum_j i una mètrica mètrica_j. El consum total de l'activitat A és $\text{consum total}_A = \sum_{j=1}^{n} \text{consum}_j$. El pes de l'actiu a_i és:

$$w_i = \frac{\text{consum}_i}{\text{consum total}_A} \tag{3}$$

La mètrica de l'activitat A és:

$$\text{mètrica}_A = \sum_{j=1}^{n} w_j \times \text{mètrica}_j \tag{4}$$

Per a *Agua Suministrada* (Nivell 3: Activitat → Explotació):

De manera anàloga, per una explotació E, formada per les activitats relacionades $A_1, \ldots, A_r$, el pes d'una activitat A_k és:

$$w_k = \frac{\text{consum total}_{A_k}}{\sum_{l=1}^{r} \text{consum total}_{A_l}} \tag{5}$$

La mètrica de l'explotació E es calcula com:

$$\text{mètrica}_E = \sum_{l=1}^{r} w_{A_l} \times \text{mètrica}_{A_l} \tag{6}$$

Per a *Agua Registrada Devengada* (Nivell 2: Punt de Servei → Explotació):

El procés és anàleg al de *Agua Suministrada*, però operant sobre els punts de servei en comptes d'actius.

Configuracions Específiques Generals:

- Si la suma de pesos és ≤ 0 (tots els consums zero/negatius descartats pel filtre), es fa fallback a mitjana aritmètica simple.
- Si un grup té una sola entitat filla, es copien directament els scores sense ponderar.
- Els consums negatius (resets de comptador) es filtren abans de calcular el pes, de manera que no afecten la validesa de la mitjana ponderada.

3.5 Validació sobre Agua Registrada Devengada

Un cop validada l'arquitectura genèrica amb el cas d'ús d'*Agua Suministrada*, es va procedir a aplicar la mateixa solució sobre *Agua Registrada Devengada*. Aquesta fase va permetre comprovar que la generalització realitzada permetia incorporar nous casos d'ús sense modificar la lògica principal del motor DQA.

Per implementar aquest nou cas d'ús, només va ser necessari crear una nova Stored Procedure amb la configuració específica corresponent. La UDTF no va requerir cap modificació, ja que la seva lògica havia estat desacoblada dels paràmetres particulars de cada font de dades mitjançant la configuració JSON.

Durant les primeres execucions sobre aquest conjunt de dades es va identificar una nova limitació relacionada amb el volum de dades processat. A diferència d'*Agua Suministrada*, que treballava amb 3.346 actius, *Agua Registrada Devengada* conté més de 5 milions de punts de servei, fet que va incrementar considerablement les necessitats computacionals.

Inicialment, l'execució es va intentar realitzar amb un *warehouse* de mida XS, utilitzat anteriorment per al primer cas d'ús. Tanmateix, aquest dimensionament no era suficient per processar el volum de dades dins d'un temps raonable. Per aquest motiu, es va augmentar la capacitat de còmput utilitzant un *warehouse* de mida M, que va ser l'opció disponible aprovada per l'equip de negoci de Synectic.

Amb aquesta configuració, el procés va completar correctament l'execució del motor DQA, obtenint les mètriques de qualitat per als nivells d'agregació definits. Aquesta validació va permetre confirmar tant la reutilització de l'arquitectura genèrica com la capacitat de la solució per adaptar-se a casos d'ús amb volums de dades molt superiors.

3.6 Tecnologies

Per al desenvolupament i desplegament de la solució s'han utilitzat diferents eines tecnològiques:

- **Atlassian Confluence**: [1] Plataforma de documentació col·laborativa utilitzada per a la documentació tècnica del projecte.
- **Git/GitLab**: [6] Control de versions del codi, seguint l'estructura d'un repositori corporatiu.
- **Jira**: [2] Gestió de tasques i seguiment del progrés per sprints.
- **Pandas**: Biblioteca de Python per a manipulació de dades, especialment per a la funció 'MS' (*month start*).
- **Python 3.10**: Llenguatge de programació del mòdul existent i de la lògica d'integració, compatible amb Stored Procedures a Snowflake.
- **Snowflake**: Plataforma principal d'integració i execució del mòdul DQA.
- **Streamlit**: [11] Eina de visualització de resultats, incorporada com a ampliació per facilitar la presentació de les mètriques de qualitat a l'equip de negoci.
- **Visual Studio Code**: Entorn de desenvolupament local, amb extensió de Snowflake per a la gestió de la sessió i l'execució de comandes.

4 Resultats i Discussió

4.1 Volum de Dades Processades

El motor DQA ha processat correctament les dades dels dos casos d'ús:

Agua Suministrada:

- Actius únics: 3.346
- Activitats: 756
- Explotacions: 623
- Mida de dades: 5.7 milions de registres, 99.4 MB

Agua Registrada Devengada:

- Punts de servei únics: 5.588.428
- Explotacions: 831
- Mida de dades: 291.702.744 registres, aproximadament 22,73 GB

4.2 Mètriques de Qualitat Obtingudes

Les mètriques de qualitat s'han calculat per a ambdós casos d'ús en els seus respectius nivells d'agregació. A continuació es presenten els resultats en dues taules.

Interpretació dels Resultats:

Els resultats mostren patrons interessants en ambdós casos d'ús que reflecteixen la naturalesa diferent de les dades.

Per a *Agua Suministrada* (Taula 3), les mètriques de *Completeness*, *Consistency* i *Uniqueness* són altes (superiors a 0.95), indicant que la majoria de registres són vàlids i únics. Tanmateix, *Timeliness* és més baixa (0.60 a nivell d'actiu), indicant que hi ha buits significatius en les sèries temporals. Això era esperat per l'equip de negoci, ja que les dades històriques contenen períodes amb registres incomplets. Els valors d'*Adjusted Timeliness* són molt similars als de *Timeliness*, indicant que el problema no és la distribució dels buits (antics vs recents) sinó l'absència general de registres.

Per a *Agua Registrada Devengada* (Taula 4), la qualitat de dades és molt alta en general, amb *Completeness*, *Consistency* i *Validity* pràcticament perfectes (1.00). Els valors de *Timeliness* i *Adjusted Timeliness* al voltant del 94-96% indiquen que hi ha alguns buits en les sèries temporals, però són minoritaris. Aquesta qualitat superior es deu a la naturalesa més regular i controlada de les dades mensuals. Les diferències entre les variants Lineal i Ajustada són mínimes, fet que indica que, per a aquest conjunt de dades, l'ajust aplicat sobre el consum no altera de manera rellevant els valors finals de les mètriques de qualitat.

Nota sobre Llindar de Qualitat: No s'ha definit un llindar concret per a aquestes mètriques. Els valors acceptables dependran de les necessitats específiques de cada organització. En aquest projecte, l'equip de negoci de Synectic podrà definir els seus propis llindar segons els requisits operacionals. Una de les funcionalitats principals de l'aplicació Streamlit és precisament permetre establir aquests llindar de manera flexible, facilitant la identificació de sèries temporals que no compleixen els criteris de qualitat desitjats.

4.3 Millora de Rendiment

La principal aportació tècnica ha estat la millora dràstica de rendiment aconseguida amb la UDTF i PARTITION BY en el cas d'*Agua Suministrada*:

- **Execució anterior** (Stored Procedure seqüencial): 32 minuts per execució completa.
- **Execució nova** (UDTF amb PARTITION BY): 28 segons per execució completa.
- **Millora**: 98% de reducció de temps.

Aquesta millora va ser aconseguida gràcies a l'ús de les funcionalitats natives de Snowflake.

Per a *Agua Registrada Devengada*, l'execució requereix 1 hora 16 minuts amb un warehouse de mida M (en comptes de XS utilitzat per a *Agua Suministrada*). Aquesta diferència es deu a dos factors: (1) el volum de dades és molt superior (5,5 milions de punts de servei vs 3.346 actius), i (2) es fa dues vegades el càlcul (variants Lineal i Ajustada), duplicant el temps. Aquest resultat confirma que, tot i que l'arquitectura és reutilitzable per a conjunts de dades de dimensions molt diferents, el dimensionament del warehouse esdevé un factor determinant per mantenir uns temps d'execució acceptables.

Decisió sobre Execució Periòdica: Inicialment es va considerar implementar un scheduler de Snowflake per a execucions automàtiques periòdiques del motor DQA. Tanmateix, aquesta opció va ser descartada per dues raons principals: (1) les mètriques de qualitat es calculen sobre dades històriques acumulades de molts anys, per la qual cosa els nous registres de l'últim dia o setmana no modifiquen significativament els valors de les mètriques, i (2) l'execució és computacionalment costosa pel cas d'*Agua Registrada Devengada* (1 hora 16 minuts) i executar-la recurrentment consumiria massa crèdits de Snowflake, que és un recurs compartit amb altres projectes. Per tant, s'ha decidit executar el motor DQA sota demanda quan sigui necessari, en comptes de de manera periòdica.

4.4 Arquitectura Genèrica i Escalabilitat

Aquesta validació representa un dels principals resultats del projecte, ja que demostra que la incorporació de nous casos d'ús no requereix modificar el motor DQA, sinó únicament definir la configuració corresponent i adaptar l'orquestració del procés. Això redueix considerablement l'esforç de manteniment i facilita l'evolució futura del sistema. La incorporació d'*Agua Registrada Devengada* es va realitzar mitjançant una nova configuració JSON i una Stored Procedure específica, mantenint la mateixa UDTF i sense modificar la lògica de càlcul.

Aquesta capacitat de reutilització confirma que el desacoblament entre configuració i motor DQA millora l'escalabilitat del sistema i facilita futures extensions del Data Hub.

4.5 Visualització amb Streamlit

S'ha desenvolupat una aplicació Streamlit funcional amb les següents pestanyes:

- **Pestanya General**: Mostra una visió global dels resultats (nombre total d'entitats, valors mitjans de les 6 mètriques, gràfic radar integrat).
- **Pestanya Explotacions**: Permet explorar les mètriques a nivell d'explotació, amb funcionalitat d'ordenació i cerca per ID o nom en concret. També llista les activitats o punts de servei que formen part de cada explotació.
- **Pestanya Alertes**: Funcionalitat especialment útil per a l'equip de negoci. Permet seleccionar una mètrica i establir un valor de llindar, retornant una llista de totes les explotacions, activitats i actius que estan per sota d'aquell llindar, ordenats per valor de mètrica (els pitjors primers).

TAULA 3: MÈTRIQUES DE QUALITAT PER A *Agua Suministrada* PER NIVELL D'AGREGACIÓ

Nivell	*Completeness*	*Validity*	*Timeliness*	*Adj. Timeliness*	*Consistency*	*Uniqueness*
Actiu	0.976	0.960	0.598	0.599	0.999	0.984
Activitat	0.972	0.959	0.638	0.644	1.000	0.988
Explotació	0.971	0.953	0.631	0.637	1.000	0.989

TAULA 4: MÈTRIQUES DE QUALITAT PER A *Agua Registrada Devengada* PER NIVELL D'AGREGACIÓ I VARIANT

Taula	*Completeness*	*Consistency*	*Timeliness*	*Adj. Timeliness*	*Uniqueness*	*Validity*
Punt Serv. Lineal	1.00	1.00	0.940	0.941	0.998	1.00
Punt Serv. Ajustada	1.00	1.00	0.943	0.944	0.998	1.00
Explotació Lineal	1.00	1.00	0.956	0.956	0.999	1.00
Explotació Ajustada	1.00	1.00	0.959	0.959	0.999	0.999

L'aplicació Streamlit està desplegada en l'entorn de preproducció i s'alimenta de les taules de resultats generades pel motor DQA per a ambdós casos d'ús.

A l'Apèndix A.3 es poden consultar captures de l'aplicació per a ambdós casos d'ús.

4.6 Discussió Crítica dels Resultats

Els resultats obtinguts demostren que la solució proposada és viable i escalable. La millora de rendiment del 98% en *Agua Suministrada* transforma la viabilitat del sistema en producció, fent-lo pràctic per a execucions sota demanda en entorns de producció. La validació sobre dos casos d'ús amb característiques molt diferents també confirma que l'arquitectura genèrica és reutilitzable i facilita l'extensió del sistema a futurs conjunts de dades.

Tanmateix, hi ha alguns aspectes a considerar:

- **Volum de dades**: El cas d'*Agua Registrada Devengada* requereix un warehouse de mida M i 1 hora 16 minuts per execució, el que pot ser limitant per a execucions molt freqüents. Tanmateix, donat que les mètriques no es preveu que canviin gaire amb dades noves (per la raó explicada anteriorment), s'ha decidit executar el motor només quan sigui necessari.

- **Adjusted Timeliness:** Els resultats obtinguts mostren que les diferències entre *Timeliness* i *Adjusted Timeliness* són mínimes en ambdós casos d'ús. Això suggereix que la hipòtesi inicial—que els buits recents penalitzarien significativament més que els històrics—no s'ha confirmat amb aquestes dades en concret. Una interpretació possible és que el problema de qualitat temporal no és tant la distribució dels buits en el temps sinó l'absència general de registres al llarg de tota la sèrie temporal. Tot i això, la dimensió aporta valor com a mètrica addicional per a futurs conjunts de dades on aquesta distribució sí que pugui ser rellevant.

- **Integració amb sistemes de negoci**: L'aplicació Streamlit és actualment un mock-up per a exploració. Per a producció, es prevé desenvolupar un Power BI que integri millor amb els sistemes corporatius.

5 CONCLUSIONS

5.1 Aportacions Realitzades

En aquest projecte s'han realitzat les següents aportacions principals:

1. **Integració completa del motor DQA a Snowflake**: El mòdul de qualitat de dades està completament integrat i funcional, processant 3.346 actius, 756 activitats i 623 explotacions en *Agua Suministrada*, i 5.588.428 punts de servei i 831 explotacions en *Agua Registrada Devengada*.

   A més, s'ha desenvolupat i avaluat la nova dimensió temporal (*Adjusted Timeliness*), que tot i que aquesta mètrica no modifica significativament l'avaluació de qualitat en els casos d'ús analitzats, pot ser útil en casos d'ús futurs.

2. **Arquitectura genèrica escalable**: Pas d'una solució hardcodejada a un sistema configurable basat en JSON, que permet afegir nous casos d'ús sense modificar codi.

3. **Millora de rendiment del 98%**: Implementació de UDTF amb PARTITION BY en *Agua Suministrada*, aconseguint una reducció de 32 minuts a 28 segons per execució.

4. **Visualització de resultats**: Desenvolupament d'aplicació Streamlit amb pestanyes de general, explotacions i alertes, especialment útil per a l'equip de negoci, implementada per a ambdós casos d'ús.

5. **Documentació de patrons reutilitzables**: Creació de patrons (UDTF amb PARTITION BY, configuració JSON, agregació per nivells) que són aplicables a futurs projectes de qualitat de dades.

6. **Preparació per a futurs casos d'ús**: Demostració que la solució és reutilitzable, com s'ha evidenciat amb la implementació d'*Agua Registrada Devengada*.

5.2 Objectius Assolits

- **OE1**: Completat. S'ha analitzat i comprès el mòdul DQA existent.

- **OE2**: Completat. S'ha estudiat l'arquitectura del Data Hub i les capacitats de Snowflake.
- **OE3**: Completat. S'ha proposat i avaluat l'arquitectura basada en UDTF + Stored Procedures genèriques.
- **OE4**: Completat. S'ha implementat la solució per a ambdós casos d'ús (*Agua Suministrada* i *Agua Registrada Devengada*).
- **OE5**: Completat. S'ha desenvolupat la visualització amb Streamlit per a ambdós casos d'ús.
- **OE6**: Completat. S'ha aconseguit optimitzar la solució inicial a partir de les capacitats d'Snowflake.

5.3 Limitacions Identificades

- **Execució sota demanda**: No s'ha automatitzat l'execució periòdica del motor DQA atès que les mètriques es calculen sobre dades històriques acumulades, i els nous registres no modifiquen significativament els valors resultants. A més, l'eina està orientada principalment a identificar explotacions amb mètriques baixes, un anàlisi que no requereix actualització constant. Per tant, s'ha optat per una execució sota demanda quan sigui necessari.
- **Integració amb sistemes de negoci**: L'aplicació Streamlit és un mock-up d'exploració. Per a producció, es prevé desenvolupar un Power BI.

5.4 Possibles Extensions Futures

- **Noves dimensions de qualitat**: Afegir noves mètriques DQA; això requerirà actualitzar la configuració JSON.
- **Més fonts de dades**: Integrar més casos d'ús requerirà només inserir una nova configuració JSON a la taula de configuració.
- **Automatització periòdica**: Implementar un scheduler de Snowflake per a execucions automàtiques si es determina que és necessari (per exemple, si les mètriques canvien significativament amb dades noves).
- **Expansió de Streamlit**: Afegir més visualitzacions (forecasting, detecció d'anomalies, comparatives temporals).
- **Integració amb sistemes de notificació**: Connectar les alertes de Streamlit amb email o Slack.
- **Reutilització en altres projectes**: Els patrons desenvolupats són aplicables a futurs projectes de qualitat de dades o processament de sèries temporals.

AGRAÏMENTS

Vull expressar el meu agraïment a Cetaqua i Synectic per haver-me donat l'oportunitat de desenvolupar aquest projecte. Ha estat una experiència molt enriquidora que m'ha permès tancar la meva etapa universitària de la manera més rodona possible.

Gràcies a la meva tutora, la Dra. Cristina Fernández Córdoba, per l'acompanyament i el suport rebut al llarg de tot el camí.

Gràcies també a les meves amigues de la universitat, sense les quals aquests quatre anys no haurien estat el mateix, i als meus companys de feina, que han estat un gran referent per a mi.

Finalment, gràcies a la meva família, per haver estat sempre al meu costat.

REFERÈNCIES

[1] Atlassian, "Confluence Documentation," [En línia]. Available: https://www.atlassian.com/software/confluence. [Consultat: Maig 2026].

[2] Atlassian, "Jira Software Documentation," [En línia]. Available: https://support.atlassian.com/jira-software-cloud/. [Consultat: June 2026].

[3] S. Baena-Miret, T. Vrijhoeven, A. Aksan, A. Vedruccio, S. Seshan, D. Rosado Rodríguez, J. Haro Reyes, S. van der Meulen, M. Basile, E. Vargiu, G. Antzoulatos, and S. Vrochidis, "Ensuring data quality in the water sector: Challenges, framework, and best practices," *Cambridge Prisms: Water*, vol. 1, p. e1, 2024.

[4] C. Batini and M. Scannapieco, *Data and Information Quality: Dimensions, Principles and Techniques*. Springer, 2016.

[5] Cetaqua, "Centre Tecnològic de l'Aigua – Web corporativa," [En línia]. Available: https://www.cetaqua.com. [Consultat: June 2026].

[6] Git, "Git Documentation," [En línia]. Available: https://git-scm.com/doc. [Consultat: June 2026].

[7] International Organization for Standardization, "ISO/IEC 25012:2008 – Data quality model," [En línia]. Available: https://www.iso.org/standard/35736.html. [Consultat: June 2026].

[8] The Pandas Development Team, "Pandas Documentation," [En línia]. Available: https://pandas.pydata.org/docs/. [Consultat: June 2026].

[9] K. Schwaber and J. Sutherland, "The Scrum Guide," [En línia]. Available: https://scrumguides.org/scrum-guide.html. [Consultat: June 2026].

[10] Snowflake Inc., "Snowflake Documentation," [En línia]. Available: https://docs.snowflake.com. [Consultat: June 2026].

[11] Streamlit Inc., "Streamlit Documentation," [En línia]. Available: https://docs.streamlit.io. [Consultat: June 2026].

[12] Synectic, "Web corporativa," [En línia]. Available: https://www.synectic.es. [Consultat: June 2026].

[13] Veolia, "Veolia – Web corporativa," [En línia]. Available: https://www.veolia.com. [Consultat: June 2026].

APÈNDIX

A.1 Estructura del Repositori

El repositori GitLab corporatiu està organitzat de la següent manera:

- **src/**: Codi Python del motor DQA (DQA.py, DQA_scores.py, Params.py, Compare.py, Utilities.py).
- **deploy/**: Scripts SQL numerats per al desplegament a Snowflake (00_setup_stage.sql fins a 12_test_5k_devengada_lineal.sql).
- **streamlit/**: Aplicació Streamlit (streamlit_dqa.py, environment.yml).
- **docs/**: Documentació funcional i tècnica (01_dqa_general.md, 02_dqa_sps_y_udtf.md, 03_dqa_estructura_proyecto.md).

Aquesta estructura modular permet mantenir el codi, els scripts de desplegament, la visualització i la documentació de manera organitzada i fàcil de navegar. Com es pot veure a la Figura A.1, la jerarquia de carpetes reflecteix les diferents fases del projecte: desenvolupament del motor (src/), desplegament a Snowflake (deploy/), visualització de resultats (streamlit/) i documentació (docs/).

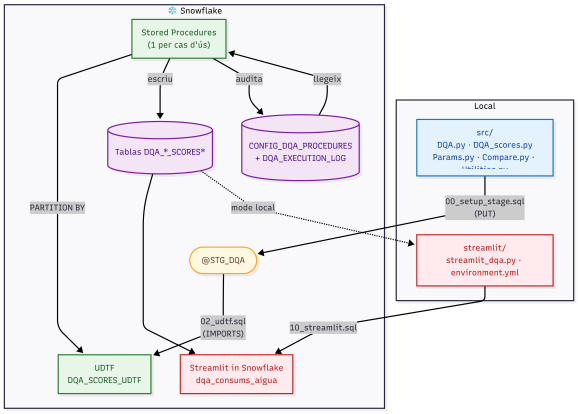


Fig. A.1: Estructura de carpetes del repositori GitLab corporatiu

A.2 Exemple de Configuració JSON

La configuració JSON per a *Agua Suministrada* defineix els paràmetres següents:

- **Metadata**: Nom, descripció, versió, autor i data de creació.
- **Source**: Taula d'origen, columnes de timestamp i valor, filtres.
- **Partition Keys**: Claus de partició que defineixen cada actiu únic.
- **Dimensions**: Les 6 mètriques a calcular.
- **Aggregation Levels**: Definició dels 3 nivells d'agregació (Actiu, Activitat, Explotació), incloent tipus d'agregació, taules de sortida, mapes de relacions.
- **UDTF Config**: Paràmetres de la UDTF (nom de la funció, esquema, paràmetres de validació).
- **Output**: Esquema de sortida i estratègia (OVERWRITE).

A.3 Captures de l'Aplicació Streamlit

Les captures següents (Figures A.2–A.7) mostren l'aplicació Streamlit desenvolupada per a *Agua Suministrada*. Cal destacar que totes les funcionalitats mostrades estan igualment implementades per a *Agua Registrada Devengada*.

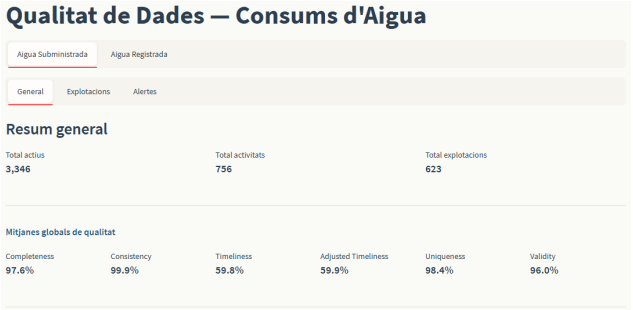


Fig. A.2: Pestanya general de l'aplicació Streamlit: visió global de les mètriques per a *Agua Suministrada*.

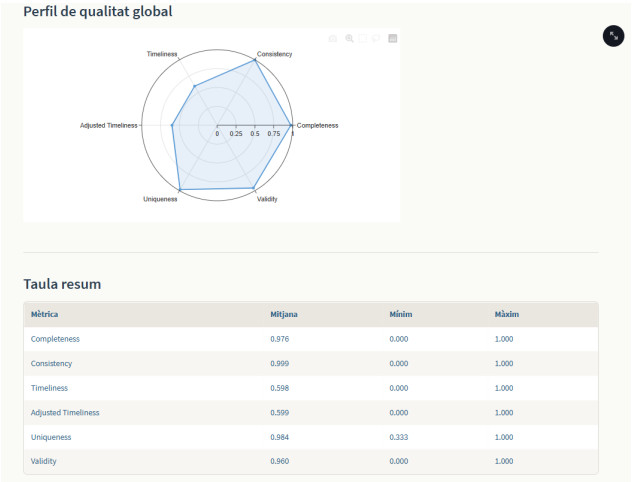


Fig. A.3: Pestanya general de l'aplicació Streamlit: gràfic radar de les mètriques per a *Agua Suministrada*.

A.4 Diagrama de Gantt de la Planificació

El diagrama de Gantt següent mostra la planificació temporal del projecte, estructurada en sprints de dues setmanes seguint la metodologia Agile Scrum.

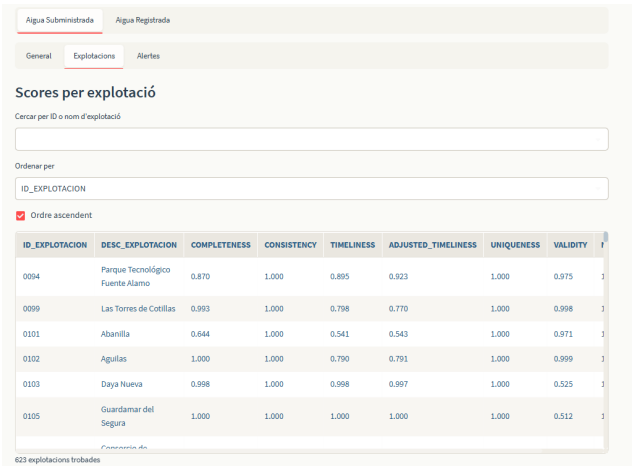


Fig. A.4: Pestanya d'explotacions: llistat de mètriques per explotació per a *Agua Suministrada*.

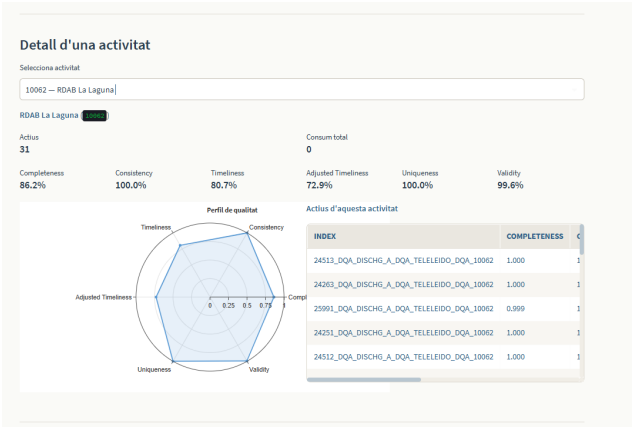


Fig. A.5: Pestanya d'explotacions: funcionalitat de cerca d'activitats per ID o nom dins d'una explotació.

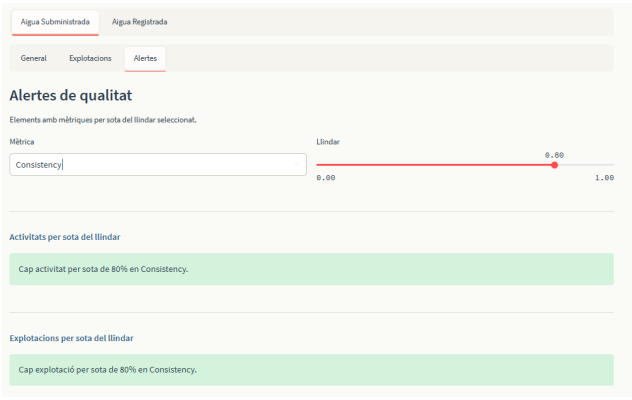


Fig. A.6: Pestanya d'alertes per a *Agua Suministrada*: totes les explotacions compleixen la mètrica de *Consistency* amb llindar 0.8.



Fig. A.7: Pestanya d'alertes per a *Agua Registrada Devengada*: explotacions que no compleixen la mètrica de *Timeliness* amb llindar 0.95.

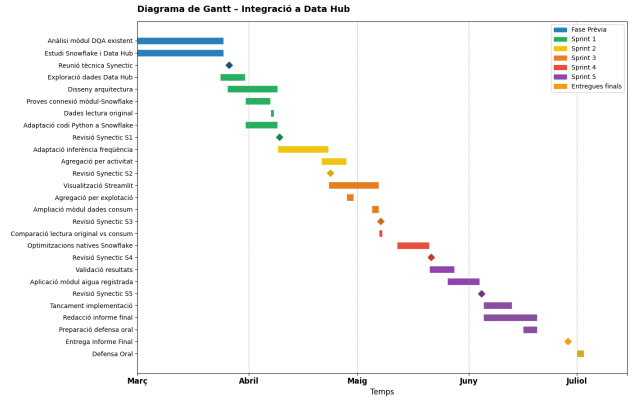


Fig. A.8: Diagrama de Gantt de la planificació del projecte.