Estandarización en la generación de logs para rutinas en Calendarización Analítica

**Autores**

Equipo Calendarización

**Fecha de última revisión**

13/06/2022

**Historial de Revisión y Aprobación**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Fecha  (DD/MM/YYYY) | Versión | Descripción/cambio realizado | Autor y/o Participantes | Instancia de Aprobación |
| 28/09/2018 | 1.0 | Creación del documento | Equipo LZ |  |
| 13/06/2022 | 2.0 | Nueva estrategia en la generación de logs para garantizar el funcionamiento y estabilidad de las rutinas | Equipo Calendarización |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

Tabla de contenido

[Objetivo: 4](#_Toc106026731)

[Conceptos: 4](#_Toc106026732)

[Log de compilación: 4](#_Toc106026733)

[Log de estabilidad 5](#_Toc106026734)

[Orquestador (Clase Logger) 5](#_Toc106026735)

[Logs 6](#_Toc106026736)

[Estructura 6](#_Toc106026737)

[Lectura 6](#_Toc106026738)

[Calendarización Nueva Rutina 6](#_Toc106026739)

[Modificación de Rutina 7](#_Toc106026740)

[Modificación significativa: 7](#_Toc106026741)

[Modificación no significativa: 7](#_Toc106026742)

[Modificación por incidente: 8](#_Toc106026743)

[Generador log de compilación 8](#_Toc106026744)

[Generador Log horario calendarización 8](#_Toc106026745)

[Revisión Log 8](#_Toc106026746)

[Errores Tolerables 8](#_Toc106026747)

[Errores No tolerables 9](#_Toc106026748)

[Revisor automático de logs 9](#_Toc106026749)

# Objetivo:

Describir el estándar para la generación de archivos que contienen la grabación secuencial de todos los pasos ejecutados en una rutina que busca ser desplegada en un ambiente productivo.

Todos los procesos descritos en este documento buscan facilitarle al usuario la generación de evidencias que garanticen que su rutina es estable y funcional.

# Conceptos:

## Log de compilación:

Un log de compilación tiene el objetivo de evidenciar el funcionamiento completo de una rutina. Con este se busca que un paquete que desea ser desplegado no presente incidentes en cuanto a:

* Dependencias de librerías
* Parámetros
* Fuentes no ingestadas
* Flujo de orquestador
* Funciones y/o scripts adicionales

Este log se caracteriza por ser de generación rápida, de manera que el usuario pueda tener un control más ágil de las configuraciones que desea utilizar para su rutina en producción. Para esto, el log de compilación NO se genera con el 100% de los datos que consumirá la rutina sino con una muestra que permita una ejecución rápida y completa.

**Nota:** Este log debe contener como prefijo “**cmp\_**”, de esta manera se podrá identificar que es un log de compilación.

**Condiciones de Generación**: Un log de compilación se genera bajo las siguientes condiciones:

* La rutina debe estar finalizada y lista para el proceso de calendarización.
* Identificar las principales fuentes de datos (fuentes crudas o zona de resultados) y parametrizar la rutina de manera que solo se extraiga una muestra de estos.
* Configurar los parámetros adicionales tal cual cómo la rutina se debe ejecutar en producción; por ejemplo, fechas, prefijos, zonas, etc.
* Ejecutar en ambientes acorde a los dispuestos para ejecutar las rutinas en producción; por ejemplo, versiones Python, Spark, etc.
* La ejecución debe ser completa tal cual como será en producción, es decir, datos, scripts y controles adicionales que desee en su proceso.

## Log de estabilidad

Un log de estabilidad tiene como objetivo garantizar que el consumo de recursos de una rutina no representa un riesgo para los ambientes donde será desplegada. Con este log se busca que un paquete que desea ser desplegado no presente incidentes en cuanto a:

* Alto consumo de recursos de los ambientes.
* Errores de conexión por un gran tráfico de datos.
* Optimización y mantenibilidad de consulta de datos.

Este log se caracteriza por ser generado con el 100% de los datos que consumirá la rutina, el cual permite identificar cuáles son los puntos de ejecución en los que su proceso puede generar inestabilidades y sobre los cuales se recomienda establecer un control sobre el resultado que se desea obtener.

**Nota:** Este log debe contener como prefijo “**est\_**”, de esta manera se podrá identificar que es un log de estabilidad.

**Condiciones de Generación**: Un log de estabilidad se genera bajo las siguientes condiciones:

* La rutina debe estar finalizada y lista para el proceso de calendarización.
* Debe generarse con el 100% de los datos que se consumirán en producción.
* Configurar los parámetros adicionales tal cual cómo la rutina debe ejecutar en producción; por ejemplo, fechas, prefijos, caracteres, etc.
* Ejecutar en ambientes acorde a los dispuestos para ejecutar las rutinas en producción; por ejemplo, versiones Python, Spark, etc.
* La ejecución debe ser completa tal cual como será en producción, es decir, datos, scripts y controles adicionales que desee en su proceso.

## Orquestador (Clase Logger)

Las rutinas que buscan ser calendarizadas deben seguir la estructura de orquestador2, el cual es el estándar en la generación de paquetes python para ser desplegados en producción (Documentación: [Guía paquetes analíticos con orquestador 2.0 - Overview (visualstudio.com)](https://grupobancolombia.visualstudio.com/Vicepresidencia%20de%20Innovaci%C3%B3n%20y%20Transformaci%C3%B3n%20Digital/_wiki/wikis/Vicepresidencia-de-Innovaci%C3%B3n-y-Transformaci%C3%B3n-Digital.wiki/9684/Gu%C3%ADa-paquetes-anal%C3%ADticos-con-orquestador-2.0)).

Internamente la librería orquestador2 utiliza la clase Logger, la cual sigue el estándar y estructura para evidenciar el paso a paso de ejecución de cada una de las rutinas (Documentación: [logger - Overview (azure.com)](https://dev.azure.com/GrupoBancolombia/Vicepresidencia%20de%20Innovaci%C3%B3n%20y%20Transformaci%C3%B3n%20Digital/_wiki/wikis/Vicepresidencia-de-Innovaci%C3%B3n-y-Transformaci%C3%B3n-Digital.wiki/28834/logger))

# Logs

## Estructura

La clase Logger genera un archivo .log en una ruta parametrizable a través del orquestador o cualquier otra librería o scripts que haga uso de esta, al interior de estos archivos cada paso ejecutado por la rutina tendrá la siguiente estructura:



* **AA-MM-DD**: Año, mes y fecha en que el paso del proceso fue ejecutado
* **HH:MM:SS**: Hora, minuto y segundo en que el paso del proceso fue ejecutado
* **Levelname**: Caracteriza el paso ejecutado del proceso, este puede ser:
  + [INFO]: Determina que paso fue ejecutado en el proceso (por lo general evidencia un paso ejecutado con éxito).
  + [WARNING]: Determina una advertencia sobre una situación particular que se ha detectado en la rutina pero que, asumiendo ciertas condiciones, el proceso puede continuar ejecutando.
  + [ERROR]: Determina un error de ejecución, por tanto, si no es un error controlado, la rutina debe parar y realizar las modificaciones necesarias.
* **Message**: Mensaje que da a conocer al usuario el paso que está ejecutando en su rutina, la advertencia que se genera o el error que ha surgido.

## Lectura

La lectura de los logs se realiza bajo la condición de no identificar ningún error en la ejecución de la rutina, esto es, la no presencia de [ERROR] en ningún paso del proceso. Las rutinas que presenten este tipo de alertas deben realizar las modificaciones necesarias antes de ser calendarizadas.

Sin embargo, se ha identificado que existen errores controlados en las rutinas que establecen puntos de control en los que se establecen condiciones para que el flujo de ejecución no se vea interrumpido. Estos errores controlados pueden ser programados al interior de la rutina donde se tomen decisiones automáticas de acuerdo con la presencia de algún evento especial, por ejemplo: reintentos de conexión, fuentes no ingestadas, validación de parámetros, etc. Cabe resaltar que es responsabilidad del usuario el resultado de cada ejecución al definir algún error controlado.

# Calendarización Nueva Rutina

Calendarizar una nueva rutina debe seguir las siguientes condiciones en la generación de logs:

* Generación de 3 logs de compilación, con una muestra del n% de los datos de las fuentes principales. Debe generarse con las condiciones descritas y el porcentaje de datos lo define el usuario de acuerdo con las fuentes a utilizar.
* Generación de logs de estabilidad considerando el tiempo de ejecución y las condiciones descritas para este tipo de log:

\*\*\* Rutinas con un tiempo de ejecución menor o igual a 4 horas >>>> se debe generar 3 logs de estabilidad.

\*\*\* Rutinas con un tiempo de ejecución mayor a 4 horas >>> se debe generar 2 logs de estabilidad.

Se espera que al menos un log de estabilidad debe ser generado en el horario en que se desea ser calendarizado, los demás logs deben ser generados en cualquier horario con un espacio de mínimo 2 horas entre ejecuciones.

# Modificación de Rutina

Las modificaciones de rutinas calendarizadas se pueden dividir en los siguientes tipos:

## Modificación significativa:

Una modificación significativa se evidencia cuando los usuarios realizan la modificación de un archivo Python o SQL dentro del directorio fuente (src/), este escenario será catalogado como una modificación en la lógica de la rutina. Para este escenario se define las siguientes condiciones para la generación de logs.

* Generación de 2 logs de compilación, con una muestra del n% de los datos de las fuentes principales. Debe generarse con las condiciones descritas.
* Generación de 1 log de estabilidad donde la ejecución debe realizarse en el horario en que la rutina se encuentra calendarizada y considerando las condiciones descritas para este tipo de log.

## Modificación no significativa:

Este tipo de modificación se relaciona a un cambio en los archivos externos a la carpeta fuente (src/), como la modificación de archivos in, py, json, etc. Asimismo, se considera un cambio no significativo a la modificación de archivos de configuración dentro de la carpeta fuente archivos json, config, params, etc. Para este escenario se define las siguientes condiciones para la generación de logs.

* Generación de 1 logs de compilación, con una muestra del %n de los datos de las fuentes principales. Debe generarse con las condiciones descritas.
* Generación de 1 log de estabilidad donde la ejecución debe realizarse en el horario en que la rutina se encuentra calendarizada y considerando las condiciones descritas para este tipo de log.

## Modificación por incidente:

En caso de que la rutina presente un error en su ejecución y ya se encuentre calendarizada por medio del aplicativo Control M, se procede a revisar cuál de los dos siguientes escenarios aplicaría:

* Si la rutina presenta una afectación de cara a los usuarios y/o organización, se procederá a realizar un paso por medio de una orden de cambio de emergencia. En estos escenarios no es necesario certificar el paso a producción, por lo que es el único escenario donde no se van a requerir logs.
* En caso de que la afectación no cumpla la condición anterior, se requiere adicionalmente a los cambios de la rutina, 2 los de compilación y un log de estabilidad siguiendo el escenario de una modificación significativa.

# Generador log de compilación

El generador de log de compilación es una ejecución que se realiza sobre el orquestador 2.0 siguiendo una configuración especifica. Esta aprovecha una funcionalidad de la librería impala helper que permite ejecutar rutinas solo con una porción de los datos mediante la función TABLESAMBLE de Impala, además permite establecer controles en tiempo real de si los datos están siendo consumidos como se espera.

Para observar la configuración y generación de un log de compilación diríjase a:

# Revisión Log

Al momento de revisar una rutina se tendrá en cuenta los siguientes tipos de errores:

## Errores Tolerables

En la lectura de los archivos .log pueden existir errores en algunos pasos del proceso ejecutado, sin embargo, existen hoy en día una serie de condiciones programadas en algunas librerías que toman decisiones automáticas que identifican errores y realizan un manejo controlado de estos de manera que no desencadenan una interrupción en la ejecución.

A continuación, se presentan aquellos errores que pueden presentarse en un archivo log pero que son tolerados por el proceso de calendarización:

* Reintentos de ejecución de queries sobre los servicios de impala. Comúnmente programados en librerías como el Helper, Sparky y por ende utilizador por la librería Orquestador2
* Errores manejados por los usuarios frente a fuentes no ingestadas o parámetros no encontrados, en este hay que aclarar que el resultado de la rutina en el manejo de un error de este tipo responsabilidad del usuario.

## Errores No tolerables

Los errores no tolerables son aquellos marcados en el archivo .log como un error que desencadena una interrupción en el proceso de ejecución o que evidencie un mal funcionamiento en la rutina. Este tipo de errores tiene como consecuencia que su iniciativa no pueda ser desplegada en producción hasta que se realicen las modificaciones necesarias.

Los principales desencadenadores de un error no tolerable son:

* Fuentes no ingestadas.
* Parámetros mal configurados (DSN, fechas, prefijos, zonas, etc).
* Dependencias de librerías.
* Consumo masivo de recursos.

# Revisor automático de logs

Para el análisis automático de los logs adjuntados por el usuario, se realizó la implementación de un script que se encarga de revisar los logs adjuntos por el usuario y revisar que estos cumplan las políticas descritas anteriormente. Para comprender el resultado del análisis realizado por el revisor, revisemos cuales son los objetos que revisa y los parámetros que toma:

Primero, el revisor se encarga de buscar los logs compartidos en la rutina, todos estos ubicados en la ruta base de la rutina, en un directorio llamado **logs-calendarizacion,** un ejemplo de esto:

*(nombre de la rutina)/logs-calendarización/*

**Notas:**

* En caso de no poseer los logs en esta carpeta, el pipeline que posee al revisor automático de logs no podrá terminar exitosamente.
* Al montar estos logs verifique que la carpeta logs-calendarización no esta siendo descartada por el archivo .gitignore.

Una vez que el revisor encuentra la carpeta con los logs, se empiezan las siguientes validaciones:

* Primero basado en la cantidad de logs que se compartan en esta carpeta, se va a definir el escenario que están presentando los usuarios con la rutina. Entre estos escenarios: Primera calendarización rutina larga, primera calendarización rutina corta, modificación significativa de rutina, modificación no significativa de rutina. Observemos esta pequeña tabla que muestra los escenarios:

|  |  |  |
| --- | --- | --- |
| **Escenario** | **Cantidad Logs Compilación** | **Cantidad Logs Estabilidad** |
| First Schedule (New Routine) - Long Routine (More than 4 hrs) | 3 | 2 |
| First Schedule (New Routine) - Short Routine (Less than 4 hrs) | 3 | 3 |
| Modification (Significative) | 2 | 1 |
| Modification (Non-Significative) | 1 | 1 |

* Una vez identificada la cantidad de logs, se procede a mostrar el escenario encontrado (alguno de los mencionados anteriormente) y preguntar si este es el escenario que requiere solicitar el usuario.

**Nota:** En caso de que no se encuentren las cantidades mencionadas, saldrá un error. Es importante validar si el caso que evidencia el revisor es el que requiere el usuario promotor de la rutina).

* Segundo, el revisor se encarga de chequear que los logs de compilación y de estabilidad no superen el tiempo de expiración. Este tiempo está definido como 2 meses, es decir, 62 días aproximadamente. Por lo que, al momento de correr el revisor, si los logs son menores a este margen de tiempo, se definirán como logs válidos.

**Nota:** Se espera que los logs no superen este tiempo, de manera que, si son logs muy viejos, se recomienda subir otros nuevos.

* Tercero, se realiza la validación de errores significativos. Para estos, el revisor se encarga de auditar que el orquestador haya terminado exitosamente. En caso de presentarse algún error de plataforma, se esperaría que por medio de reintentos la rutina termine exitosamente.

En caso de que el error sea relacionado a la lógica de la rutina, esta debería fallar y reportar el error y este log se catalogaría como un log con un fallo significativo.

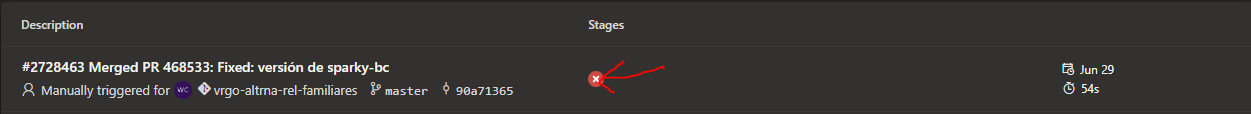
* Por último, se examina que los logs de estabilidad posean como mínimo 2 horas de diferencia entre ellos. Esta validación es exclusiva para este tipo de logs, ya que conlleva una ejecución real de la rutina.

**Nota:** En caso de solo presentar 1 logs de estabilidad, no fallaría esta prueba.

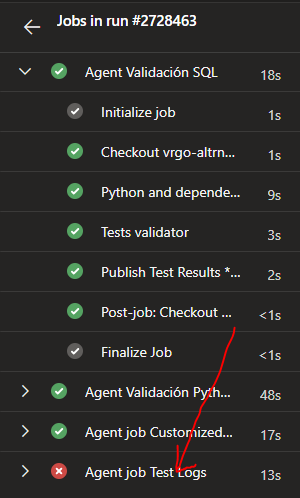
Finalmente, en caso de presentarse algún hallazgo en las validaciones anteriores, el revisor fallará y mencionará que encontró erróneo en la validación de logs.

Para revisar el resultado del revisor, seguir los siguientes pasos:

1. Buscar el pipeline fallido y presionar el botón con el error:



2. Buscar el agente encargado de la ejecución del revisor:



3. Buscar la pestaña “Test” y finalmente, revisar el output de la prueba:

