**Descripción de las alertas y notificaciones en Snowflake**

* Establecer la confianza y satisfacción del cliente es crucial para cualquier negocio
* La observabilidad juega un papel importante en el logro de este objetivo
* Esta es una de las principales tendencias en el panorama de datos
* Para permitir una mejor observabilidad y medibilidad de las cargas de datos, Snowflake ha estado agregando contínuamente características nativas
* La última incorporación es la función alertas y notificaciones de Snowflake
* Actualmente esta función se encuentra en public preview
* Esta función proporciona una valiosa observabilidad de tus datos y de tus cargas

**¿Qué son las alertas de Snowflake?**

* Una alerta de Snowflake es un objeto a nivel de esquema que especifica lo siguiente:
  + Una condición que activa la alerta (por ejemplo, el error de una task, el uso de recursos excede un umbral, etc)
  + La acción a realizar cuando se cumple la condición (por ejemplo, enviar una notificación por correo electrónico o capturar los datos en una tabla para un análisis posterior o ambos)
  + Cuándo y con qué frecuencia se debe evaluar la condición (por ejemplo, cada 30 minutos, todos los días a las 7 p.m., etc)
* La acción desencadenada por la alerta puede ser una de las siguientes:
  + Enviar una notificación mediante el procedimiento SYSTEM$SEND\_EMAIL
  + Ejecutar consultas SQL
  + Llamar a un procedimiento almacenado personalizado que se puede utilizar para realizar más acciones
* Las alertas necesitan un warehouse para ejecutarse
* La programación de la alerta se puede especificar mediante una expresión cron o mediante un intervalo de minutos
* Sintaxis SQL para crear una alerta en Snowflake:

CREATE OR REPLACE ALERT MY\_ALERT

WAREHOUSE = MY\_WH

SCHEDULE = ‘10 MINUTE’ ó ‘USING CRON 0 3,11,18,22 \* \* \* Europe/Madrid’

IF(EXISTS(CONDITION)) THEN ACTION;

* El valor máximo admitido para el número de minutos es 11520 (8 días)
* Al crear una alerta en Snowflake hay que tener en cuenta los siguientes factores:
  + Es recomendable utilizar un rol personalizado para crear y ejecutar alertas
  + Los privilegios de ejecución de alertas solo los puede otorgar el ACCOUNTADMIN
  + El rol personalizado debe tener los privilegios de uso y de creación de alertas en el esquema donde se desea crear las alertas
  + El rol personalizado debe tener privilegios de uso en la base de datos y el warehouse correspondientes
  + Cuando la alerta se crea inicialmente, se suspende de forma predeterminada
  + Para reanudar una tarea, utilizar el comando ALTER ALERT MY\_ALERT RESUME;

**Envío de notificaciones por correo electrónico**

* Esta característica se encuentra actualmente en versión public preview
* Snowflake puede enviar notificaciones por correo electrónico utilizando el procedimiento almacenado incorporado SYSTEM$SEND\_EMAIL
* Snowflake administra la entrega de correos electrónicos y esta característica utiliza el objeto de integración de notificaciones
* Para enviar una notificación por correo electrónico:
  + Asegúrate de que los destinatarios previstos de la notificación hayan verificado sus direcciones de correo electrónico (pueden hacerlo a través de su perfil en la interfaz de usuario de Snowsight)
  + Crea una integración de notificaciones de tipo email
    - Este paso fallará si incluyes cualquier correo electrónico que no esté verificado
  + Otorga el privilegio al rol que necesita utilizar la integración de notificaciones
* Ten en cuenta que puedes definir un máximo de 10 integraciones de notificaciones por correo electrónico para una cuenta determinada

**Creación de una integración de notificaciones**

* De forma predeterminada solo el rol ACCOUNTADMIN tiene el privilegio para crear una integración de notificaciones
* Este privilegio se puede conceder a roles adicionales según se necesario
* A continuación, veamos cómo se puede crear una integración de notificación por correo electrónico para enviar mensajes de correo electrónico a 2 idenficadores de correo electrónico verificados por el sistema

CREATE OR REPLACE NOTIFICATION INTEGRATION EMAIL\_NOTIFICATION\_INTEGRATION TYPE = EMAIL

ENABLED = TRUE

ALLOWED\_RECIPIENTS = (‘myid@domain.com’, ‘admin@domain.com’);

* Puedes otorgar el privilegio de usar una integración de notificaciones a los roles que necesites

Enviar una notificación por correo electrónico

**Ejemplo de envío de una notificación por correo electrónico**

CALL SYSTEM$SEND\_EMAIL (

‘EMAIL\_NOTIFICATION\_INTEGRATION’,

‘myid@domain.com’ ó ‘[myid@domain.com, admin@domain.com]’,

‘Alert: Task failed’,

‘Task TASK\_INGEST\_RAW failed’);

**Casos de uso**

* Las alertas y notificaciones de Snowflake ofrecen capacidades valiosas para diversos casos de uso relacionados con la observabilidad
* Las alertas se pueden configurar para responder rápidamente a cualquier problema o anomalía
* Las alertas y notificaciones se pueden utilizar de forma independiente o en combinación para mejorar la supervisión de tus datos y de tus cargas.

**Caso de uso 1: Comprobar y notificar cuando las consultas se ejecutan más de lo esperado**

* Supongamos que la mayoría de consultas de tu data warehouse suelen completarse en 10 minutos
* Ocasionalmente, ciertas consultas suelen tardar más en finalizar, lo que indica una necesidad de optimización o un almacén más grande
* Para mantenerse informado y tomar las medidas necesarias, sería beneficioso recibir notificaciones cuando las consultas superen un umbral de 30 minutos y aún se estén ejecutando
* Un enfoque es configurar una alerta que examine el historial de consultas cada 15 minutos, indentificando dichas instancias y notificándolo rápidamente al usuario

CREATE OR REPLACE ALERT LONG\_RUNNING\_QUERIES\_ALERT

WAREHOUSE = MY\_WH SCHEDULE = ‘USING CRON \*/15 \* \* \* \* Europe/Madrid’

IF(EXISTS((

SELECT \* FROM TABLE(SNOWFLAKE.INFORMATION\_SCHEMA.QUERY\_HISTORY())

WHERE EXECUTION\_STATUS ILIKE ‘RUNNING’ AND START\_TIME <= CURRENT\_TIMESTAMP() – INTERVAL ’30 MINUTES’))

THEN CALL SYSTEM$SEND\_EMAIL (‘EMAIL\_NOTIFICATION\_INTEGRATION’, ‘myid@domain.com’, ‘Alert: Long running queries, ‘There are queries running for more than 30 minutes in your DWH. Please check.);

* La dirección de correo electrónico que recibe las notificaciones ya debería estar agregada en la integración de notificaciones
* En el DDL proporcionado, la alerta está programada para ejecutarse cada 15 minutos utilizando una programación cron
* Es importante tener en cuenta que puede haber consultas que tardaron más de 30 minutos pero que se completaron entre intervalos de alerta, por tanto, podemos no estar reportando todas las queries que tardaron en ejecutarse más de 30 minutos
* Al implementar una condición que detecta consultas de larga duración, puedes identificar y priorizar fácilmente las consultas que requieren atención inmediata

Captura de pantalla de una notificación por correo electrónico de Snowflake para la alerta anterior

**Caso de uso 2: Usar un procedimiento almacenado personalizado como acción de la alerta para capturar o resumir los datos que desencadenaron la alerta**

* Exploraremos la posibilidad de capturar los datos que activaron la alerta y almacenarlos en una tabla, además de enviar notificaciones
* Esto puede ser valioso para realizar un análisis más detallado de los datos cuando sea necesario
* Profundizaremos en un caso de uso específico para comprender cómo podemos lograr esto
* Imagina que eres una empresa de productos con el objetivo de optimizar tu proceso de resolución de quejas
* Uno de tus requisitos es capturar todas las quejas plantedas por los miembros premium y notificar a un equipo dedicado periódicamente
* Para lograr esto puedes configurar una alerta que se ejecute en una programación de 30 minutos con la siguiente configuración
  + Condición: cuando un miembro premium presenta una nueva queja
  + Acción: llamar a un procedimiento almacenado que inserta los datos de la queja en una tabla designada y envía una notificación al equipo
* Para capturar nuevas quejas cada vez que se ejecuta la alerta, puedes utilizar las funciones LAST\_SUCCESSFUL\_SCHEDULED\_TIME y SCHEDULED\_TIME dentro de la alerta
* Esto garantiza que solo se capturen y procesen las últimas quejas
  + SCHEDULED\_TIME devuelve la marca de tiempo que representa cuándo se programó la alerta actual
  + LAST\_SUCCESSFUL\_SCHEDULED\_TIME devuelve la marca de tiempo que representa cuando se programó la última alerta evaluada correctamente
* Ten en cuenta que estas dos funciones solo se pueden utilizar dentro de la alerta

CREATE OR REPLACE ALERT CAPTURE\_AND\_ALERT\_NEW\_PREMIUM\_MEM\_COMPLIANT

WAREHOUSE = MY\_WH SCHEDULE = ’30 MINUTES’

IF(EXISTS(

SELECT \* FROM CUSTOMER\_COMPLAINTS

WHERE COMPLAINT\_LOG\_TS BETWEEN SNOWFLAKE.ALERT.LAST\_SUCCESSFUL\_SCHEDULED\_TIME() AND SNOWFLAKE.ALERT.SCHEDULED\_TIME() AND MEMBERSHIP\_TYPE ILIKE ‘PREMIUM’))

THEN CALL SP\_PREMIUM\_MEM\_COMPLIANT\_CAPTURE\_AND\_ALERT(SNOWFLAKE.ALERT.LAST\_SUCCESSFUL\_SCHEDULED\_TIME(), SNOWFLAKE.ALERT.SCHEDULED\_TIME());

* En el procedimiento almacenado, los datos se capturan en una tabla y también se envía una notificación
* También puedes capturar los datos sin enviar ninguna notificación

CREATE OR REPLACE PROCEDURE TEST\_COMPLAINTS.CUSTOMER\_SERVICE.SP\_PREMIUM\_MEM\_COMPLIANT\_CAPTURE\_AND\_ALERT(‘START\_TS’ VARCHAR, ‘END\_TS’ VARCHAR)

RETURNS VARCHAR LANGUAGE JAVASCRIPT EXECUTE AS CALLER AS $$

var sql\_ins = ‘ INSERT INTO PREMIUM\_MEMBER\_COMPLAINTS(COMPLAINT\_ID, COMPLAINT\_TYPE, COMPLAINT\_LOG\_TS)

SELECT COMPLAINT\_ID, COMPLAINT\_TYPE, COMPLAINT\_LOG\_TS FROM CUSTOMER\_COMPLAINTS

WHERE COMPLAINT\_LOG\_TS BETWEEN :1 AND :2 AND MEMBERSHIP\_TYPE = ‘PREMIUM’’;

var stmt1 = snowflake.createStatement({sqlText: sql\_ins, binds: [START\_TS, END\_TS]});

stmt1.execute();

var email\_query = ‘CALL SYSTEM$SEND\_EMAIL (‘EMAIL\_NOTIFICATION\_INTEGRATION’, ‘myid@domain.com’, ‘Email alert for PREMIUM membership: New complaints logged’, ‘There are complaints from members with PREMIUM memberships. Please check.’)’;

var stmt2 = snowflake.createStatement({sqlTest: email\_query});

stmt2.execute();

return ‘Data capture and Notification evaluation successfully completed’

$$;

* Ten en cuenta que se deben otorgar los privilegios necesarios al rol autor de la llamada
* Tenemos la opción de mejorar el procedimiento almacenado personalizado para incluir un resumen de la condición de alerta, que se puede enviar junto con la notificación
* Este paso adicional les permite proporcionar más contexto e información relevante a los destinatarios de la notificación

**Caso de uso 3: Supervisar y generar una alerta si los datos no se actualizan según la** programación esperada

* Si tienes una tabla actualizada con frecuencia que sirve como base para informes casi en tiempo real, es crucial asegurarse de que los datos se actualicen en intervalos regulares
* Para abordar este requisito, las alertas de Snowflake se pueden utilizar de manera efectiva
* Veamos cómo podemos implementar una alerta que verifique la actualización de los datos
* Podemos configurar una alerta para supervisar la última hora de actualización de la tabla
* Si la actualización no se ha producido en los últimos 30 minutos, indica un posible problema que requiere atención
* La alerta se activará en función de esta condición, lo que permitirá tomar las medidas adecuadas
* La siguiente alerta está programada para ejecutarse cada 30 minutos y se activa si la taba no se modificó entre LAST\_SUCCESSFUL\_SCHEDULED\_TIME y SCHEDULED\_TIME

CREATE OR REPLACE ALERT NOTIFY\_TABLE\_NOT\_REFRESHED

WAREHOUSE = MY\_WH SCHEDULE = ’30 MINUTES’

IF(EXISTS(

SELECT \* FROM MONITORING.INFORMATION\_SCHEMA.TABLES WHERE TABLE\_SCHEMA = ‘REPORTING’ AND TABLE\_NAME = ‘LOGS’ AND LAST\_ALTERED NOT BETWEEN SNOWFLAKE.ALERT.LAST\_SUCCESSFUL\_SCHEDULED\_TIME() AND SNOWFLAKE.ALERT.SCHEDULED\_TIME()))

THEN CALL SYSTEM$SEND\_EMAIL (‘EMAIL\_NOTIFICATION\_INTEGRATION’, ‘myid@domain.com’, ‘Alert: Log table not refreshed in the last 30 minutes’, ‘The log table has not been refreshed for the last 30 minutes.’);

* Si prefieres tener la flexibilidad de suspender o reanudar la alerta, o quieres comprobar contínuamente la actualización de la tabla independientemente de las funciones de programación de alertas, puedes utilizarla siguiente consulta dentro de la condición

SELECT DATEDIFF(MINUTE, LAST\_ALTERED, CURRENT\_TIMESTAMP()) AS MINUTES\_SINCE\_LAST\_UPDATE

FROM MONITORING.INFORMATION\_SCHEMA.TABLES

WHERE TABLE\_SCHEMA = ‘REPORTING’ AND TABLE\_NAME = ‘LOGS’ AND MINUTES\_SINCE\_LAST\_UPDATE > 30;

**Otros casos de uso para alertas y notificaciones de Snowflake**

* Notificar cuando los datos no cumplan con una regla de negocio
* Notificar cuando se produzca un error en la task
* Al utilizar alertas y etiquetas de Snowflake, puede supervisar eficazmente el uso de recursos agrupando recursos y recibiendo notificaciones cuando superan los umbrales predefinidos establecidos para cada grupo
  + Esto permite un mejor control y administración de la asignación de recursos dentro de tu entorno de Snowflake
* Para un nuevo pipeline de ingesta en producción, es beneficioso configurar alertas que te avisen cuando se produce una cola o derrame remoto en un warehouse
  + Estas alertas sirven como indicadores para evaluar si es necesario escalarar horizontalmente los clusters o aumentar el tamaño del warehouse
  + Al supervisar y responder de forma proactiva a estas alertas, puede optimizar el rendimiento y la eficiencia de su pipeline de ingesta para satisfacer la demanda de su carga de trabajo
* Configura una alerta para supervisar el uso de los servicios en la nube del warehouse e identificar su supera un porcentaje específico en relación con los créditos informáticos
  + Este enfoque proactivo permite evaluar si se requieren ajustes para administrar eficazmente sus costes
  + Al supervisar de cerca los patrones de uso y alinearlos con tu presupuesto, puedes optimizar la optimización de los recursos en la nube y garantizar la rentabilidad de tus operaciones

**¿Cómo supervisamos la ejecución de las alertas de Snowflake?**

* Las alertas se pueden usar para monitorizar tus cargas
* Para monitorizar y solucionar problemas del estado de las alertas, considera las siguientes opciones:
  + Checkea el historial de ejecuciones de alertas ejecutando la función de tabla ALERT\_HISTORY en el INFORMATION\_SCHEMA
    - Por ejemplo, la siguiente consulta proporcionará el historial de ejecuciones de alertas y las alertas programadas durante la última hora:

SELECT \* FROM TABLE(SNOWFLAKE.INFORMATION\_SCHEMA.ALERT\_HISTORY(SCHEDULED\_TIME\_RANGE\_START => DATEADD(‘hour’, -1, CURRENT\_TIMESTAMP())))

ORDER BY SCHEDULED\_TIME

* + - El historial devuelve una fila para cada ejecución de alerta
    - Cada registro mostrará los detalles de ejecución, incluido el estado de la alerta, que puede ser uno de los siguientes: SCHEDULED, EXECUTING, FAILED, CANCELLED, CONDITION\_FALSE, CONDITION\_FAILED, ACTION\_FAILED, TRIGGERED
  + Si no ves tu alerta en el ALERT\_HISTORY, comprueba que la alerta se reanudó despues de la creación
    - El estado de la alerta (SUSPENDED/STARTED) y otros detalles se pueden comprobar ejecutando cualquiera de los siguientes comandos: SHOW ALERTS; DESC ALERT MY\_ALERT;
  + El CONDITION\_QUERY\_ID y el ACTION\_QUERY\_ID de la ejecución de una alerta se devuelven en el historial de alertas
  + Puedes checkear los resultados de la query de la condición que desencadena la alerta pasando el CONDITION\_QUERY\_ID en la función RESULT\_SCAN: SELECT \* FROM TABLE(RESULT\_SCAN(<CONDITION\_QUERY\_ID\_FROM\_ALERT\_HISTORY>));

**Limitaciones y consideraciones**

* Seguridad y privacidad de datos
  + La característica del procedimiento almacenado SEND\_EMAIL de Snowflake se aloja actualmente en AWS en algunas regiones seleccionadas que si están fuera de la región de tu instancia de Snowflake puede requerir que salgas tu región
  + Para evitar el riesgo de exponer información confidencial y datos PII (información de identificación personal), se recomienda limitar el contenido del correo electrónico para notificar al usuario de la condición o el resumen que activó la alerta
  + Cualquier información relevante sobre la condición puede ser capturada dentro del sistema para su posterior análisis en lugar de ser incluida en el cuerpo del correo electrónico
  + Sigue las mejores prácticas y cumple con las regulaciones de protección de datos relevantes
* Dado que el proveedor de correo electrónico para el procedimiento almacenado SEND\_EMAIL es externo a tu organización, es importante tener en cuenta que no es posible usar direcciones de correo electrónico de tu organización para enviar notificaciones
* Al programar una alerta es importante tener en cuenta el warehouse que se va a utilizar
  + Para alertas frecuentes, es aconsejable seleccionar un warehouse que ya esté activo en lugar de iniciar un nuevo warehouse únicamente con el propósito de las alertas
  + Este enfoque optimiza la utilización de recursos y reduce la necesidad de activar warehouses adicionales
* Ten en cuenta el potencial de ruido de las alertas
  + Las alertas irrelevantes o excesivas pueden abrumar el sistema de monitorización
  + Revisa y perfecciona regularmente las reglas de alerta para alinearlas con la dinámica cambiante del sistema

Resumen

* Las alertas de Snowflake ofrecen una funcionalidad valiosa para supervisar y solucionar problemas de cargas
* Cuando se activa una alerta, los usuarios tienen la flexibilidad de realizar diversas acciones, como enviar notificaciones o ejecutar sentencias SQL para iniciar procesos posteriores
* Al aprovechar el sistema de notificación nativo de Snowflake junto con las alertas, puedes garantizar una comunicación oportuna y efectiva, lo que permite responder rápido a eventos críticos
* Dado que tenemos la capacidad de llamar a un procedimiento almacenado personalizado como una acción del desencadenador de la alerta, abre oportunidades para realizar acciones no estándar adicionales
  + Por ejemplo, puedes activar una función externa desde el procedimiento, lo que permite enviar un correo electrónico con su proveedor de correo electrónico o agregar la información de alerta a una cola externa que, a su vez, pasa la información de alerta a un canal de Slack o Teams

Pendiente

* Estructurar la exposición
  + Problemática que vienen a resolver
  + Qué son y cómo funcionan
  + Limitaciones
  + Cómo se implementan: guía de los pasos a alto nivel
  + Implementación de varios casos de uso
* Redactar la exposición en word y después hacer la presentación en power point tomando como referencia el word
* Enseñarle a JR la exposición

**Exposición**

**Objetivo**: mejorar la observabilidad de nuestros datos y cargas

**Propuesta**: crear alertas en Snowflake que nos avisen cuando ocurre un problema con nuestros datos y/o cargas, y que realicen las acciones necesarias de forma automática

**Implementación**: crear una integración de notificación, crear una alerta y monitorizar

**Casos de uso**: checks técnicos y/o funcionales fallidos (para controlar el flujo de la ETL), tasks fallidas, encolamientos, sorpaso de umbrales de consumo

**Buenas prácticas**: capturar las causas de la alerta en una tabla y no compartirlas   
**Limitaciones**: solo en ciertas regiones de AWS, incompatibilidades con el proveedor de correo

**Práctica** (cuenta trial de Snowflake): crear una integración de notificación con un grupo de Teams, crear una alerta que envíe un mensaje al grupo cuando haya duplicados en una tabla y que inserte los registros duplicados en otra tabla

**Nota**: añadir los detalles técnicos

**Propuestas de JR:**

* Como pueden enviarse mails a un buzón, cabe la posibilidad de diseñar bots con terceras partes que escuchen el buzón y actúen en consecuencia, lo cual abre la puerta a una automatización completa del alertado
  + Con bot nos referimos a un script que se conecte cada hora y vea si han llegado mails de fallos de las tasks y apliquen la solución que corresponde automáticamente (por ejemplo, si una task ha fallado porque el stream se ha quedado corrupto, la solución será recrear automáticamente el stream)
* Mencionar que en la propia alerta puedes llamar a un procedimiento almacenado personalizado que actúe según la casuística del trigger de la alerta
  + De esta forma no solo alertamos sino que actuamos
* Posibilidad: slide final en la que se nombren las múltiples posibilidades de ampliar el alertado para automatizar el proceso de resolución de problemáticas
  + Posibles ampliaciónes son: automatizar acciones con Snowflake, automatizar acciones con terceros, incorporar modelos entrenados, etc
* Recomendación: siempre que propongas una solución es bueno contraponer (tienes esta solución, esta y la otra)
  + Que tengas opciones unificar todos tus procesos en la misma plataforma no significa que restringirte a ella
  + Para la presentación, una estrategia puede ser exponer solo la posibilidad de automatizar con Snowflake y reflexionar al final de ella sobre el posible uso de otras herramientas para ir todavía más allá (notar que esto puede ser muy atractivo por la cantidad de herramientas con las que se puede integrar Snowflake de forma sencilla)
* Para cada tabla de mi modelo, tener una tabla de duplicados en la que estén solo los registros duplicados en la tabla a la que hace referencia
* Además, tener una tabla global (i.e., para todo el modelo) que contenga el conteo de los duplicados para todas las entidades del modelo, y sobre esta tabla construir un dashboard en Snowflake que permita una consulta rápida de las posibles problemáticas existentes
  + De esta forma dentro del entramado de alertado añadimos una característica que mejora la observabilidad
* A partir de lo anterior, hay que aumatizar el saber si un duplicado viene de origen y reportarlo al productor (u otras características frecuentes)
* Para la demo usar tasks, streams, procedimientos, alertas, mail y una tabla de logs con un dashboard

**Credenciales cuenta trial Snowflake**:  
SUAlert

Password=1234

https://kzrnqwx-co84058.snowflakecomputing.comhttps://kzrnqwx-co84058.snowflakecomputing.com

**Siguientes pasos**:

* Construir la demo:
  + Añadir al procedimiento almacenado personalizado la funcionalidad de que se cargue los duplicados
  + Crear alerta que ejecute el procedimiento almacenado personalizado que crea una tabla con los duplicados y notifica por email
  + Crear un dashboard sobre la tabla de duplicados
  + Replicar el proceso ya hecho con otras tablas hasta crear un modelo y ver si se puede agrupar código
  + Construir una ETL completa con base de datos del productor y nuesto modelo completo sobre el que aplicar el alertado.
    - Crear otros usuarios con correos verificados para reclamar a ellos en caso de duplicados
  + Crear un dashboard global con el conteo de duplicados en mi modelo
* Mejorar la demo actual (más complejidad (pensar si tiene cabida usar tasks, streams), más acciones, otros casos de uso a parte del de duplicados, etc)
* Rellenar una ppt y un guión (incluir slides para la demo)
  + Recordar la slide final con propuestas de ampliación del alertado (automatización a partir del alertado)
* Enseñar presentación y demo a JR para que me dé feedback y me proponga mejoras