Problemas:

* Codigo espagetti
* Comentarios dentro del mismo codigo (Poner ejemplo).
* Codigo que funcionaba anteriormente es imposible que el desarrollador pueda a volvear a obtenerlo porque fue sobrescrito.
* Comparar diferentes versiones del mismo código para determinar el error.
* Por la naturaleza de la data que se actualiza constantemente, en un momento se obtienen los datos a partir de una versión de la data y luego al pasar el tiempo (meses, dias) no se obtienen datos.
  + Esto hace dudar de la efectividad del trabajo del QA y del desarrollador.
* Codigo en diferentes formatos (letras pequeñas, grandes, notación encamellada, etc) y estilos que hacen muy difícil la comprensión del mismo por otros desarrolladores.
* Una curva de aprendizaje enorme.
* Algoritmos (Script de BigQuery) sin documentación (White Paper) lo que no permite estudiar o conocer la lógica del algoritmo.

**Propuesta:**

Usar git y github como sistema y repositorio e control de versiones.

* Utilizar github actions como sistema de Continous Integration / Continous Deployment (CI/CD) para la validación y chequeo del código fuente de los algoritmos.
* Propuesta de jerarquía de directorios y código fuente:

PORA

README.md

<general\_docs>

sql\_code\_style.md

<esquema\_name\_in\_BigQuery>

<algorithm\_name\_1>

algorithm\_name.sql

<WP>

algorithm\_name\_idAlgoritm\_wp.docx

<TABLES>

finalPool\_v2.csv

<OUTPUTS>

result\_algorithm\_name.csv

…..

<algorithm\_name\_n>

**Rama master en github:**

Cada issue del Jora esta asociado a una rama de nombre #isssue\_nombre del algoritmo que de master la estructura principal y el resto del código.

* Al desarrollar un algoritmo siempre que se haga un commit en el texto se debe color al final el #issue asociado al Jira para que quede comentado cualquier avance y sea fácil relacionar y acceder desde el Jira.
* Para desarrollar un algoritmo nuevo o arreglar un bugs de alguno ya creado:
  + Respectar las normas de code style para SQL.
  + Seguir la estructura en el árbol de directorios propuesta anteriormente.
  + Cuando se termine la realización del algoritmo se debe crear un pull request contra la rama master.
  + En el punto anterior si el proceso de QA a finalizado entonces se aprueba el PR, no sin antes revisar el cumplimiento del code style y otros requerimientos.
  + Mientras se encuentra en el proceso el PR el código fuente de un algoritmo un team developer lead o personal designado revisara el código y chequeara el cumplimiento con el code style y podrá hacer sugerencias al desarrollador.
  + Solo un personal autorizado o con privilegios hará la mezcla sobre master (en caso que se decida pueda ser el mismo desarrollador por el nivel de confianza, aunque no se sugiere).
* Realizar un push sobre la rama en la que se desarrolla el código fuente de un algoritmo.
  + Si se tienen tablas de pruebas se debe crear en el esquema de pruebas.
  + Crear o actualizar el algoritmo en el esquema de pruebas.
  + Ejecutar el algoritmo para validar que el código este sintácticamente correcto.
  + Obtener la salida para crear Sample\_File o Final File
  + Salvar la salida en archivos de tipo csv de tipo Sample\_File