29-11-2024

Captura y Preparación de Datos.

PEC3

Juan Luis Acebal Rico
GRADO DE CIENCIA DE DATOS APLICADA

Indice

Parte teórica

Q1.

En el proceso de integración, observamos que hay diferentes técnicas para lidiar con los registros, como es el caso de la fusión. Mediante la fusión se permite gestionar la evolución del esquema del modelo de datos, ¿por qué es tan importante manejar correctamente dicho esquema? A día de hoy existen herramientas que te permiten definir el esquema del modelo a integrar y otras que lo evolucionan de modo automático. ¿Cuál preferirías tú y por qué?

Manejar bien el esquema del modelo de datos es imprescindible por muchas razones, quizás las que encuentro yo más importantes son, la consistencia de datos, prevención de errores, y cambios controlados:

Consistencia de los datos: Crear un sistema bien gestionado requiere que los datos sean consistentes, estén alineados con reglas (por ejemplo, de protección de datos, o seguridad de datos secretos, como contraseñas y números de cuenta o tarjeta) y que tengan restricciones definidas y conocidas. Esto es como se debe de hacer ya que aseguras la integridad referencial, y además es necesario para ser consumido por las aplicaciones que consumen estos datos y que necesitan ser preparadas para cada cambio en los datos, incluso cuando estos son pequeños, tales cambiar un INT por un VARCHAR.

Cambios controlados: A medida que las necesidades del negocio cambian, las tecnologías y casos de uso también. Gestionar esta evolución de forma eficiente hace que controlar su evolución sea necesario para no perjudicar a otras partes de negocio o sistemas existentes.

Prevención de errores: Un manejo incorrecto del sistema puede producir pérdidas de datos, duplicidades, o incompatibilidades de algún tipo, que son muy costosas de solucionar e incluso a veces genera una pérdida de datos importante.

De las herramientas disponibles, yo preferiría definir el modelo e integrar de forma manual, con scripts en bash o Python(que pueden usarse tanto onPremise con Airflow como en cloud, en el servicio Azure Managed Airflow, entre muchos que existen, que ya podríamos hablar de otras muchas formas como montar un contenedor), o onPremise herramientas como Pentaho data integration, también me parece interesante en cloud Databrics, Azure Data Factory o Azure Synapse Analytics.

El motivo es que yo lo haría de forma manual, es que tienes un control total de los datos, tienes una mayor trazabilidad, puedes documentar mientras los integras o migras los datos, y el equipo aprende el esquema de los datos mientras lo hace. Además, hacerlo de forma automática puede crear cambios no deseados que puede costar mucho arreglar.

También me parece muy interesante el nuevo paradigma de zero-ETL, que tal y se comenta en el artículo, tiene muchas ventajas, aunque tiene también algunos inconvenientes, tales como la

dependencia de la nube, una curva de aprendizaje más pronunciada o que la solución de problemas es más complicada, tiene por otro lado análisis en tiempo real, o la racionalización de la ingeniería, es decir, uso eficiente de recursos. Aunque me considero un cloud avocate, me genera inquietud depender de más en más de 3 compañías para tantos servicios y necesidades básicas.

Q2.

Tengo dos tablas a integrar en mi sistema de analítica avanzada, una es la cabecera de la factura y la segunda el detalle. Ambas se relacionan 1:1 mediante el Identificador único FacturaId. Actualmente el proceso de captura de datos se está ejecutando en modo "Full" (nos traemos todo el conjunto de registros de ambas tablas) pero queremos optimizar el proceso y cambiar a modo "Delta" (reducir el conjunto de registros a extraer a únicamente los nuevos y/o los actualizados) para ello, observamos que la primera tabla (cabecera de factura) dispone del atributo UPDATED_ON que nos permite identificar aquellos registros que han sido creados o actualizados, si bien la segunda tabla (detalle de factura) no dispone de dicho atributo. ¿Qué técnica de selección emplearías para cargar ambas tablas en modo Delta? Y si no existiera ese campo (UPDATED_ON) en ninguna de las tablas, pero la lógica de gestión de facturas cambiara y ahora ninguna de ellas se pudiera actualizar, ¿Cómo harías para recoger los nuevos registros en modo Delta?

Para la primera parte, usaría seleccionar el último updated de la tabla de destino, y MAX(UPDATED_ON) y de ahí introducir todos los registros que tienen UPDATED_ON>UltimaCarga o también, UPDATED_ON(origen)>UPDATED_ON(destino).

Teniendo los FacturaId identificados a actualizar o insertar, ya se podrían eliminar-insertar, o hacer un MERGE en la tabla de destino, y la tabla de detalles igual.

Por ejemplo:

```
WITH Ultima_actualizacion AS (SELECT MAX(UPDATED_ON) AS MaxUPDATED_ON FROM CabeceraOrigenFactura)

WITH Facturas_Nuevas_Actualizadas AS (

   SELECT FacturaId FROM CabeceraFactura

   WHERE UPDATED_ON > (SELECT MaxUPDATED_ON  FROM Ultima_actualizacion)

);
```

```
SELECT COUNT(*) FROM CabeceraFactura

WHERE FacturaId IN (SELECT FacturaId FROM Facturas_Nuevas_Actualizadas);


SELECT COUNT(*)  FROM DetalleFactura -- Al hacer count veo si es coherente con la Cabecera

WHERE FacturaId IN (SELECT FacturaId FROM Facturas_Nuevas_Actualizadas);
```

Ahora podría hacer un DELETE-INSERT (Usando una transacción, y con commit y rollback) o un MERGE.

Respecto a la segunda parte, hay varias opciones. Al tener UPDATED_ON en la cabecera, podemos hacer un select de los que se han modificado desde la última modificación, usando como key la FacturaId para que alcance a la segunda tabla.

Si no tenemos UPDATED_ON en ningún lugar, solamente podríamos insertar nuevos registros, no podríamos modificar los existentes, tomando el máximo FacturaId cargado o última fecha cargada, aunque aquí podemos apreciar varias cosas, ya que existen multitud de estrategias para gestionarlo:

¿Cuál es el periodo de modificación actual de la factura? Si la factura ya no se modifica, por ejemplo, pasados 10 días, podemos hacer un delete & insert de todas las facturas de los últimos 10 días, como es una plataforma de analítica, no tiene que ser en tiempo real, puede hacerse este cambio por las noches.

Otra estrategia, un rango de fechas (del detalle de la factura, ya que todas las facturas tienen fecha), usando un delete en la tabla de destino y un insert a la tabla de destino con un select(tabla origen) con una área de staging.

Hay varias opciones más, tales como utilizar aparte del FacturaId, usar el historial de modificaciones en la BD, tales como CHANGE_TRACKING, CHANGE_RETENTION, etc (numerosas opciones que tienen las bases de datos modernas para almacenar un historial de cambios)

Aunque no es una práctica tan recomendada, yo en mi trabajo tenemos un DAG que se ejecuta para los últimos 40 días, y otro para los últimos 3, para ciertas tablas, y que hace update en SnowFlake todas las tablas que hay desde Oracle (Tenemos ambas BD, que se está haciendo una integración que dura ya 3 años). Es un DAG para cuando falla algún proceso ETL de alguna tabla concreta, no se usa de base para migrar o integrar en si los datos.

Q3.

Hace unos días, el equipo de Científicos de Datos terminó de construir un modelo de clasificación binario y han decidido medir su rendimiento para decidir si lo pasan a producción o iteran de nuevo sobre él. Sabiendo que el coste por Falso Negativo multiplica por 4 el coste de un Falso Positivo, ¿qué medida contemplarías como la más idónea para evaluar el modelo? (justifícalo)

Nos interesa que tenga una buena predictibilidad, para ello hay una serie de métricas, que lo miden, sin embargo, no nos interesa solamente la métrica que mide todo, que es la exactitud (accuracy), la cual es muy sencilla, simplemente es el total de casos positivos entre el total de casos, ya que en esa métrica tiene en cuenta los falsos positivos y los falsos negativos, nos interesa una métrica que tenga en cuenta antes los falsos negativos 4 veces más que los falsos positivos, es decir, los TIPO II vs los TIPO I, por tanto, usaríamos la fórmula de la sensibilidad o recall:

$$Sensibilidad = \frac{VP}{P}$$

Y queremos darle menos importancia a la precisión, que es:

$$Precision = \frac{VP}{VP + FP}$$

Si vemos F1-score, que es, una media armónica:

$$F1 = 2\,x\,\frac{Precision\ x\ Sensibilidad}{Precision + Sensibilidad}$$

Podemos adaptarlo para que tenga más peso la sensibilidad:

$$F2 = (1 + \beta^2)\,x\,\frac{Precision\ x\ Sensibilidad}{\beta^2\ x\ Precision + Sensibilidad}$$

Si usamos $\beta^2 = 4$, es decir $\beta = 2$, que es el peso relativo de la sensibilidad respecto a la precisión. En este caso, 4 significa 4 veces más. Si Queremos otro valor de β, para darle más importancia a los falsos negativos, tendríamos que usar$\beta^2 = k$, donde k seria la proporción que quiero usar de precisión respecto a sensibilidad. Si es 7 veces más, seria $\beta^2 = 7$, etc., entonces si fuera 7, tendría que poner 7 donde β^2

Y también, al contrario, que sería, si quiero que la precisión tenga 3 veces más de peso:

$$\frac{Peso\ precision}{Peso\ recall} = k\ ;\ \frac{1}{3} = k\ \ \beta^2 = \frac{1}{3}\ ;$$

Dicho todo esto, hay que ver en general todas las métricas, y si bien F2 sería la principal en este caso, no sería la única a observar. Tenemos que combinarla con F1 para que el la diferencia entre ambas no sea muy grande en favor de la F2, viendo el contexto, con la curva ROC-AUC e incluso creando una métrica personalizada.