Løsningsdokumentation for Picarro i Data Estate

1. Generelt om projektet/forretningen

Projektet udspringer af et behov fra Husdyr, om hurtigere indlæsning af data. Deres teknikere vil gerne kunne teste, at data kommer ind i databasen som forventet allerede under opsætning ude i stalden, og ikke som hidtil, hvor de kun indlæses en gang i døgnet. Samtidig er der ønske om enklere opsætning af nye picarro-maskiner i Data Estate.

Ved at omlægge behandlingen af de rå filer fra SQL DB til Data Estate og loade nye filer inkrementelt opnås hurtigere behandling. Opsætningen af nye maskiner i Data Estate kræver ikke længere oprettelse af individuelle tabeller og stored procedures, og er dermed enklere - dataen behandles nu automatisk når blot filer fra en ny Picarro-maskine lander i landing-mappen.

På Picarro-maskinerne lokalt ligger PicarroDotNet-applikationen, og sørger for at flytte filerne til DE landing. Den vil også fremadrettet være en del af dataflowet, men er ikke en del af Data Estate. Der er ønsker om at data bliver opdateret oftere, hvilket bl.a. vil kræve justeringer i DotNet applikationen, som sandsynligvis skal sættes op til at trigger filoverførsel, når der kommer data ind I de nuværende mapper på Picarro-maskinerne - eller som kan startes manuelt ved behov for at teste opsætningen.

2. Løsningsbeskrivelse

2.1 Løsningsdetaljer

- **Frekvens af rå fil uploads:** hver picarro-maskine skriver én fil i timen. De uploades i batches til landing én gang i døgnet.
- **tidspunkt for upload til landing:** "At 4:00 AM every day" via task scheduler på den enkelte maskine.
- **Størrelse af rå filer:** som regel mellem 200 kb og 5 mb, men enkelte maskiner med mange kolonner har filer på over 12 mb.
- **Lokation for rå filer:** https://adlsdataestateprod.blob.core.windows.net/landing/picarro/data/"maskinenavn"
- **Filtype for rå filer:** .dat-filer i tekstformat (ikke kommasepareret). Hver kolonne er 26 tegn, hvilket kræver ekstra operationer i opsplitningen af data i kolonner.

2.2 Løsning

- PicarroDotNet-applikationen er en del af dataflowet. Den ligger lokalt på maskinerne, og uploader til DataEstate landing.
- Filerne i landing bliver loadet til raw vha. "load_to_raw_picarro"-notebooken, og herefter cleanes og transformeres data i "transform_enriched_picarro"-notebooken. Som sidste step, benyttes "load_to_integration_picarro_view" til at danne to views for hver picarro-maskine - et integration view for alle kolonner og et view for aggregeringer og udvalgte kolonner. De endelige views udstilles i integration-laget.
- Fordi mængden af data er så stor, loades data inkrementelt i lagene (vha. autoloader i raw-laget og vha. MaxDwCreatedDate i enriched-laget).

2.3 Notebooks

- load_to_raw_picarro
- transform_enriched_picarro
- load_to_integration_picarro_view

3 Beskrivelse af notebooks (raw->enriched->integration)

3.1 Load to raw

- Denne notebook loader den rå data og skiller header og data i hver sin dataframe, der først samles i enriched-laget. Header-rækker og data-rækker identificeres ved brug af en condition. Time_stamp og filnavn tilføjes data ud fra filnavnet med brug af regex.
- Udfordringen i raw-laget ligger i, at de rå filer er simple tekstfiler, hvor hver kolonne består af 26 tegn, uden separator. Derfor kræves manuel opsplitning vha. substring() og split(), for at kunne danne headers og kolonner. Desuden loades filerne med autoloader, hvilket begrænser brugen af fx .select eller .first til at splitte data.

3.2 Transform enriched

- Denne notebook cleaner og transformerer målinger for hver enkelt picarro-maskine. Der defineres måleperioder "PeriodeID" og beregnes gennemsnitlige måleværdier for disse perioder.
- Notebooken både loader, cleaner og transformerer data i et stort loop, der kører for hver enkelt picarro-maskine. Loopet kører for hvert systemnavn/maskine over en liste af alle systemnavne. Loopet er for overskuelighed inddelt i fem sektioner:
    1. **Load data**: data og headers fra raw-laget samles i dataframen df_combined
    2. **Clean data**: datatyper og dubletter håndteres

3. **Transformation**: kolonner tilføjes til at assistere i aggregering af data
4. **Transformation**: aggregering af data
5. **Write data**: to dataframes, df_combined and df_combined_agg, skrives til catalog og storage

Load data

- MaxDwCreatedDate bruges til at finde den seneste række i enriched-laget. Det gøres ved at finde den maksimale dw_created_date i tabellen. Hvis der ikke findes nogen datoer, bruges en standardværdi. Data nyere end MaxDwCreatedDate betragtes som ny data og indlæses fra raw-laget.
- Data og header loades som to separate dataframes fra raw og kombineres til df_combined. Bemærk at dw_created_date tilføjes manuelt i både headers og i selektionen af kolonnerne i data.

Clean data

- Dubletter i dataframen omdøbes med funktionen rename_duplicate_columns.
- Datatyper castes vha en dictionary med kolonnenavn som key og datatype som value. Alle kolonner, der ikke tager decimalType(38,24), er defineret heri, mens øvrige kolonner castes som decimalType(38,24).
- "MPVPosition" er picarro-maskinens måleposition. Efter aftale med Simon (SWYG) bliver rækker hvor MPVPosition ikke er et heltal IKKE filtreret fra.

Transformation: columns to assist in aggregations

- Vinduer specificeres over dataframen og nye kolonner og tilføjes. De vigtigste kolonner at forstå er:
- "PeriodeID" bruges til at identificere og gruppere sammenhængende rækker, der har samme MPVPosition, og som er sorteret efter DateTimeStamp. Laves ved en sum() over periodestart, så PeriodeID vil stige 1,2,3 etc.
- "Markering" markerer "1" første og og sidste række af ny data i df_combined. Det hjælper med at markere hvad der hører til et batch af data, når det skrives til databasen.
- "TidTilNaesteKanalSkift" og "TidTilSidsteKanalSkift" markerer afstand til kanalskifte inden for en periode. Kolonnerne bruges til at filtrere målinger, der ligger tæt på et kanalskifte fra, da der er risiko for upræcise målinger lige før og efter picarro-maskinen skifter MPVPosition/kanal.

Transformation: aggregering af data

- Der oprettes en dictionary for aggregeringsfunktioner med kolonnenavne som key og aggregering som value. Derefter konstrueres en SQL select-statement, der anvender aggregeringerne grupperet på "PeriodeID". Rækker der er for tæt på kanalskifte filtreres fra i where-clause. Outputtet fra fx "g2509_agg" har således rækker for gennemsnitlige måleværdier for hver periode for en given maskine.

Write data

- Der skrives en aggregeret dataframe og en ikke aggregeret dataframe for hver maskine til brug i integrationslaget. SCD1 anvendes.

3.3 Integration

- Med funktionen CreatePicarroViews oprettes et aggregeret og ikke aggregeret view baseret på df_combined og df_combined_agg fra enriched-laget. Funktionen looper over en liste af systemnavne.

4 Workflows

- "Master picarro"

5 Tilhørende funktioner

Funktionen rename_duplicate_columns kaldes i enriched-laget som del af cleaning af data. Den omdøber dublet kolonnenavne med suffikset _1, _2 etc. Funktionen looper over dataframens kolonnenavne og tæller i dictionaryen col_count hvor mange gange hvert navn er set. Første forekomst omdøbes ikke (ALARM_STATUS), men de næste omdøbes (ALARM_STATUS_1, _2, etc.) Bemærk at loopet itererer over kolonnenavnene i lower case (column.lower()), for at håndtere inkonsekvent brug af case i den rå data.