Ejercicio de Gestión

Índice

- **Objetivo**
- **Diagnóstico del Problema**
- **Plan de Optimización**
    - Enfoque General
    - Acciones Clave
    - Ejecución por Fases
- **Propuesta de Arquitectura Moderna**
    - Infraestructura Tecnológica
    - Etapas de Migración

Objetivo

Diseñar estrategia de optimización para pipeline con tiempo y recursos disponibles.

Diagnóstico del Problema

El pipeline en cuestión alimenta reportes diarios para varias áreas del negocio y enfrenta fallas esporádicas debido a:

- **Latencia** en la respuesta de algunas de las múltiples fuentes de datos que consume.
- **Calidad de los datos.**
- **Intervención Humana:** Falta de automatización en la detección y resolución de errores provoca un obligado monitoreo humano que consume tiempo de los recursos del equipo.

Estas fallas provocan **retrasos de entrega de información clave** y oportuna para la toma de decisiones, **pérdida de confianza del usuario** a la información presentada con **riesgo** de buscar o generar otras soluciones no escalables como los ‘data shadow systems’ con el propósito de satisfacer su demanda operacional y **la sobreutilización y des orquestación del equipo para ‘apagar fuegos’** cuando se encuentra enfocado en ofrecer valor a otras áreas claves del negocio.

Plan de Optimización

Enfoque general

Se aplica una estrategia de **prioridad alta y bajo esfuerzo** para lograr mejoras rápidas sin desviar significativamente al equipo de operaciones críticas del negocio. Esta práctica está alineada con técnicas de priorización como el impact-effort map, ampliamente usadas en ingeniería y gestión de proyectos Branching Minds+2Fibery+2.

Acciones Clave

Acción 1 (A1) : Validaciones tempranas en la ingesta (Automatizado)

- Implementar chequeos automáticos de esquema, tipos de datos y calidad desde los puntos de entrada.
- Objetivo: bloquear datos no conformes antes de que ingresen y evitar fallos que escalen a etapas posteriores del pipeline.
- Justificación: entre el **33 % y 35 % de los errores operativos** en pipelines provienen de inconsistencias en tipos o formatos de datos, especialmente en etapas iniciales XenossKanerika.

Acción 2 (A2): Retries automáticos con lógica de backoff exponencial (Automatizado)

- Incluir lógica de reintento automática ante errores temporales (por latencia o fallo de fuente), por ejemplo, hasta 3 intentos con backoff exponencial antes de fallar la ejecución.
- **Objetivo:** reducir fallos visibles y eliminar necesidad de reinicios manuales.
- Beneficio demostrado en producción para mejorar resiliencia con mínimo esfuerzo operativo DreamFactory Blog, previene sobrecarga de sistemas temporales, mejora la tasa de éxito en reintentos y reduce el riesgo de colapso por carga masiva.

Acción 3 (A3) : Alertas mínimas y accionables

- Usaremos **correo electrónico** para enviar alertas críticas configuradas desde *Google Cloud Monitoring*.
- Las alertas se diseñarán para ser **relevantes y claras**, evitando notificaciones innecesarias.
- Esto permite detectar problemas importantes sin necesidad de chequear manualmente todo el tiempo y evitar el agotamiento del equipo por exceso de alertas (alert fatigue)

Acción 4 (A4) : Logging estructurado y observabilidad básica

- **Registro de metadata por ejecución:** Incluye campos como ID del job, timestamp, paso ejecutado, número de registros procesados y mensaje de error.
- **Formato JSON para los logs, debido a que:**
    - **Equilibra legibilidad humana y máquina,** facilitando análisis y lectura rápida.
    - **Es compatible con herramientas de monitoreo** como Data Studio, que procesan estructuras clave-valor eficientemente.
    - **Permite flexibilidad y consistencia,** ya que es sencillo añadir o eliminar campos sin afectar sistemas existentes.
    - **Cuenta con soporte amplio en numerosos entornos y lenguajes,** siendo prácticamente universal para logging estructurado.
- **Objetivo:** Facilitar trazabilidad (correlación de eventos) y diagnóstico ágil ante fallos en pipelines complejos; esta observabilidad embebida acelera significativamente la resolución de incidentes **newrelic.combetterstack.com.**

Acción 5 (A5): Dashboard ligero (opcional)

- Crear un dashboard sencillo (Looker Studio o Data Studio) con métricas clave como número de errores, reintentos exitosos, latencia promedio.
- Proporciona visibilidad al negocio y facilita el monitoreo sin requerir intervención técnica constante.

Ejecución por fases

Fase	Acciones principales	Objetivo principal
1	Validaciones (A1) + alertas (A3)	Detectar errores frecuentes y mitigar impactos visibles
2	Retry automático (A2) + logging estructurado(A4)	Recuperación automática y diagnóstico más eficiente
3	Dashboard ligero (si hay tiempo) (A5)	Visibilidad al negocio, transparencia operativa sin costo

Este enfoque incremental permite soluciones rápidas en fases 1 y 2, mientras se sientan las bases para una gobernanza futura sin comprometer la capacidad del equipo.

Propuesta de Arquitectura Moderna

Infraestructura Tecnológica

Ingesta y Orquestación
Utiliza Cloud Composer (Airflow) para coordinar y ejecutar pipelines de manera eficiente, manejando dependencias y flujos ETL/ELT.

Almacenamiento y Gobernanza Centralizada
Emplea BigQuery para almacenar datos en bruto y Dataplex como catálogo inteligente que unifica descubrimiento de datos, clasificación, lineage, control de acceso y calidad automatizada. Esto promueve gobernanza contextual y automática en toda la plataforma. **Medium+7Airbyte+7Google Cloud+7Medium+2Airbyte+2Y42+12Google Cloud+12Google Cloud+12**

Modelado Analítico con dbt
Transformaciones en dbt sobre BigQuery, incorporando pruebas automáticas (dbt test), generación de documentación y lineage para asegurar calidad, transparencia y trazabilidad del modelo.

Control de Calidad Automatizado
Aprovecha Dataplex para ejecutar validaciones rule-based (como NonNull, Uniqueness) y perfiles de calidad automatizados, sin necesidad de implementar scripts manuales extra. **Reddit+3astrafy.io+3Medium+3dbt LabsMedium**

Versionado y Despliegue Simplificado
Confía en las capacidades integradas del IDE de dbt, que permite versionar, crear ramas, hacer commits y merges dentro del mismo entorno de desarrollo, manteniendo todo centralizado y sin complicaciones externas.

Etapas de Migración

1. Análisis del Pipeline Legado

- Documentar el pipeline actual: lógica, fuentes y transformaciones en Confluence.
- Identificar stakeholders, reportes afectados y dependencias.

- Detectar puntos críticos: latencia, baja calidad de datos, errores frecuentes.

2. Diseño y Prototipado con dbt

- Usar entornos controlados (Dev vs Prod) gestionados desde el IDE de dbt.
- Construcción gradual: raw_ → stg_ → mart_, con pruebas automáticas (dbt test), documentación y lineage en cada paso.
- Comparar resultados entre el modelo legado y el nuevo en ejecución paralela.
- Utilizar Slim CI, ejecutando solo los modelos modificados y sus dependencias, sin reconstruir todo el pipeline.
  Secoda+15dbt Developer Hub+15dbt Developer Hub+15

3. Gobernanza Progresiva con Dataplex

- Activar lineage automático, escaneo de calidad y trazabilidad en Dataplex.
- Implementar reglas de calidad: NonNull, Uniqueness, pruebas SQL personalizadas.
- Migrar gradualmente al catálogo universal para centralizar la gobernanza.

4. Control de Versiones y Despliegue

- Usar las funciones de versionado del IDE de dbt: commit, creación de ramas, pull requests y merges dentro del entorno de trabajo integrado.
  dbt Developer Hub

5. Monitoreo Continuo y Feedback

- Configurar alertas con GCP Monitoring para visibilidad operativa.
- Mantener tickets activos para monitorear y mejorar pipelines con feedback continuo de stakeholders.

6: Documentación y Transferencia (Handover)

- Elaboración de un documento de handover que consolide:
  - El estado actual del pipeline, riesgos y decisiones técnicas clave.
  - Pendientes críticos, accesos relevantes y contactos operativos.
  - Infraestructura utilizada (entornos, herramientas, repositorios).
- Transferencia gradual y tangible:

- Realizar sesiones revisando el documento junto al equipo receptor o stakeholders.
- Asegurar que quienes continúen con el proyecto reciban una “visibilidad total” desde el inicio.

Un artículo de Miquido lo resalta como una etapa clave en ingeniería para mantener calidad y alineación técnica a largo plazo https://www.miquido.com/blog/software-project-handover-checklist/. Además, incorporar un checklist estructurado —identificando responsables, fechas y canales de comunicación—está alineado a buenas prácticas de management técnico para asegurar una transición sin fricciones onsiter.medium.com.