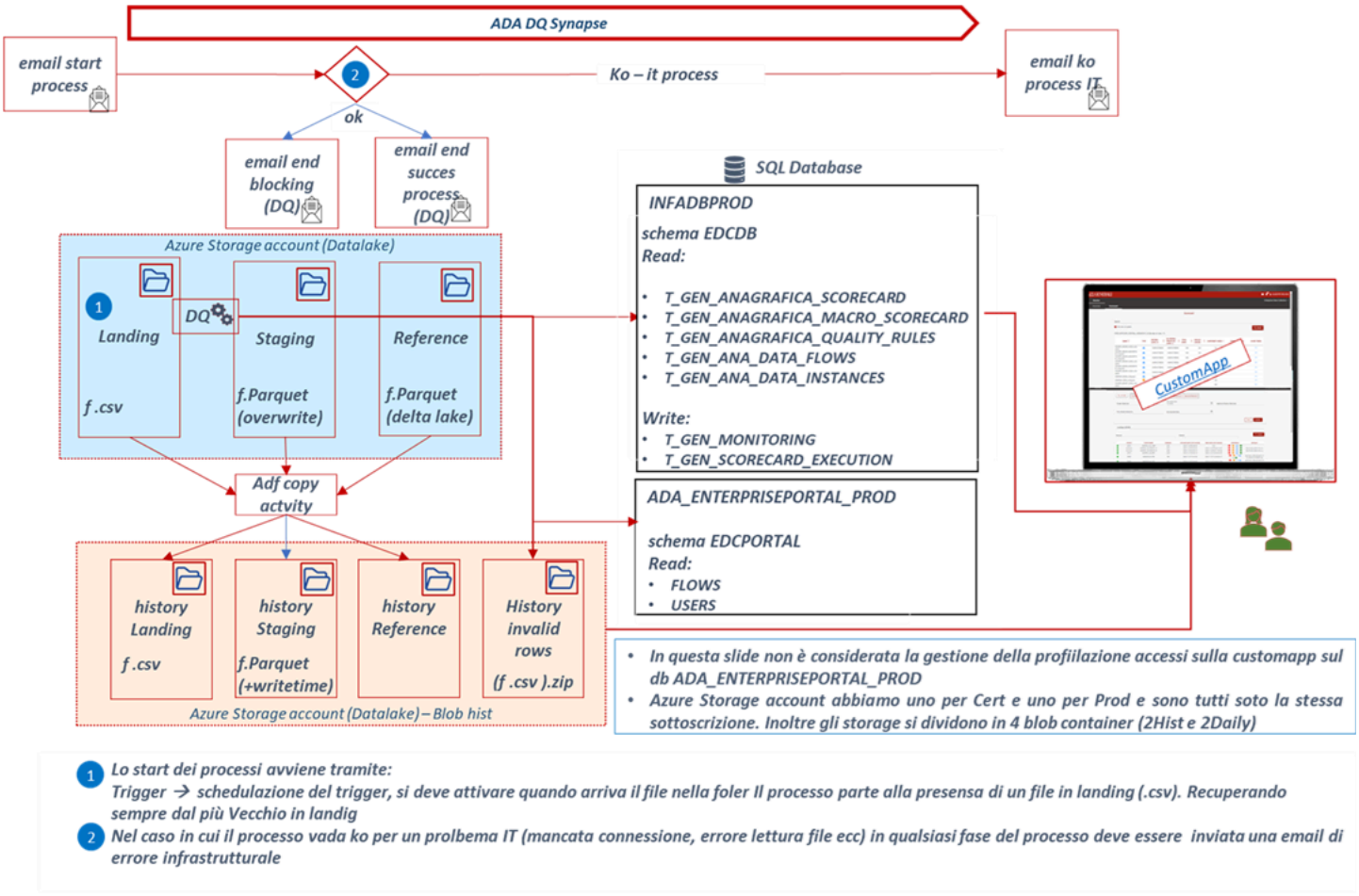
Framework DQ ADA - Synapse - DRAFT

Last updated by | Brunetti Milena (NTT Data) | 14 Jun 2025 at 18:12 CEST

In questa sezione viene descritta la procedura ETL ADA sul prodotto Synapse.

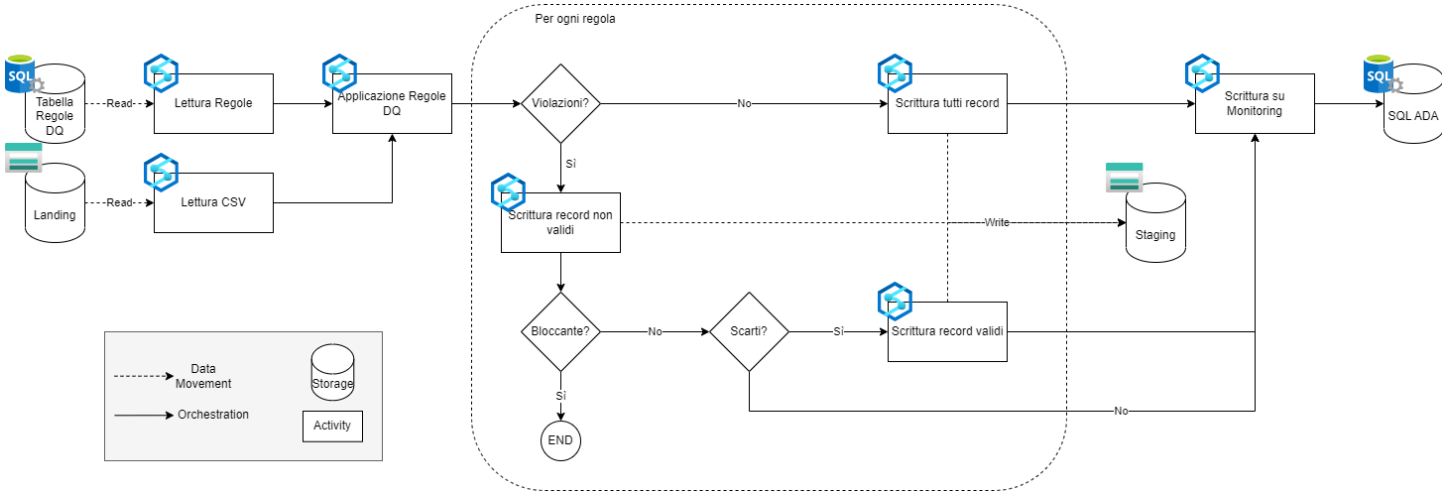
esempio di ppt per aggiornare le relative immagini
Framework_Ada_Synapse.pptx

Di seguito il flow di processo ad alto livello:



DQ Framework su Synapse ADA

Di seguito viene mostrato il flusso di esecuzione del framework di Data Quality sviluppato su Synapse basandosi sulla libreria Great Expectations.



Di seguito vengono dettagliati li step di esecuzione del Framework.

Lettura files CSV

Il framework legge i files CSV dal path ricevuto in input.

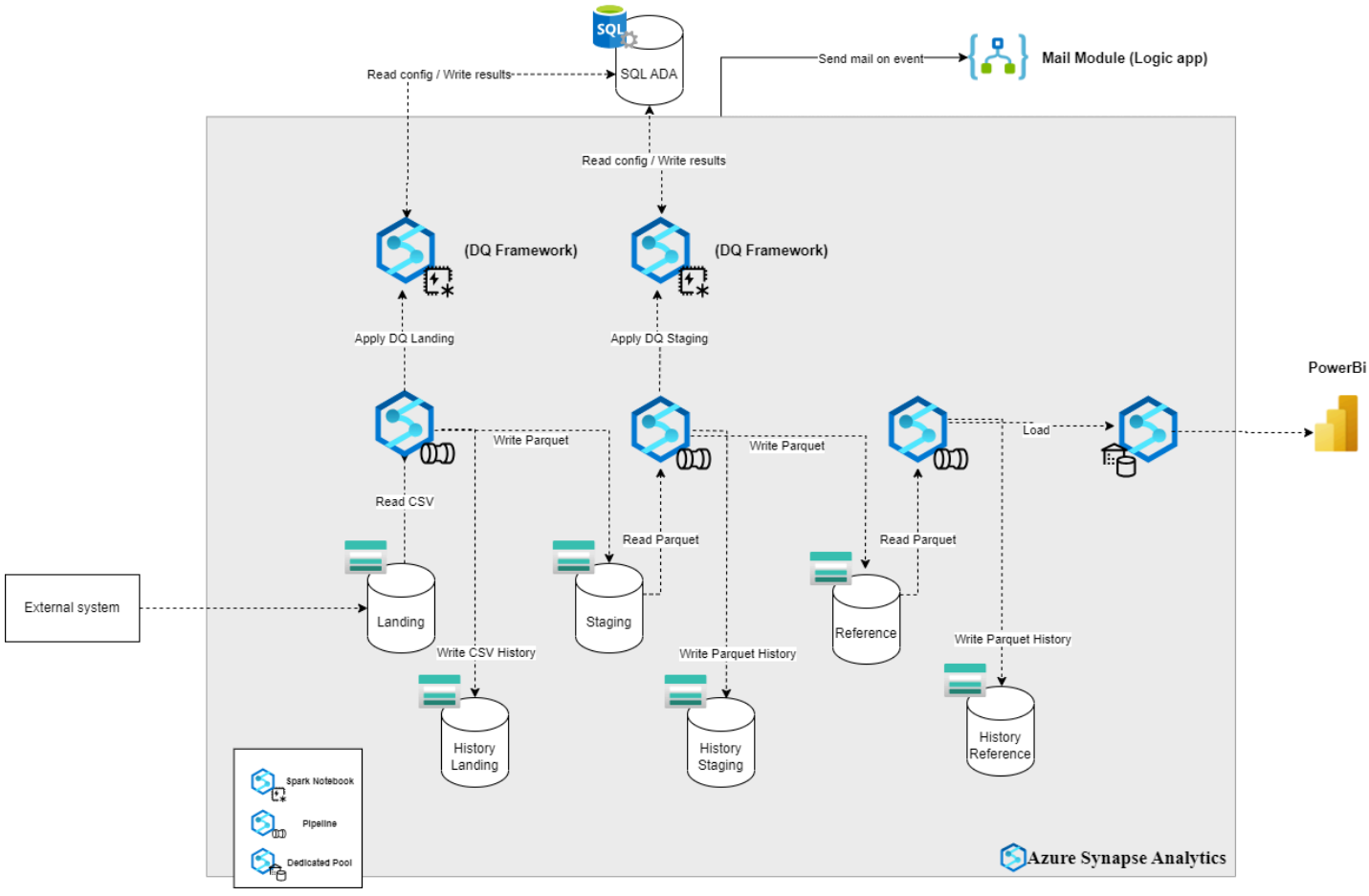
Lettura Regole

Il framework prevede la lettura delle regole di Data Quality da applicare sul file in input usando dei files YML di configurazione salvati su Data Lake, in combinazione con una tabella salvata su SQL Server.

INSERIRE QUI QUALI SONO GLI ELEMENTI SALVATI SU YML E QUALI SU TABELLA

Processo ETL Synapse ADA

Di seguito viene rappresentato il diagramma di flusso che rappresenta l'esecuzione della pipeline all'interno del servizio Synapse.



Il processo avrà come input File in formato csv che verranno depositati sul Datalake Azure nel path di landing **landing/prj_idqs/**.

Per ogni flusso potranno essere caricati due file (uno di anagrafica e uno di risultati) identificabili dal naming; per ciascun dominio saranno usate directory differenti:

- i files afferenti all'anagrafica arriveranno sul path **landing/prj_idqs/ana/<dominio>** e avranno la naming convention: **<identificativo_file>_DQCONTROL_<yyyyMMddHHmmSS>.csv**; ad esempio **ITS_000003949_DQCONTROL_20231127110000.csv**
- i files afferenti ai fatti arriveranno sul path **landing/prj_idqs/fct/<dominio>** e avranno la naming convention: **<identificativo_file>_DQCONTROLRESULTS_<yyyyMMddHHmmSS>.csv**; ad esempio

ITS_000003949_DQCONTROLRESULTS_20231127111200.csv

La schedulazione della pipeline sarà di tipo event-based, di tipo Storage Event sui path sopra definiti. Quindi all'arrivo di ogni file verrà triggerata la pipeline che elaborerà ognuna il singolo file appena arrivato. Le pipelines saranno quindi indipendenti l'una dall'altra.

1. Invio mail Start processo e storicizzazione file input

In questa fase il processo recupererà a partire dal pattern del filename *(da capire con più precisione appena avremo il naming definitivo)* le informazioni relative al flusso che si sta caricando. Ogni flusso infatti sarà censito sulla tabella FLOWS nel database SQL Server e oltre alle informazioni sul flusso andranno recuperati anche i destinatari delle email *(da definire ancora con Luciano la gestione delle email)*

Oltre ad inviare l'email di start il file in landing deve essere copiato nel relativo container di history sul datalake, aggiungendo le relative sottocartelle per anno/mese/giorno : es.
h_landing/prj_idqs/YYYY/MM/DD/file_input

2. Data Quality, gestione invalid rows e area di staging

in questa fase  verranno applicate le regole di qualità tramite il framework di DQ, le regole da applicare verranno censite sul database SQL Server *(da aggiungere allegato con mappatura campi/tabelle anagrafiche regole)*

Le regole possono essere di 3 tipologie:

- BLOCKING
- ALERT WITH DISCARD
- ALERT WITHOUT DISCARD

Per ogni regola che ha almeno un record che non supera il relativo controllo verranno create le invalid rows (file csv divisi per nome regola fallita contenenti i relativi record di input che hanno fallito la specifica regola, se un record fallisce più controlli sarà presente in ognuno dei csv per cui ha fallito il controllo) i csv verranno zippati in un unico zip (*nomeinput_InvalidRows.zip*) e salvati sul datalake nel path di history h_invalidrows/prj_idqs/ZIP/YYYY/MM/DD/nome_zip

Se NON ci sono controlli con esito KO di tipo BLOCKING il file di input verrà portato nel path di staging staging/prj_idqs/ e convertito in .parquet (se tutti i record superano i controlli di qualità o ci sono solo alert senza scarto verrà portato tutto il contenuto del file, se ci sono record che falliscono controlli di tipo alert con scarto questi record NON verranno portati in staging *-da capire se ci saranno questo tipo di regole in idqs*) Il nome del file in staging sarà lo stesso dell'input ma senza la versione (in caso viene ricaricato un file per quel flusso deve essere quindi sovrascritto), mentre invece verrà storicizzato nell'history *h_staging/prj_idqs/YYYY/MM/DD/file_parquet* il file di staging (quindi in parquet e con eventuali record filtrati), mantenendo però la versione del file di input (quindi stesso naming del file di landing ma con il contenuto del file di staging e in formato parquet)
Nel caso in cui invece c'è anche un solo record che non supera una regola di tipo BLOCKING il file NON deve essere portato in staging ne storicizzato in h_staging
In questo caso verrà mandata una mail di blocking (*da capire il template e se riusciamo a dare info sui ko*), verrà rimosso il file dalla landing, verrà effettuata la scrittura sulle tabelle SQL Server T_GEN_MONITORING e T_GEN_SCORECARD_EXECUTION (*da aggiungere allegato con mappatura campi scrittura output*) e terminerà il processo.

3. Reference, scrittura su db e mail finale

Se NON ci sono stati ko blocking, una volta portato il file in staging e storicizzato in h-staging, il file verrà copiato in reference reference/prj_idqs/ (*da capire se semplice copia parquet in overwrite o delta lake e in base a questo se va fatta o meno la storicizzazione in h_reference*) una volta effettuata la copia (non sono previste attualmente altre operazioni/trasformazioni sui file da portare in reference) verrà mandata una mail di success specificando se il file è andato tutto in success o se è terminato correttamente ma c'è stato qualche alert con o senza scarto (*da capire il template e se riusciamo a dare info sui ko*). In questa fase verrà effettuata anche la scrittura sulle tabelle SQL Server T_GEN_MONITORING e T_GEN_SCORECARD_EXECUTION (*da aggiungere allegato con mappatura campi scrittura output*)

GESTIONE ERRORI

Se in qualsiasi punto del processo c'è un errore generico/infrastrutturale non gestito dalle precedenti casistiche, va fatta una insert sulla tabella SQL Server T_GEN_MONITORING per tracciare il run KO (*da aggiungere mappatura campi scrittura*) e viene mandata una mail di processo failed *(da capire se si può dare ed eventualmente come riportare un'indicazione su cosa ha generato il ko)*
Oltre ad inviare la mail va SEMPRE ripulito il file dalla landing.

GESTIONE FILE DI APPOGGIO SU DATALAKE

Se necessario creare file di appoggio sul datalake durante il processo vanno utilizzati i seguenti path:
technical/elaboration/landing/prj_idqs/
technical/elaboration/staging/prj_idqs/
technical/elaboration/reference/prj_idqs/
I path di technical vanno sempre ripuliti al termine del processo

SCHEDULAZIONE

Il processo dovrà partire alla presenza di un file nel path di landing, se presenti più file li elaborerà sequenzialmente in ordine di caricamento (dal più vecchio al più recente)
(la schedulazione ad evento su synapse come funziona? posso controllare i primi n giorni del mese? se caricano a n+1 quando la schedulazione è chiusa o va giù synapse quando caricano il file in landing che succede quando riattivo la schedulazione? riparte da solo a ricaricare in ordine o vanno ricaricati i file in landing? In caso da capire come gestire)
Attualmente dovremmo caricare un file alla volta, da confermare con Luciano se sarà così o se potranno essere previsti caricamenti di file in parallelo .

4. Popolamento Data Model

TBD (*finire analisi con Luciano e definire se sarà un etl separato con una schedulazione ad hoc*)

Configurazione Database *(questa sezione è quella attualmente del processo di Informatica, seguirà quella per IDQS)*

Schema EDCPORTAL

- FIELDS (una riga per stream progettuale es. IFRS9, IFRS17)
- FLOWS (una riga per singolo flusso, FK verso FIELDS)

Schema EDCDB

- T_GEN_ANAGRAFICA_MACRO_SCORECARD (1 riga per scorecard)
- T_GEN_ANAGRAFICA_SCORECARD (1 riga per scorecard/domain, FLOW_ID = NAME di FLOWS, FK verso T_GEN_ANAGRAFICA_MACRO_SCORECARD)

- T_GEN_rule_scorecard (anagrafica delle regole di qualità, una riga per ogni regola di ogni macro_scorecard, FK logica verso T_GEN_ANAGRAFICA_MACRO_SCORECARD)
- T_GEN_SEND_EMAIL (1 riga per chiave flusso/domain, contiene i destinatari delle mail di quel flusso)
- T_GEN_ SCHEDULATORE (1 riga per flusso, NOME_FLUSSO = NAME di FLOWS)