Here I explain en línies generals i resumidament el que el codi fa. No es una explicació detallada com la que hauria de ser (del estil API documentation). Però pretén servir com a resum informal del codi i la feina feta durant aquestes últimes setmanes.

El codi per cada una de les funcionalitat esta separat en diferents packages. Per altre banda tenim el fitxer control.properties que m'ajuda a fer el set de valors utilitzats comunitariament per les diferents funcionalitats. (e.g., extraction times from one specific data source). En general alguns noms s'haurien de modificar per poderlos entendre millor i fer a l'usuari del codi més entendibles tot.

Sprint 1. This sprint is related to Objective 1. Having a proper data synchronization between the master repository and WIMEDS, Administration Units exportation.

Work done:
El codi d'aquesta part esta a *HBaseToWIMEDSDB* package inside the Github repository. Aquí hi ha el codi que implementa la extracció de la taula "WHO-DEV-OrgUnits" d'HBase i fa la transformació de les dades per introduir-les en una taula de Postgres que simula la WIMEDS database. Aqui hi ha el codi que implementa el connector for reading data from WICD (HBase) and exporting to WIMEDS (PostgreSQL) juntament amb el codi for updating the data that conflicts with the master data stored in HBase.

Els fitxers que hi han dins es poden dividir en les fases full d'extracció i de update de les dades.
Per la part full d'extracció tenim els fitxers: *HBase_orgUnitExtraction.java, organisationUnitsExtraction.java*
Per la part d'update de les dades tenim els fitxers: *updateDB_orgUnits.java*, *organisationUnitsUpdate.java*

a) En la fase d'extracció de la taula de d'administration Unit de HBase cap a Postgres els passos són:

- creem scanner per a la taula HBase
- utilitzem el scanner per recórrer i extreure les dades de la taula HBase
- per cada batch retornat per l'scanner codifiquem les dades per tal d'introduïrles a Postgres en forma de SQLQuery.
- un cop l'scanner ha acabat de recorrer la taula, eliminem el scanner i actualitzem els temps d'extracció (*thisExtraction, prevExtraction).*
- fem load de les dades a Postgres.

b) La fase de update s'assembla molt a la de extracció. L'únic que canvia es la manera en com es defineix el scanner, i òbviament les SQLQuery que es genera a partir de les dades.

- creem scanner amb un batch, startTime, endTime.
- utilitzem el scanner per recórrer i extreure les dades de la taula HBase que s'han modificat desde la ultima extracció.
- per cada batch retornat per l'scanner codifiquem les dades per tal de fer update a la taula AdministrationUnit de Postgres.
- un cop l'scanner ha acabat de recorrer la taula, eliminem el scanner i actualitzem els temps d'extracció (*thisExtraction, prevExtraction).*

- fem update de les dades a Postgres

Valors del ficher *control.properties* utilitzats en aquestes dues funcionalitats:

PostgresSQL(WIMEDS): *DBurl, DBusr, DBpwd*
HBase(WICD-landing zone): *url, batch*
Els valors *thisExtraction* i *prevExtraction* guardats a *control.properties* són compartits per les operacions de full extraction i la de update.

Possible improvements:

- //estaria bé com a millora guardar per separat els ultim temps d'extracció i l'ultim temps de update. Per si en algun moment és necesita saber aquesta informació. Ja que unes es overwrite a les altres.
- //pensar una manera fàcil de parametritzar els attributes que es volen per a tenir a la taula Administration Units de WIMEDS. De moment es fan servir només els següents: *parentid, name, shortname, datelastupdated, leaf, levelnumber i url.*
- //fer que la crida a HBase REST API em retorni un json i no un xml (m'estalviaria codi)

Sprint 2 .This sprint is related to Objective 2. Loading application data to the data lake.

Work done:
El codi d'aquesta part esta a *WIMEDSbonitaToHBase* package inside the Github repository. Els fitxers java són els seguents: *WIMEDSextraction.java, IndividualRequestExtraction.java, HbaseUtilities.java, ReqMedProcessMetadataExtraction, BonitaRESTAuth.java.*

El codi implementat exporta data related to *REQ_RequestMedicines* process from WIMEDS to the WICD landing zone. En concret el codi exporta les dades de tables (names from WIMEDS tables):
Request, ShipmentR, RequestDocument, Disease, RequestStatus, MedicalSupply and *Manufacturer.*

These are the steps the code follows in order to export WIMEDS *REQ_RequestMedicines* process data to HBase:

a) create the *WIMEDS-Table-Data* in HBase if it has not been created yet.
b) ask Bonita REST API for *REQ_RequestMedicines* process metadata last update date.
b.1) if (*last update date* > l*ast WIMEDS data extraction)* then update metadata version (for the row key). No WIMEDS metadata is stored in HBase at this moment. If this happens a new metadata extraction would be needed.
b.2) else nothing has to be done with respect to metadata
c) export all the data from the WIMEDS tables mentioned before
d) update last WIMEDS extraction time

Valors del ficher *control.properties* utilitzats en aquestes dues funcionalitats are the ones commented as *--HBase_RequestMedicineProcess* and *--BonitaWIMEDS*.

Possible improvements:

- Dir el tema de que no esta implementat un update. Cada cop que es vol actualitzar les dades de HBase de la taula WIMEDS-Table-Data un full extraction de les dades de WIMEDS es necessária, that is, another execution of WIMEDSextraction.java code would be done. S'extreuen les noves dades i es fa store d'aquestas under a new row key. which has the format:
  Subsystem$Individualdata$process$metadataVersion$extractionDateTime
  WIMEDS$Individual$RequestMedicine$1$2020-12-29T16:59:55.789
  Seria bó trobar una manera de fer un update eficient i no una full extraction.
- parametrització de les taules a extreure de WIMEDS. Per si en cas de que es vulguin exportar més no sigui necessari accedir al codi.
- tenir en compte que hi haurán taules (e.g., Request) que poden arribar a creixer molt i s'hauria d'aplicar una tècnica per a distribuir millor les dades. As it is done for some WIDP case (blocks). No se si la tècnica de guardar-ho tot sota la mateix row i columna es la òptima.
- Store *REQ_RequestMedicines* process metadata

To correct, bug:
Entre els pasos a) i b) hi hauria d'haver uncommented la part del codi implementada per *BonitaRESTAuth.java* en el que s'estableixen el jsessionid i el X-Bonita-API-Token per les següents crides. Aquest codi esta comentat per que no he sigut capaç de que amb codi java ….explicar.... Ara ho faig amb el Postman i cada cop que ...bla bla…

Sprint 3: This sprint is related to Objective 3. The focus will be on data pre-processing and transformation of the data lake data.

Work done:
El codi d'aquesta part esta a *HBaseLZtoPostgresTZ* package inside the Github repository. Els fitxers java són els seguents: *mainToTransformedZone.java, organisationUnitsExtractionToTZ.java, toTransformedZone.java, tz_mainUpdate.java, tz_updateAdminUnit.java* and *PostgresSqlTransformedZone.java*

Explicar quines taules tinc, dimension tables i fact table. measures sobre fact table.
pero bàsicament aquí es parla de fer el schema, i com faig el ETL process.
Explico el codi i ja esta…
dir que adjunto també els dos fitxers .txt amb els create tables.

The java code implements the ETL steps for loading the data from the WICD landing zone (HBase) to the modeled WICD transformed zone (PostgreSQL). The main steps are:

a) create Tables and Materialized Views in Postgres transformed zone if needed
b) perform Administration Units extraction from HBase landing zone table and export it to transformed zone DB (ETL process)

c) extract data from WIMEDS-Table-Data in landing zone and export it to transformed zone DB (ETL process)
d) set extraction times
e) populate medicalSupply_manufacturer table (it is populated with data from medicalSupply and manufacturer table)
f) refresh materialized views

Step d) is basically needed for updating AdministrationUnits table in the transformed zone since the logic used is the same as the one used in *HBaseToWIMEDSDB.*

The schema structure modelled for multidimensional analysis is:
FER SCHEMA en UML sencillet només de 1-* i així.

Requests table is the fact table where the measures come from. Other tables are used as dimension tables.
ShipmentR table is there but still not used by the Tableau (which uses materialized views).

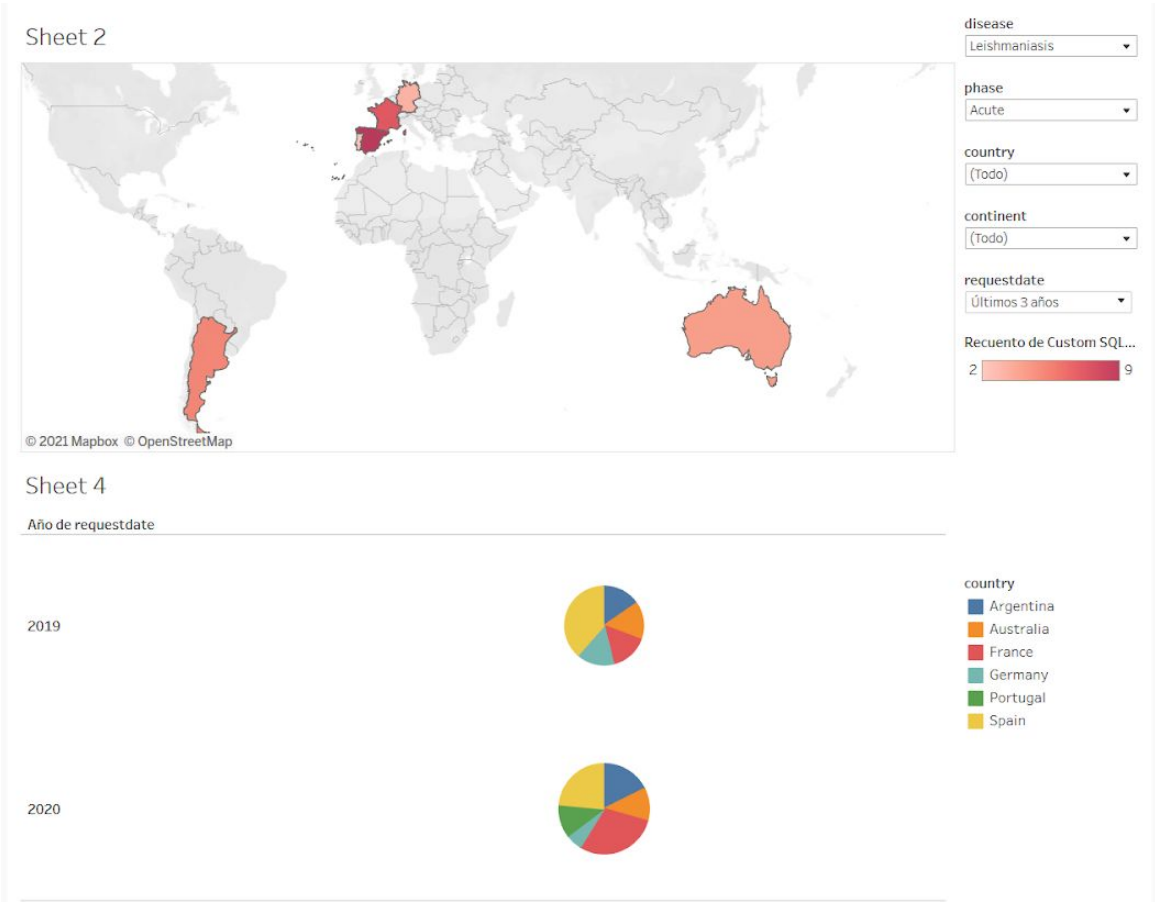
Possible improvements:

- A millorar seria mirar el dateTime de la rowKey i compararla amb la ultima. Si no s'ha de fer actualització no es fa. Si s'ha de fer es fa una full extraction.
  Estaria bé trobar una manera per tal de només fer una extraction d'aquelles columns(WIMEDS tables) que hagin sigut modificades.
- parametritzar les dades de les que es vol fer extraction, that is WIMEDS Tables stored in HBase as columns under the same row key.
- only refresh materialized views if needed. Now it is always done since always the code is executed, an update into the transformed zone is done (it would be not needed if data in the landing zone has not changed since last update).
- canviar com es fan les taules, necessito el FK statement…
- create materialized view with shipmentR data to enable further analysis over shipments data.

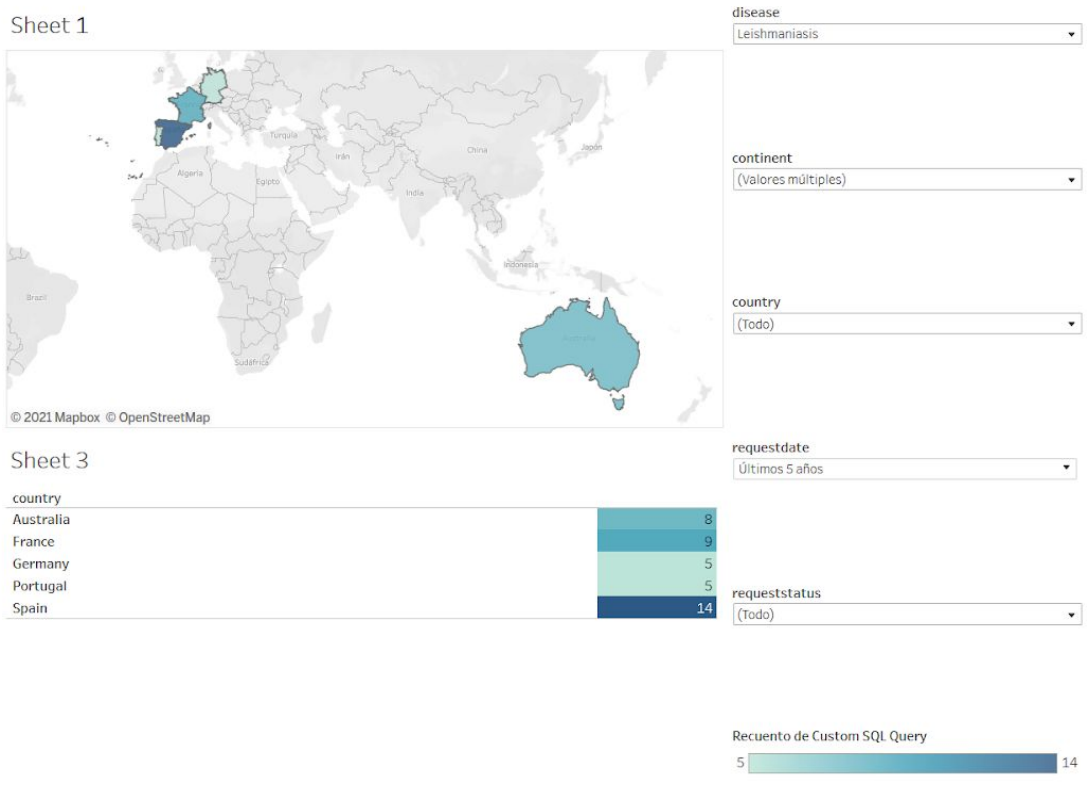
Sprint 4: This sprint is related to Objective 4. The focus will be on bringing descriptive analysis over WIMEDS data to end users. The sprint is divided into the next tasks:

i Perform a state-of-the-art of different OLAP tools
ii Perform a SWAT analysis and select an OLAP tool
iii Design and develop a dashboard using the OLAP tool selected
iv Perform multidimensional analysis and reporting for WIMEDS/WIDP data

Aquesta part s'ha portat a terme amb l'ús de Tableau. Els Dashboards y Sheets estan publicats en el meu personal Tableau Online. Aquí algunes figures dels Dashboards creats a partir de les materialized views de la transformed zone:



Dashboard 1



Dashboard 2


Sheet 1
Recuento de Custom SQL Query
patient_age
continent
(Todo)
country
(Todo)
disease
Leishmaniasis
phase
Acute
Sheet 3
Recuento de Custom SQL Query
patient_weigthinkg

Dashboard 3

Dashboard 4

Dashboard 5

Per tal de que em sortissin aquestes dades que es poden veure als dashboards únicament he fet els inserts del fitxer insertsTestsTableau.txt per tal de veure bé el funcionament de Tableau. (Més ràpid que fent-los desde la aplicación de WIMEDS).

Possible improvements:

- add analysis for Shipments