REPORTE OFICIAL | ABRIL DE 2016

Prueba de Extracción, Transformación y Carga de Datos Totalmente Automatizada: Guía Paso a Paso

Sección 1

El papel fundamental de la extracción, transformación y carga de datos en las organizaciones modernas

Desde su erupción en el mundo del almacenamiento de datos y la inteligencia empresarial, la extracción, transformación y carga de datos (ETL) se ha convertido en un proceso ubicuo en el mundo del software. Como su nombre indica, una rutina de ETL consta de tres pasos distintos, que a menudo tienen lugar en paralelo: los datos se extraen de una o varias fuentes de datos, se convierten en el estado requerido y se cargan en el destino deseado, normalmente en un almacenamiento de datos, data mart o base de datos. Una rutina desarrollada de ETL a menudo también incluye una infraestructura de control de errores, de registro y un entorno de rutina.[1]

Hasta ahora, la ETL se ha utilizado principalmente para preparar con frecuencia grandes volúmenes de datos y dispares para el análisis y la inteligencia empresarial. Sin embargo, su uso se está ampliando más allá del simple movimiento de datos: la migración de datos de nuevos sistemas se está convirtiendo en una aplicación cada vez más común, así como también el control de las integraciones, la organización y la uniones de datos.[2]

Por lo tanto, la ETL constituye una característica importante del ciclo de vida moderno y veloz del desarrollo en el cual se desarrollan varias ediciones y versiones en paralelo en un momento determinado. Las organizaciones deben ser capaces de mejorar, integrar e innovar sus software de forma continua, y de brindar datos en el estado correcto a los evaluadores y desarrolladores para cada iteración y edición. Los datos se extraen de las mismas fuentes, pero se deben transformar para satisfacer las necesidades específicas de cada equipo individual. Esto es especialmente cierto si una organización se está esforzando por ser “ágil” o por implementar correctamente una entrega continua.

Un buen ejemplo del papel importante que la ETL desempeña en la entrega continua se vio en un gran banco multinacional con el cual CA Technologies trabajó. El banco se encontraba en el proceso de realizar una compra y tenía que migrar los clientes, los productos y las finanzas de los bancos adquiridos en sus infraestructuras existentes. Esto significó la recuperación, conversión y validación de 80 archivos de inserción, antes de cargarlos en los sistemas back-end de los bancos. Además, dichos datos debían estar disponibles en 47 proyectos, en paralelo, y se debía mantener la integridad de referencia de los datos. En este caso, la ETL fue importante para permitir el paralelismo requerido a fin de proporcionar una entrega continua de forma correcta.

Sin embargo, a pesar del mayor uso e importancia de la ETL, la prueba de ETL refleja que el estado de las pruebas, en general, es demasiado lento y manual, pero que permite una cantidad inaceptablemente elevada de defectos en la producción. En este documento se presentará cómo normalmente se abordan los retos encontrados en una estrategia típica de la prueba de ETL y se explorarán los retos encontrados. Se presentará una estrategia amplia y alternativa basada en modelo, y se considerará cómo esta podría hacer que la prueba de ETL sea más eficiente, eficaz y sistemática.

Sección 2

La estrategia típica de la prueba de ETL y los retos comunes encontrados

Al validar las reglas de transformación de la ETL, los evaluadores normalmente crean un conjunto de códigos paralelos, que se usa para transformar los datos, y luego comparan los resultados reales con los esperados. Normalmente, los scripts o SQL de ETL se copian de forma manual en los datos de origen y se ejecutan; los resultados se graban. El mismo script luego se copia en los datos de destino y los resultados se registran. Luego, se comparan los dos conjuntos de resultados (reales y esperados) para validar que los datos se transformaron correctamente.

El problema principal: complejidad y comprobación

El problema fundamental detrás de dicha validación manual es que las rutinas de ETL, por su naturaleza, se tornan muy complejas rápidamente. A medida que la empresa crece y que la variedad de volúmenes de datos que esta recolecta aumenta, las reglas de ETL crecen para poder controlarlos. En la denominada "era de la información", este crecimiento se produce de forma más rápida de lo que los métodos de prueba tradicionales pueden controlar. De hecho, la gran cantidad de información que las organizaciones basadas en datos recolectan ha crecido de forma tan rápida que, solo en los dos últimos años, se recolectó un 90 % de datos en el mundo[3], mientras la cantidad de datos recolectados por las organizaciones promedio se duplica cada año.[4]

La complejidad de los sistemas diseñados para recolectar, transformar operar y presentar estos datos crece exponencialmente con cada decisión añadida. Esto incluye las reglas de ETL, y existen numerosos factores que pueden tener un impacto en la complejidad de las transformaciones:

- la cantidad y variedad de orígenes de datos involucrados, incluidos los tipos de bases de datos relacionales y no relacionales, además de los archivos planos;
- la cantidad y variedad de destinos de datos;
- la cantidad y variedad de transformaciones simples y complejas, desde búsquedas simples hasta uniones activas y normalizaciones;
- la cantidad de transformaciones, fragmentos de código y uniones reutilizables;
- la cantidad de tablas creadas.[5]

El enfoque actual en las soluciones en tiempo casi real y la complicación añadida que estos presentan empeoran todos estos factores.[6]

La documentación no ayuda

La creciente complejidad tiene un impacto directo en la comprobación de las rutinas de ETL. Es especialmente problemática para la prueba de ETL, ya que las reglas de transformación normalmente se almacenan en una documentación deficiente que carece de resultados esperados explícitos. Las reglas se diseñan a menudo durante la propia fase de desarrollo y con frecuencia se almacenan en documentos u hojas de cálculo escritas; o, lo que es peor, es posible que existan solo en las mentes de los evaluadores y desarrolladores.[7] En este caso, no existe una documentación real a partir de la cual se puedan obtener con confianza casos de prueba (es decir, el código paralelo).

Con un equipo de inteligencia empresarial con el cual trabajamos, los requisitos se almacenaron como documentos escritos, mientras que los casos de prueba se almacenaron como hojas de cálculo. Esta documentación estática presentó un "muro de palabras" a partir del cual se tuvo que descifrar los pasos lógicos de las rutinas de ETL. Los documentos se centraron en una "ruta sin errores" y no contenían condiciones negativas, de modo que se pasó por alto aproximadamente el 80 % de la lógica posible que se debía probar en el sistema promedio. Dicha documentación ambigua e incompleta significó que los evaluadores no tenían ningún modo de comprender de forma fácil o precisa las rutinas de ETL.

Demasiado a menudo, se dejaba a los evaluadores completar los espacios en blanco, pero cuando se equivocaban, se introducían defectos en las rutinas de ETL. Por lo tanto, se copiaban datos no válidos en el destino, a pesar de que el código y los casos de prueba reflejaban una interpretación plausible de la documentación de requisitos.

“Basura entrante, basura saliente”: obtención manual de casos de prueba y resultados esperados

De hecho, un gran porcentaje del 56 % de los defectos que llega a la producción se remonta a la ambigüedad de los documentos de requisitos.[8] Esto se debe en parte a que los casos de prueba y los resultados esperados se obtienen de forma manual de la documentación deficiente: un proceso que requiere mucho tiempo y que normalmente conduce a una cobertura de prueba deficiente.

Calidad

La derivación manual es ad hoc y poco metódica, y normalmente lleva a la creación de cualquier caso de prueba que existe en las mentes de los evaluadores. Dada la complejidad analizada de las rutinas de ETL, combinada con la documentación deficiente que se ofrece, no podemos esperar que incluso el evaluador más talentoso cree todas las pruebas necesarias para validar las posibles combinaciones de datos. Por ejemplo, si un sistema simple de 32 nodos y 62 bordes está diseñado de forma lineal, pueden existir 1 073 741 824 rutas posibles a través de su lógica (más de lo que una persona puede pensar).

Por lo tanto, la derivación ad hoc genera pruebas masivas insuficientes y sobrepruebas, donde solo se prueban una fracción de la lógica posible involucrada en una rutina de ETL. La prueba negativa constituye un reto particular y los casos de prueba, como la documentación, normalmente se centran casi exclusivamente en las rutas sin errores. Sin embargo, son estos valores atípicos y resultados no esperados los que se deben probar, ya que es fundamental que las rutinas de ETL rechacen dichos “datos incorrectos”.

Por ejemplo, una de las compañías de servicios financieros con la cual CA Technologies trabajó se basó en 11 casos de prueba con una cobertura de prueba de solo el 16 %. Esta cifra es bastante estándar y nuestras auditorías observaron que una cobertura funcional de prueba del 10 % y el 20 % es la norma. Otro proyecto de la empresa se sobreprobó con un factor de 18, ya que acumularon casos de prueba en un esfuerzo por probar completamente el sistema, pero aún así no lograron la cobertura máxima. La ejecución de los 150 casos de prueba adicionales realizada por un proveedor contratado les costó USD 26 000.[9]

El resultado de dicha cobertura deficiente es que se introducen defectos en el código, que son caros y requieren mucho tiempo para corregirlos: en los estudios se observó que puede costar entre 40 y 1000 veces más recursos[10] y 50 veces más tiempo[11] corregir un error durante la prueba en comparación con detectarlo antes. Peor aún, es posible que los errores no se detecten, y, por lo tanto, se copien datos no válidos en el objetivo activo, lo que puede amenazar la integridad del sistema. Además, al enfrentarse con la documentación estática, los evaluadores no tienen una manera segura de medir la cobertura de sus casos de prueba: simplemente no pueden afirmar con confianza cuánto o cuán poco de una determinada rutina están probando ni tampoco pueden priorizar pruebas en función de la importancia.

Tiempo y esfuerzo: la prueba simplemente no puede seguir el ritmo

Los casos de prueba escritos de dicha documentación también requieren mucho tiempo y mano de obra. En el ejemplo anterior, llevó 6 horas crear los 11 casos de prueba, mientras que la sobreprueba desenfrenada en la empresa consumió mucho más tiempo. Este tiempo desperdiciado en el diseño manual de casos de prueba empeora aún más con el tiempo que luego debe emplearse en la comparación de los resultados reales con los esperados.

Comparar el gran número de campos individuales de los resultados esperados requiere mucho tiempo, dada la cantidad de datos que una rutina compleja de ETL genera y el hecho de que los datos de origen con frecuencia se almacenan en una diversa variedad de tipos de archivos y bases de datos. También resulta muy difícil, ya que los datos transformados se deben validar en varios niveles:

- Los evaluadores deben controlar la integridad de los datos y asegurarse de que los recuentos del destino y origen de datos coinciden.
- Debe garantizarse la integridad de los datos y controlar si los datos de destinos son coherentes con los datos de origen.
- La transformación debe coincidir con las reglas empresariales.
- Debe garantizarse la coherencia de datos y se debe identificar cualquier dato duplicado inesperado.
- Debe mantenerse la integridad de referencia y se debe detectar cualquier registro huérfano o clave externa perdida.[12]

A veces se llega a un acuerdo y solo se valida un conjunto de datos de muestra. Sin embargo, esto también compromete la minuciosidad de la prueba de ETL y, por la tanto, la confiabilidad de las transformaciones se ve afectada. Dado el papel de muchas de las rutinas de ETL en las operaciones empresariales críticas, dicho peligro es inaceptable. La calidad se ve afectada todavía más por la naturaleza propensa a errores de las comparaciones manuales, especialmente cuando los resultados esperados no están correctamente definidos o, lo que es aún peor, no se definen en absoluto de forma independiente del código paralelo utilizado en la prueba. En este caso, los evaluadores tienden a suponer que se pasó una prueba, a menos que el resultado real sea especialmente estrafalario: sin resultados esperados predefinidos, es posible que asuman que el resultado real es el resultado esperado[13] y, por lo tanto, no pueden determinar con seguridad la validez de los datos.

El problema de los datos

Hasta ahora, los problemas encontrados al obtener las pruebas (código paralelo) necesarias se debían a la validación de las reglas de ETL en las que se centraban. Sin embargo, una vez que se obtienen los casos de prueba, los evaluadores necesitan datos de origen ficticios para bombearlos a través del sistema. Esta es otra de las causas frecuentes de cuellos de botella y defectos.

¿Tiene todos los datos que necesita para probar las complejas rutinas de ETL?

Tener suficientes datos "incorrectos" es importante para obtener una prueba de ETL eficaz, ya que es fundamental que, cuando está en ejecución, una regla de ETL rechace estos datos y los envíe al usuario correspondiente en el formato correcto. Si no se rechazan, es posible que estos datos incorrectos creen defectos o incluso hagan que el sistema falle.

Los "datos incorrectos" en este contexto pueden definirse de varias maneras, lo que se corresponde con los modos en los que los evaluadores deben validar los datos. Pueden ser datos que, en función de las reglas empresariales, nunca deben aceptarse; por ejemplo, valores negativos en un carro de compras en línea, cuando no hay un vale. Pueden ser datos que amenazan la integridad de referencia de un almacenamiento, como datos interdependientes y obligatorios faltantes, o datos que faltan entre los propios datos entrantes.[14] Por lo tanto, los datos de prueba que se bombean a través de una regla de validación de ETL deben contener el rango completo de datos no válidos para permitir una cobertura de prueba funcional del 100 %.

Dichos datos rara vez se encuentran entre los orígenes de datos de producción que aún se proporcionan a los equipos de pruebas en muchas organizaciones. Esto se debe a que los datos de producción se extraen de los escenarios de "enfoque habitual" que tuvieron lugar en el pasado y que, por lo tanto, se sanean por su propia naturaleza para excluir los datos incorrectos. No contienen los resultados inesperados, los valores atípicos ni las condiciones de límite necesarias para realizar la prueba de ETL y, en su lugar, se centran en las "rutas sin errores". De hecho, nuestras auditorías de datos de producción descubrieron que una cobertura de tan solo el 10 % y el 20 % constituye la norma. La ironía es que cuanto mejor se creó una rutina, se incorporan menos "datos incorrectos", lo que significa que existe una menor cantidad de datos de una variedad suficiente para probar por completo las reglas de ETL de cara al futuro.

¿Los datos están disponibles cuando los necesita?

Otro problema importante de la validación de ETL es la disponibilidad de los datos. Los datos de origen pueden extraerse de 50 orígenes diferentes en una empresa. El problema de la prueba de ETL, y de las pruebas en general, es que se la considera como una serie de etapas lineales, de modo que los equipos de pruebas se ven obligados a esperar los datos que otro equipo está usando.

Tomemos el ejemplo de una cadena de migración bancaria, donde los datos se extraen de un banco y se convierten en los sistemas de otros mediante una herramienta de reconciliación. En cada etapa, se deben validar los datos para comprobar que se convirtieron de forma correcta en el marco de control financiero, que se recuperó el número de cuenta, que hubo una exactitud de horas, etcétera. Este proceso puede implicar varias etapas diferentes, desde la entrada básica, a través de la desduplicación y preparación, hasta la propagación y el registro de los datos. Además, es posible que existan varios equipos involucrados, incluidos los equipos de ETL y los de no ETL, especialmente aquellos que trabajan en el mainframe.

Si los datos de cada dato de la empresa no están disponibles para cada equipo en paralelo, los retrasos aumentarán mientras los equipos esperan de brazos cruzados. Los evaluadores escribirán sus códigos fantasmas y luego no tendrán los datos de origen para validar una regla de ETL, ya que otro equipo la está utilizando. De hecho, observamos que los evaluadores promedio pueden pasar el 50 % del tiempo esperando, buscando, manipulando o creando datos. Esto puede constituir un gran porcentaje del 20 % del total del ciclo de vida de desarrollo de sistemas (SDLC).

¿Qué sucede cuando las reglas cambian?

La derivación manual de casos de prueba y de datos de requisitos estáticos es sumamente arreactiva al cambio. Las rutinas de ETL cambian a la velocidad que la empresa evoluciona, y el volumen y la variedad de datos que esta recolecta crecen con ella. Sin embargo, cuando se produce dicho cambio constante, la prueba ETL no puede seguir el ritmo.

Se puede decir que la principal causa de los retrasos de los proyectos en este caso es tener que revisar y actualizar casos de prueba existentes cuando las rutinas cambian. Los evaluadores no tienen ninguna manera de identificar de forma automática el impacto de un cambio realizado en los requisitos estáticos y los casos de prueba. En lugar de ello, tienen que revisar cada caso de prueba existente de forma manual, sin ningún otro modo de medir si realmente se mantuvo la cobertura.

Con el equipo de inteligencia empresarial mencionado más arriba, en los casos en los que los requisitos y los casos de prueba se almacenaron en documentos y hojas de cálculo escritos respectivamente, el cambio fue especialmente problemático. A un evaluador le llevó 7,5 horas revisar y actualizar un conjunto de casos de prueba cuando se realizó un cambio en una sola regla de ETL. En otra organización en la cual trabajamos, a los evaluadores les llevó dos días revisar cada caso de prueba existente cuando se realizó un cambio en los requisitos.

Sección 3

La alternativa viable: prueba de extracción, transformación y carga de datos totalmente automatizada

Por lo tanto, queda claro que siempre que los casos de prueba se obtengan manualmente y que los resultados se comparen de forma manual, la prueba de ETL no podrá seguir el ritmo de la velocidad con los que los requisitos empresariales cambian constantemente. A continuación, presentamos una estrategia posible para mejorar la eficiencia y la eficacia de la prueba de ETL. Es una estrategia basada en requisitos y en modelo, diseñada para desplazar a la izquierda el esfuerzo de la prueba y generar calidad en el ciclo de vida de la ETL desde el principio. Dicha estrategia Basada en modelo introduce la automatización en cada etapa de la prueba y desarrollo, a la vez que hace que la prueba de ETL sea totalmente reactiva al cambio constante.

1) Inicio con un modelo formal

Introducir el modelado formal en la prueba de ETL ofrece la ventaja fundamental de que desplaza a la izquierda el esfuerzo de la prueba, donde se pueden obtener todos los activos de evaluación/desarrollo a partir del esfuerzo inicial por asignar una regla ETL a un modelo. Por lo tanto, el modelo formal se convierte en el pilar de la validación de ETL totalmente automatizada.

Sin embargo, el modelado formal también puede ayudar a solucionar problemas más específicos de ambigüedad y de estado incompleto de los requisitos descritos anteriormente. Ayuda a mantener la comprobación a pesar del constante crecimiento de la complejidad de las reglas de ETL para que los evaluadores puedan comprender exactamente, y de forma rápida y visual, la lógica que debe probarse. Pueden conocer con facilidad tanto los datos válidos como no válidos que deben introducirse para probar por completo una regla de transformación y cuál debe ser el resultado esperado en cada caso.

Por ejemplo, un modelo de gráfico de flujo desglosa la documentación del "muro de palabras", de lo contrario, compleja en fragmentos digeribles. Reduce la ETL en la lógica de causa y efecto, y la asigna a una serie de sentencias condicionales "Si, entonces" vinculadas a una jerarquía de procesos.[15] En efecto, cada uno de estos pasos se convierte en un componente de prueba, y comunican al evaluador exactamente lo que se tiene que validar. Por lo tanto, modelar las rutinas de ETL como un gráfico de flujo elimina la ambigüedad de la documentación de requisitos, lo que evita el 56 % de los defectos que provienen de esta.

A medida que las rutinas de ETL se tornan cada vez más complejas, el gráfico de flujo funciona como un único punto de referencia. A diferencia de los gráficos y los documentos escritos "estáticos", se puede añadir fácilmente una lógica adicional al modelo. Además, la abstracción de rutinas muy complicadas es posible gracias a la tecnología de subflujos, lo que permite aumentar la capacidad de prueba. Los componentes inferiores pueden incorporarse en los flujos principales para que las distintas rutinas que conforman un conjunto muy complejo de reglas de ETL se puedan consolidar en un único diagrama visual.

Además de reducir la ambigüedad, el modelado del gráfico de flujo también ayuda a combatir el estado incompleto. Obliga al modelador a pensar en términos de limitaciones, condiciones negativas, restricciones y condiciones de límite, y a preguntarse "qué sucede si esta causa o este desencadenador no están presentes". Por lo tanto, deben idear de forma sistemática rutas negativas y adaptar la prueba negativa que debe constituir la mayor parte de la validación de ETL. Se pueden aplicar aún más los algoritmos de control de integridad, porque el modelo formal es un diagrama matemáticamente preciso de la regla de ETL.

Esto elimina la lógica faltante, como los "si no pendientes", para que se puedan obtener los casos de prueba que cubren el 100 % de las combinaciones posibles de datos (sin embargo, se debe tener en cuenta que casi siempre habrá más combinaciones que posiblemente puedan ejecutarse como pruebas; es por ello que las técnicas de optimización se analizarán más tarde). Otra gran ventaja es que los resultados esperados se pueden definir en el modelo, independientemente de los casos de prueba. En otras palabras, con un gráfico de flujo, el usuario puede definir el modelo para incluir las entradas límite y propagar el resultado esperado a puntos terminales diferentes en el modelo. Esto define claramente qué debe aceptar y rechazar una regla de validación para que los evaluadores no asuman equivocadamente que las pruebas pasaron cuando el resultado esperado no es explícito.

Debe señalarse que la adopción de la Prueba basada en modelo para la validación de la ETL no requiere la adopción completa de una estrategia basada en requisitos para la prueba y el desarrollo de toda la empresa. No exige un completo cambio radical y, en nuestra experiencia, lleva tan solo 90 minutos modelar una rutina de ETL como un gráfico de flujo "activo". Luego, el equipo de ETL o de pruebas puede usar este modelo con el único fin de probar y volver a probar la rutina.

2) Obtención automática de casos de prueba del modelo de gráfico de flujo

La introducción de la Prueba basada en modelo puede automatizar uno de los principales elementos manuales de la prueba de ETL: el diseño de los casos de prueba. Los evaluadores ya no necesitan escribir un código fantasma ni copiar manualmente SQL de la base de datos de origen en la de destino. En lugar de ello, las rutas a través del gráfico de flujo se convierten en los casos de prueba, que se pueden usar para bombear datos a través de las reglas de transformación. Estas se pueden obtener de forma sistemática, de una manera que no es posible cuando se escriben códigos a partir de los requisitos estáticos.

Esta derivación automática es posible porque el gráfico de flujo se puede superponer a toda la lógica funcional involucrada en un sistema. Luego pueden aplicarse los algoritmos matemáticos automatizados para identificar cada ruta posible a través del modelo, lo que permite generar casos de prueba que cubren cualquier combinación de entradas y salidas (se pueden usar la causa y efecto, o el análisis homotópico para hacer esto).

Dado que los casos de prueba están vinculados directamente al modelo, cubren toda la lógica definida en él. Por lo tanto, proporcionan una cobertura funcional del 100 %, de modo que usar un modelo de gráfico de flujo para obtener una documentación completa equivale a trabajar para probar de forma completa una rutina de ETL. Otra ventaja de este método es que la prueba se torna mensurable. Dado que se puede obtener cualquier caso de prueba posible, los evaluadores pueden determinar exactamente cuánta cobertura funcional proporciona un determinado conjunto de casos de prueba.

Optimización: pruebe más en menos pruebas

Luego pueden aplicarse los algoritmos de optimización automatizada para reducir al mínimo absoluto el número de casos de prueba, a la vez que se retiene la cobertura funcional máxima. Estas técnicas combinatorias son posibles en virtud de la estructura lógica del gráfico de flujo, donde un único paso en la lógica de causa y efecto (un bloque en el componente de la prueba/del gráfico de flujo) puede presentarse en varias rutas a través del flujo. Probar por completo la rutina de ETL se torna, por lo tanto, en un asunto de probar cada bloque individual (operador) mediante el uso de una de las técnicas de optimización existentes (Todos los bordes, Todos los nodos, Todos los bordes entrantes/salientes, Todos los pares). Por ejemplo, un conjunto de 3 casos de prueba puede, en realidad, ser suficiente para probar por completo la lógica involucrada en 5 rutas.

En la compañía de servicios financieros mencionada arriba, esto resultó ser extremadamente valioso para reducir la sobreprueba, acortar los ciclos de prueba y mejorar la calidad de la prueba. Incluido el tiempo que conlleva modelar el gráfico de flujo, por ejemplo, llevó 40 minutos crear 19 casos de prueba con una cobertura del 95 % (en comparación con los 150 casos de prueba con una sobreprueba del 80 % y 18 % utilizada anteriormente). En otro proyecto, llevó 2 horas crear 17 casos de prueba con una cobertura del 100 %: una mejora drástica en la cobertura del 16 % lograda en 6 horas anteriormente.

3) Creación automática de datos necesarios para ejecutar pruebas

Una vez que se crean los casos de prueba, los evaluadores requieren datos que puedan cubrir el 100 % de las pruebas posibles a fin de ejecutarlas. Dichos datos también se pueden obtener directamente del propio modelo, y pueden crearse de forma automática o extraerse de varios orígenes simultáneamente.

Un motor sintético de generación de datos, como CA Test Data Manager, ofrece varias maneras de crear los datos requeridos al usar la Prueba basada en modelo para impulsar la prueba de ETL. Esto se debe a que, además de la lógica funcional, un gráfico de flujo también se puede superponer a todos los datos involucrados en el sistema. En otras palabras, a medida que se modelan las reglas de ETL, se pueden definir los nombres de salida, las variables y los valores predeterminados para cada nodo individual. Cuando se crean los casos de prueba, los datos requeridos para ejecutarlos pueden generarse de forma automática a partir de los valores predeterminados, junto con los resultados esperados relevantes.

Alternativemente, usando CA Agile Requirements Designer (anteriormente Grid Tools Agile Designer), los datos del sistema se pueden crear rápidamente usando la herramienta Data Painter. Esto brinda una lista completa de funciones combinables de generación de datos, tablas de inicialización, variables del sistema y variables predeterminadas. Estos pueden utilizarse para crear datos que cubran cualquier escenario posible, incluidos los "datos incorrectos" y las rutas negativas. Dado que cada ruta es simplemente otro punto de datos, los datos requeridos a fin de probar de forma sistemática la capacidad de una rutina de ELT para rechazar datos no válidos pueden crearse de forma totalmente sintética.

Finalmente, los datos existentes se pueden encontrar en varios sistemas back-end en minutos mediante el uso de la minería de datos automatizada. Esta usa un análisis estadístico para detectar patrones en las bases de datos a fin de extraer grupos de patrones, dependencias o registros inusuales.

4) Aprovisionamiento de datos en minutos, que "se corresponden" con las pruebas correctas

Normalmente, una combinación de datos existentes, de producción y sintéticos es preferible en el caso de que la generación sintética se utilice para llevar la cobertura hasta el 100 %. Lo que es fundamental para una prueba de ETL eficiente es que los datos de la "Copia de oro" se almacenan de forma inteligente para que se puedan solicitar, clonar y entregar en paralelo los mismos conjuntos de datos. Esto, por lo tanto, elimina, los retrasos causados por las restricciones de datos.

El primer paso para almacenar de forma inteligente los datos es crear un Test Mart donde los datos "se corresponden" con pruebas específicas. Se asignan datos exactos a cada prueba, mientras que los datos se corresponden con criterios estables y definidos, no claves específicas. Los datos de la Correspondencia de pruebas pueden eliminar el tiempo que, de lo contrario, se emplearía en buscar datos en un gran origen de datos de producción, ya que los datos se pueden recuperar de forma automática del Almacenamiento de datos de las pruebas o se pueden extraer en minutos de varios sistemas back-end.

El Almacenamiento de datos de las pruebas funciona como una biblioteca central en la que se almacenan datos como activos reutilizables, junto con las consultas asociadas requeridas para extraerlos. Los grupos de datos luego se pueden solicitar y recibir en minutos, vinculados a los casos de prueba correctos y a los resultados esperados. Cuantas más pruebas se ejecutan, más grande se torna esta biblioteca, hasta que prácticamente cada solicitud de datos se pueda realizar en una fracción del tiempo.

Fundamentalmente, los datos pueden incorporarse en varios sistemas de forma simultánea y se clonan a medida que se aprovisionan. Esto significa que los conjuntos de datos provenientes de numerosas bases de datos de origen están disponibles para varios equipos en paralelo. La prueba de ETL ya no es un proceso lineal y los retrasos prolongados generados por las restricciones de datos se eliminan. Los datos originales pueden mantenerse a medida que se realizan cambios en el modelo, lo que les permite a los equipos trabajar en varias ediciones y versiones en paralelo. Este "control de versiones" también significa que los cambios realizados en las rutinas de ETL se reflejan de forma automática en los datos, lo que brinda a los equipos de pruebas los datos actualizados que necesitan para probar de manera estricta las trasformaciones de la prueba.

5) Ejecución de los datos contra las reglas y comparación automática de los resultados

Una vez que los evaluadores tienen las pruebas necesarias para probar por completo una rutina de ETL y los datos necesarios para ejecutarlas, la validación también se debe automatizar en el caso de que la prueba de ETL pueda seguir el ritmo de las necesidades cambiantes.

Uso de un motor de organización de datos

Una manera de hacer esto es usar un motor de automatización de pruebas. CA Technologies ofrece la ventaja de duplicar como un motor de organización de datos, lo que significa que se pueden tomar y bombear los datos, que se corresponden con pruebas específicas y resultados esperados, a través de una regla de validación. Esta es una "Automatización basada en datos" donde, por ejemplo, un agente de prueba creado en un motor de organización de datos puede tomar cada fila de un archivo XML, definido en el gráfico de flujo, y ejecutarla como una prueba.

Esto proporciona un resultado de aprobación o error según los resultados esperados definidos en el modelo. Esto automatiza tanto la ejecución de las pruebas como la comparación de los resultados reales con los esperados. Los evaluadores ya no tienen que copiar de forma manual los scripts del destino de datos en el origen de datos, y también pueden evitar el arduo proceso propenso a errores de tener que comparar cada campo individual generado por la transformación.

Uso de CA Test Data Manager

Por otro lado, una vez que se hayan definido los resultados esperados, se puede usar CA Test Data Manager para crear de forma automática reportes de errores y aprobación en función de estos. Esto automatiza las comparaciones manuales de datos, pero aún se debe copiar SQL del origen en el destino.

Primero, se definen los datos sintéticos en el sistema de origen mediante las distintas técnicas descritas arriba. Luego, puede crearse un Grupo de datos para simular el proceso de ETL, copiando datos del origen en el destino. Se puede copiar un conjunto válido de datos para probar que no se rechazan, así como también un conjunto de datos con errores para garantizar que se rechazan los datos no válidos. Al crear una tabla más para almacenar las Condiciones de prueba y los resultados, y al crear una Tabla basada en grupos de datos con Casos de prueba y Condiciones de datos, se pueden comparar de forma automática los resultados esperados con los resultados reales. Así, los datos erróneos o faltantes pueden identificarse rápidamente.

6) Implementación automática del cambio

Una de las mayores ventajas de la Prueba basada en modelo para la validación de la ETL es la capacidad de reaccionar al cambio de gran velocidad. Dado que los casos de pruebas, datos y requisitos están tan estrechamente vinculados, un cambio realizado en el modelo puede reflejarse de forma automática en los casos de prueba y los datos asociados. Esta rastreabilidad significa que, a medida que las rutinas de ETL se tornan rápidamente cada vez más complejas, la prueba puede seguir el ritmo y no genera cuellos de botella en la canalización de Application Delivery.

Con el modelado del gráfico de flujo, la implementación de un cambio se vuelve tan rápida y simple como añadir un nuevo bloque al gráfico de flujo. Por lo tanto, se pueden aplicar algoritmos de control de integridad para validar el modelo y garantizar que cada fragmento de la lógica se conectó correctamente a un flujo completo. Si se usa CA Agile Requirements Designer, puede usarse Path Impact Analyzer a fin de identificar aún más el impacto del cambio en las rutas a través del gráfico de flujo. Luego, se pueden eliminar o reparar de forma automática los casos de prueba afectados, y todas las pruebas nuevas deben retener una cobertura funcional del 100 % generada automáticamente.

Esto elimina el tiempo desperdiciado en revisar y actualizar las pruebas de forma manual, mientras que la desduplicación automatizada demostró acortar los ciclos de prueba un 30 %. En el equipo de inteligencia empresarial mencionado arriba, la Prueba basada en modelo introdujo un gran ahorro de tiempo en el proceso de ETL. Mientras que llevó 7,5 horas revisar y actualizar los casos de prueba al cambiar una sola regla de ETL, a CA Agile Requirements Designer solo le llevó 2 minutos. Además, del total de casos de prueba, solo 3 se vieron realmente afectados y, por lo tanto, se debió actualizarlos.

Como se describió, el control de versiones de datos también significa que a los evaluadores se les proporciona los datos actualizados que necesitan para probar por completo una rutina de ETL, incluso después de que esta cambia. Dado que los datos de prueba se pueden rastrear en el modelo, cuando los requisitos cambian, los cambios se reflejan de forma automática en los conjuntos de datos correspondientes. Los datos están disponibles en las ediciones y las versiones en paralelo, mientras que los datos requeridos para una prueba de regresión eficiente se preservan en el Almacenamiento de datos de las pruebas. Los “datos incorrectos”, interesantes o raros también se pueden bloquear para que otro equipe deje de consumirlos y evitar que se pierdan durante una actualización de datos.

Sección 4

Resumen

Introducir un mayor grado de automatización en la prueba de ETL es fundamental para cualquier organización que se esfuerza por brindar una entrega continua de un software de alta calidad. Un grado especialmente alto de esfuerzo manual permanece en la validación de ETL, de un código fantasma escrito de forma manual a requisitos estáticos, del suministro de los datos requeridos a la comparación de los resultados. La Prueba basada en modelo y la Administración inteligente de los datos de las pruebas pueden usarse para automatizar cada una de estas tareas, al mismo tiempo que permiten que numerosos equipos trabajen con los mismos orígenes de datos en paralelo.

La Prueba basada en modelo "desplaza a la izquierda" el esfuerzo de la prueba de ETL, y concentra la mayor parte del trabajo en la fase de diseño. Desde allí, cada activo requerido para la prueba de ETL puede obtenerse de forma automática en una fracción del tiempo. Los casos de prueba que proporcionan una cobertura funcional del 100 % se pueden generar de forma automática y están vinculados a resultados esperados definidos de forma independiente. Cada prueba se "Corresponde" aún más con los datos de origen exactos necesarios para ejecutarlas que, si se almacenan en el Almacenamiento de datos de las pruebas, se pueden aprovisionar a numerosos equipos en paralelo.

Dado el vínculo estrecho generado entre las pruebas, los datos y los requisitos, las pruebas requeridas para volver a probar una rutina de ETL se pueden ejecutar de forma rápida después de realizar un cambio. El tiempo desperdiciado al crear el modelo inicial, por lo tanto, se compensa por el tiempo ahorrado en comparación con crear, actualizar y volver a probar las pruebas de forma manual. La generación Basada en modelo también ofrece el beneficio sustancial de la reutilización, donde los componentes de las pruebas se pueden almacenar como activos que se pueden compartir, vinculados a los datos y los resultados esperados, en el Almacenamiento de datos de la pruebas. Cuantas más pruebas se ejecutan, más crece la biblioteca, hasta que las reglas de ETL actualizadas o nuevas de la prueba se tornan tan rápidas y fáciles como seleccionar de los componentes existentes.

La prueba de ETL ya no crea cuellos de botella en la entrega de aplicaciones y puede seguir el ritmo del crecimiento de las empresas basadas en datos. La comprobación de rutinas cada vez más complejas se mantiene para que la prueba pueda controlar la variedad y el volumen de datos recolectados, y no evita la entrega continua de las aplicaciones de calidad.

Comuníquese con CA Technologies en ca.com/ar.

CA Technologies (NASDAQ: CA) crea un software que impulsa la transformación en las empresas y les permite aprovechar las oportunidades de la economía de la aplicación. El software es el centro de cada empresa, en cada sector. Desde la planificación hasta el desarrollo, la administración y la seguridad, CA trabaja con empresas en todo el mundo para cambiar el estilo de vida, realizar transacciones y comunicarse, mediante entornos móviles, de nubes públicas y privadas, distribuidos y centrales. Para obtener más información sobre los programas para el éxito del cliente, visite **ca.com/customer-success**. Obtenga más información en **ca.com/ar**.

1 Jean-Pierre Dijcks, *Why does it take forever to build ETL processes?*, consultado el 24/07/2015 en **https://blogs.oracle.com/datawarehousing/entry/why_does_it_take_forever_to_bu**

2 Alan R. Earls, *The State of ETL: Extract, Transform and Load Technology*, consultado el 21/07/2015 en **http://data-informed.com/the-state-of-etl-extract-transform-and-load-technology/**

3 IBM, consultado el 20/07/2015 en **http://www-01.ibm.com/software/data/bigdata/what-is-big-data.html**

4 Jacek Becla and Daniel L. Wang, Lessons Learned from managing a Petabyte, P. 4. Consultado el 19/02/2015 en **http://www.slac.stanford.edu/BFROOT/www/Public/Computing/Databases/proceedings/**

5 **http://etlcode.com/index.php/utility/etl_complexity_calculator**

6 Jean-Pierre Dijcks, *Why does it take forever to build ETL processes?*, consultado el 24/07/2015 en **https://blogs.oracle.com/datawarehousing/entry/why_does_it_take_forever_to_bu**

7 ETL Guru, *ETL Startegy to store data validation rules*, consultado el 22/07/2015 en **http://etlguru.com/?p=22**

8 Bender RBT, *Requirements Based Testing Process Overview*, consultado el 05/03/2015 en **http://benderrbt.com/Bender-Requirements%20Based%20Testing%20Process%20Overview.pdf**

9 Huw Price, *Test Case Calamity*, consultado el 21/07/2015 en **https://communities.ca.com/community/ca-agile-requirements-designer/blog/2016/02/24/test-case-calamity**

10 Bender RBT, *Requirements Based Testing Process Overview*

11 Software Testing Class, *Why testing should start early in software development life cycle?*, consultado el 06/03/2015 en **http://www.softwaretestingclass.com/why-testing-should-start-early-in-software-development-life-cycle/**

12 datagaps, *ETL Testing Challenges*, consultado el 24/07/2015 en **http://www.datagaps.com/etl-testing-challenges**

13 Robin F. Goldsmith, *Four Tips for Effective Software Testing*, consultado el 20/07/2015 en **http://searchsoftwarequality.techtarget.com/photostory/4500248704/Four-tips-for-effective-software-testing/2/Define-expected-software-testing-results-independently**

14 Jagdish Malani, ETL: *How to handle bad data*, consultado el 24/07/2015 en **http://blog.aditi.com/data/etl-how-to-handle-bad-data/**

15 Philip Howard, *Automated Test Data Generation Report*, P. 6. Consultado el 22/07/2015 en **http://www.agile-designer.com/wpcms/wp-content/uploads/2014/10/00002233-Automated-test-case-generation.pdf**