# INDICE

# QuickWin - Componente de Activación de Ofertas

## INTRODUCCIÓN

**QuickWin** es un componente que permite o impide, según sea el caso, activar, inactivar, actualizar o realizar una operación sobre las ofertas disponibles en una campaña vigente de Claro, mediante un proceso de evaluación de condiciones y validación de políticas o reglas previamente configuradas en el sistema. Para ello, **QuickWin** cuenta con un modelo flexible, en el que es posible configurar las campañas de Claro y el comportamiento de las mismas, además de la configuración de las ofertas y productos que la conforman.

## BENEFICIOS y OBJETIVOS DE QuickWin

- Configuración y publicación ágil de ofertas de productos y servicios del Operador Celular.
- Mejorar Time To Market de iniciativas de negocio ofreciendo “core” funcional enriquecido para la mayoría de retos comunes en el proceso de publicación de ofertas de servicios.
- Tecnología de vanguardia mediante la explotación de Arquitectura JEE, estándares de la industria y patrones de diseño.
- Estabilidad y Disponibilidad de servicio.
- Flexibilidad

## CARACTERÍSTICAS FUNCIONALES

A continuación se describe la funcionalidad provista por el componente de activación de ofertas QuickWin.

## CAMPAIGN O CAMPAÑAS

Una campaña representa un período situacional en el cual se desea liberar un proceso de venta de productos y ofertas a los suscriptores. Ejemplo de campaña serían: Navidad, Día de la Madre, Campaña Regular diaria, Martes 2x1, entre otros.


**CAMPANA: Período en el que se libera un proceso de venta de productos.**

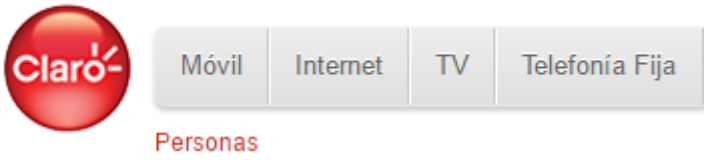
# OFFER/OFERTA

Una oferta es una composición de PRODUCTOS o SERVICIOS que se ofrecen al cliente como un paquete con costo o sin costo asociado. Una campaña puede publicar a los clientes varias OFERTAS. Se debe considerar que una OFERTA puede contener uno o más PRODUCTOS. Las ofertas tienen una naturaleza a la que llamaremos TIPO de OFERTA.


**Composición de productos a ofrecer.**

# TIPOS DE OFERTA / OFFER TYPE

El tipo de oferta define una clasificación que permite asociar varias ofertas de la misma naturaleza. Un ejemplo válido para Offer Type sería “Ofertas de Paquetes de Datos”. Podrían existir diferentes ofertas de la misma naturaleza, por ejemplo “Paquete de Internet” es una “Oferta de Paquetes de Datos” y “Paquete de Facebook” es una “Oferta de Paquetes de Datos”. En este ejemplo ambas ofertas son del tipo “Paquetes de Datos”.




**Clasificación que permite asociar varias ofertas de la misma naturaleza.**

# PRODUCTOS / PRODUCTS

Debe entenderse como producto un ítem del inventario de servicios que puede ofrecerse al cliente. Ejemplos de productos son “Paquete de Internet de 100 MB para Smartphone”, “Paquete de 100 Minutos de VOZ”. Debe considerarse que un producto puede asociarse a una o más OFERTAS para que sean parte de un paquete completo de servicios.

Dentro de la configuración de los productos, se contempla la expiración de los productos entregados/adquiridos.

| Megas | Precio | GRATIS Whatsapp | GRATIS Facebook | Vigencia | Mi Claro |
|---|---|---|---|---|---|
| 50 | $ 1 | - | - | 1 DIA | Activar |
| 100 | $ 2 | 5 MB | 15 MB | 2 DIAS | Activar |
| 200 | $ 3 | 5 MB | 15 MB | 3 DIAS | Activar |
| 400 | $ 5 | 20 MB | 80 MB | 7 DIAS | Activar |
| WHATSAPP | $ 1 | - | - | 1 DIA | Activar |

**Ejemplos de Productos**

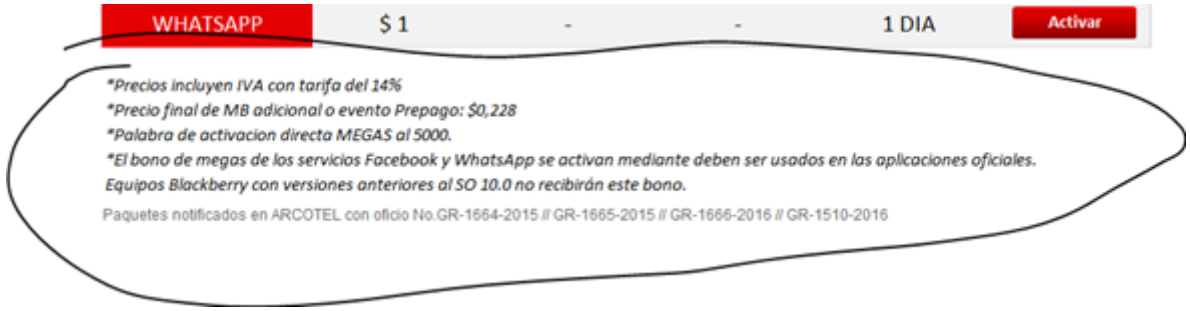
# TIPOS DE PRODUCTO / PRODUCT TYPE

El tipo de producto define una clasificación que permite asociar varios productos de la misma naturaleza. Un ejemplo válido para Product Type sería “Paquetes de Internet”. Podrían existir diferentes productos de la misma naturaleza, por ejemplo “Paquete 100 MB Internet” es un “Paquete de Internet” y “Paquete de 200 MB” también es un “Paquete de Internet”. En este ejemplo ambas ofertas son del tipo “Paquete de Internet”.


**Ejemplo de TIPOS DE PRODUCTOS en CLARO**

# COMPORTAMIENTO / BEHAVIOUR

Cada campaña tiene asociado uno y sólo un COMPORTAMIENTO. El comportamiento representa la conjunción de las políticas de venta y servicio. Podemos entender por COMPORTAMIENTO como el conjunto de políticas y reglas del juego para la comercialización de las ofertas.




**BEHAVIOUR: Políticas de ventas y reglas del juego para la comercialización de las ofertas.**

# TIPOS DE OPERACIONES / OPERATION TYPES

Un tipo de operación es todo aquello que un cliente puede realizar con una oferta, ejemplos de tipos de OPERACIONES son: ACTIVAR OFERTA, VENDER OFERTA, MODIFICAR OFERTA, ELIMINAR OFERTA, etc.

Una operación es una acción que el usuario puede realizar, por ejemplo ACTIVAR una oferta.


## COMPORTAMIENTO DE OPERACIONES/ BEHAVIOUR_OPERATION

Dado que a cada campaña le corresponde un Behaviour. Los Behaviour se combina con las Tipos de Operaciones. Esto da lugar a que podamos definir un nuevo comportamiento para cada operacion de la campaña por separado.

## STAGES/ ETAPAS.

Representan una funcionalidad a configurarse en **QuickWin** para un flujo de Operación de Campaña.

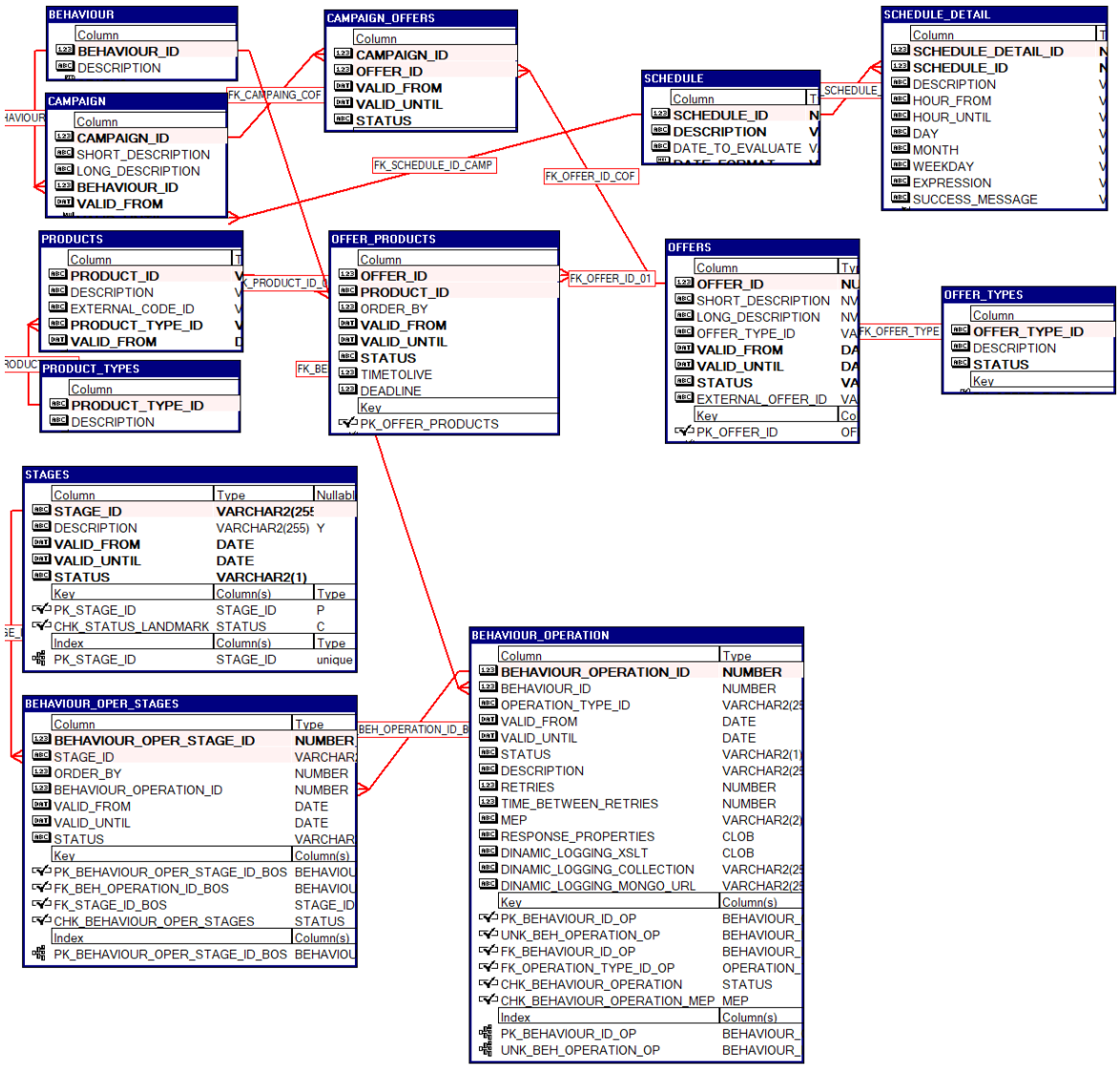
- Bloqueo.
- Validación.
- Ejecutar operación.
- Aprovisionar.

## COMPORTAMIENTO POR OPERACIONES/BEHAVIOUR_OPER_STAGE

Representa una lista de Operación de Campañas y cada una de ellas define un flujo a seguir en la operación, definido por STAGES.

# ANATOMIA DE CAMPAÑAS.

## MODELO DE ENTIDADES




Podemos observar que la campaña de forma opcional, se asocia a una configuración de Schedule. Esto es en caso que queremos mantener activa una campaña en fechas especificas. Para mas detalle de Schedule revisar la sesión de Schedule.

# 1.- QUICKWIN_MDS.BEHAVIOUR

| NAME | TYPE | NULLABLE | DEFAULT |
|---|---|---|---|
| BEHAVIOUR_ID | NUMBER | N | |
| DESCRIPTION | VARCHAR2(255) | Y | |
| STATUS | VARCHAR2(1) | Y | 'A' |

**BEHAVIOUR_ID**: Id de Behaviour.

**DESCRIPTION**: Descripción.

**STATUS**: Estado A de Activo, I de Inactivo.

# 2.- QUICKWIN_MDS.CAMPAIGN.

En esta tabla asociamos el **behaviour** a la campaña.

| NAME | TYPE | NULLABLE | DEFAULT |
|---|---|---|---|
| CAMPAIGN_ID | NUMBER | N | |
| SHORT_DESCRIPTION | VARCHAR2(255) | Y | |
| LONG_DESCRIPTION | VARCHAR2(255) | Y | |
| BEHAVIOUR_ID | NUMBER | N | |
| VALID_FROM | DATE | N | SYSDATE |
| VALID_UNTIL | DATE | N | to_date('20/03/2987 23:59:59','DD/MM/YYYY HH24:MI:SS') |
| STATUS | VARCHAR2(1) | N | 'A' |
| SCHEDULE_ID | NUMBER | Y | |

**CAMPAIGN_ID**: id numérico.

**SHORT_DESCRIPTION**: Descripción corta.

**LONG_DESCRIPTION**: Descripción larga.

**BEHAVIOUR_ID**: Id behaviour asociado a esta campaña.

**VALID_FROM**: Desde cuando es aplica.

**VALID_UNTIL**:  Hasta cuando es aplica.

**STATUS**: Estado A de Activo, I de Inactivo.

**SCHEDULE_ID**:  Id de Schedule es opcional. Indica que de acuerdo al Schedule configurado, cuando debe ejecutarse la campaña. Por decir,solo los martes de 8 a 1pm. Los 31 de cada mes, etc. Para mayor información revisar información de configuración de Schedule. Dado que por defecto no hay Schedule configurado, la campaña esta disponible durante todo el tiempo configurado por **VALID_FROM y VALID_UNTIL y STATUS.**

# 3.- QUICKWIN_MDS.OFFER_TYPES

| NAME | TYPE | NULLABLE | DEFAULT |
|---|---|---|---|
| OFFER_TYPE_ID | VARCHAR2(255) | N | |
| DESCRIPTION | VARCHAR2(255) | Y | |
| STATUS | VARCHAR2(1) | N | 'A' |

**OFFER_TYPE_ID**: Tipo de Oferta.

**DESCRIPTION**: Descripción.

**STATUS**: Estado A de Activo, I de Inactivo.

# 4.- QUICKWIN_MDS.OFFERS

| NAME | TYPE | NULLABLE | DEFAULT |
|---|---|---|---|
| OFFER_ID | NUMBER | N | |
| SHORT_DESCRIPTION | NVARCHAR2(255) | Y | |
| LONG_DESCRIPTION | NVARCHAR2(255) | Y | |
| OFFER_TYPE_ID | VARCHAR2(255) | Y | |
| VALID_FROM | DATE | N | SYSDATE |
| VALID_UNTIL | DATE | N | to_date('20/03/2987 23:59:59','DD/MM/YYYY HH24:MI:SS') |
| STATUS | VARCHAR2(1) | N | 'A' |
| EXTERNAL_OFFER_ID | VARCHAR2(255) | Y | |

**OFFER_ID**: Es la oferta que espera **QuickWin**. Por defecto toda o mínima configuración en **QuickWin** debe terminar **APROVISIONANDO** (**PROVISIONING**). Aprovisionar representa un STAGE en **QuickWin** donde podemos indicarle que de la oferta recibida vamos a aprovisionar todos los productos que en ella contenga. En este caso para aplicar esta regla, el campo de **productId** debe estar vacío. Caso contrario de esa oferta solo otorgamos el producto que se indica. Para mayor información, revisar configuración de **STAGES**.

**SHORT_DESCRIPTION**: Descripción corta.

**LONG_DESCRIPTION**: Descripción larga.

**VALID_FROM**: Desde cuando es aplica.

**VALID_UNTIL**:  Hasta cuando es aplica.

**STATUS**: Estado A de Activo, I de Inactivo.

**EXTERNAL_OFFER_ID**: Cuando se configura indica que la **OFFER_ID** va a hacer reconocida por esta.

# 5.- QUICKWIN_MDS.CAMPAIGN_OFFERS

| NAME | TYPE | NULLABLE | DEFAULT |
|---|---|---|---|
| CAMPAIGN_ID | NUMBER | N | |
| OFFER_ID | NUMBER | N | |
| VALID_FROM | DATE | N | SYSDATE |
| VALID_UNTIL | DATE | N | to_date('20/03/2987 23:59:59','DD/MM/YYYY HH24:MI:SS') |
| STATUS | VARCHAR2(1) | N | 'A' |

**CAMPAIGN_ID**: Id de Campaña.

**OFFER_ID**: Id de Oferta que aplican para esta campaña. Nótese que para una campaña se pueden configurar múltiples ofertas.  La configuración de oferta también aplican para el **STAGE PROVISIONING**. Sin embargo, a pesar que no se use o no es mandante el **STAGE PROVISIONING** es obligatorio crearlas. El aprovisionamiento de un producto se puede hacer de otra manera usando el **STAGE OPER_INVOKE**. **QuickWin** debe mantener el concepto de que una operación entregó una oferta  **X** en sus logs. En el contrato de **QuickWin** solo se espera una sola oferta a la vez y esa oferta debe estar registrada.

**VALID_FROM**: Desde cuando es aplica.

**VALID_UNTIL**:  Hasta cuando es aplica.

**STATUS**: Estado A de Activo, I de Inactivo.

# 6.- QUICKWIN_MDS.PRODUCT_TYPES

| NAME | TYPE | NULLABLE | DEFAULT |
|---|---|---|---|
| PRODUCT_TYPE_ID | VARCHAR2(255) | N | |
| DESCRIPTION | VARCHAR2(255) | Y | |
| STATUS | VARCHAR2(1) | N | 'A' |

**PRODUCT_TYPE_ID**: Id de Tipo de Producto. Normalmente el aprovisionamiento de un tipo de producto sigue un flujo de invocaciones de servicios web. Es obligatorio crearlo aunque no se lo use por defecto. Se utiliza en la tabla **PRODUCT_TYPE_PROVISIONING** que es quien define una **"RECETA"** a seguir por cada tipo de producto. Esto es en el caso de que se esté configurando el **STAGE PROVISIONING**.

**Nota: la tabla PRODUCT_TYPE_PROVISIONING** aplica la misma receta  de aprovisionamiento a todos los productos que están relacionas a un tipo de producto. Esto se hace automáticamente cuando se configura el STAGE PROVISIONING. Para mayor información revisar Configuración de **STAGES**.

**DESCRIPTION**: Descripción.

**STATUS**: Estado A de Activo, I de Inactivo.

# 7.- QUICKWIN_MDS.PRODUCTS

| NAME | TYPE | NULLABLE | DEFAULT |
|---|---|---|---|
| PRODUCT_ID | VARCHAR2(255) | N | |
| DESCRIPTION | VARCHAR2(255) | Y | |
| EXTERNAL_CODE_ID | VARCHAR2(255) | Y | |
| PRODUCT_TYPE_ID | VARCHAR2(255) | N | |
| VALID_FROM | DATE | N | SYSDATE |
| VALID_UNTIL | DATE | N | to_date('20/03/2987 23:59:59','DD/MM/YYYY HH24:MI:SS') |
| STATUS | VARCHAR2(1) | N | 'A' |

**PRODUCT_ID**: Es un nombre de producto.

**DESCRIPTION**: Descripción.

**EXTERNAL_CODE_ID**: Reemplaza al código interno de **PRODUCT_ID**.

**PRODUCT_TYPE_ID**: Tipo de producto al que pertenece.

**VALID_FROM**: Desde cuando es aplica.

**VALID_UNTIL**:  Hasta cuando es aplica.

**STATUS**: Estado A de Activo, I de Inactivo.

# 8.- QUICKWIN_MDS.OFFER_PRODUCTS

| NAME | TYPE | NULLABLE | DEFAULT |
|---|---|---|---|
| OFFER_ID | NUMBER | N | |
| PRODUCT_ID | VARCHAR2(255) | N | |
| ORDER_BY | NUMBER | Y | |
| VALID_FROM | DATE | N | SYSDATE |
| VALID_UNTIL | DATE | N | to_date('20/03/2987 23:59:59','DD/MM/YYYY HH24:MI:SS') |
| STATUS | VARCHAR2(1) | N | 'A' |
| ~~TIMETOLIVE~~ | ~~NUMBER~~ | ~~Y~~ | |
| ~~DEADLINE~~ | ~~NUMBER~~ | ~~Y~~ | |

**OFFER_ID**:  Id de Oferta.

**PRODUCT_ID**:  Productos que se asocian a la oferta.

**ORDER_BY**: Orden numérico.

**VALID_FROM**: Desde cuando es aplica.

**VALID_UNTIL**:  Hasta cuando es aplica.

**STATUS**: Estado A de Activo, I de Inactivo.

~~**TIMETOLIVE**~~: Deprecado.

~~**DEADLINE**~~: Deprecado.

# 9.- QUICKWIN_MDS.OPERATION_TYPES.

En esta tabla se identifican operaciones que pueden usarse. Sin embargo se pueden crear nuevas.

| NAME | TYPE | NULLABLE | DEFAULT |
|---|---|---|---|
| OPERATION_TYPE_ID | VARCHAR2(255) | N | |
| OPERATION_NAME | VARCHAR2(255) | N | |
| VALID_FROM | DATE | N | SYSDATE |
| VALID_UNTIL | DATE | N | to_date('20/03/2987 23:59:59','DD/MM/YYYY HH24:MI:SS') |
| STATUS | VARCHAR2(1) | N | 'A' |

**OPERATION_TYPE_ID**: Id de operación.

**OPERATION_NAME**: Nombre de la operación.

**VALID_FROM**: Desde cuando es aplica.

**VALID_UNTIL**:  Hasta cuando es aplica.

**STATUS**: Estado A de Activo, I de Inactivo.

# 10.- QUICKWIN_MDS.BEHAVIOUR_OPERATION

| NAME | TYPE | NULLABLE | DEFAULT | COMMENT |
|---|---|---|---|---|
| BEHAVIOUR_OPERATION_ID | NUMBER | N | | |
| BEHAVIOUR_ID | NUMBER | Y | | |
| OPERATION_TYPE_ID | VARCHAR2(255) | Y | | |
| VALID_FROM | DATE | Y | SYSDATE | |
| VALID_UNTIL | DATE | Y | to_date('20/03/2987 23:59:59','DD/MM/YYYY HH24:MI:SS') | |
| STATUS | VARCHAR2(1) | Y | 'A' | |
| DESCRIPTION | VARCHAR2(255) | Y | | |
| RETRIES | NUMBER | Y | 0 | Numero de Reitentos para este invoke en caso de falla |
| TIME_BETWEEN_RETRIES | NUMBER | Y | 1000 | Tiempo que debe esperar entre reintentar en Milisegundos |
| MEP | VARCHAR2(2) | Y | 'CD' | SY Sincronico (Aunque el consumidor envie otra letra en el contrato) AS Asincronico (Aunque el consumidor envie otra letra en el contrato) HB Hibrido (Aunque el consumidor envie otra letra en el contrato) CD Consumer Defined (Se hace lo que diga el consumidor) |
| RESPONSE_PROPERTIES | CLOB | Y | '[]' | Subscriber Properties que se mostraran en la respuesta de la peticion a Quickwin |
| ~~DINAMIC_LOGGING_XSLT~~ | ~~CLOB~~ | ~~Y~~ | | |
| ~~DINAMIC_LOGGING_COLLECTION~~ | ~~VARCHAR2(255)~~ | ~~Y~~ | | |
| ~~DINAMIC_LOGGING_MONGO_URL~~ | ~~VARCHAR2(255)~~ | ~~Y~~ | | |

---

**BEHAVIOUR_OPERATION_ID**: Id numérico.

**BEHAVIOUR_ID**: Id behaviour.

**OPERATION_TYPE_ID**: Id de Operación.

**VALID_FROM**: Desde cuando es aplica.

**VALID_UNTIL**:  Hasta cuando es aplica.

**STATUS**: Estado A de Activo, I de Inactivo.

**RETRIES**:  Indica que se desea alertar a la operación para que cada vez que la campaña o **QuickWin** responda con un código **NEGATIVO** ya sea por una invocación o condición, automáticamente aplique un reintento de forma **Asíncrona**. Sin no se desea reintentos, entonces el valor **0** es no aplica reintentos. Si es mayor a  0 estos reintentos se estarán registrando en la tabla de **OPERATION_HISTORY**, y se los identifica por  la columna **TRACKING_OPER_HISTORY_ID** y **RETRIES_COUNT** que es el numero de reintentos ejecutados. Si el numero de reintentos se

agota entonces terminan en una tabla **OPERATION_REPROCESSING** la cual sirve para que un operador los reintente después de corregir cualquier eventualidad o haga una nueva reinyección de reintentos . Existen tres formas de reinyectar que son a través del id de la operación de **QuickWin**, por fechas, pero reintenta todas las operaciones que allí apliquen a esa fecha (**falta filtra solo por operación**) y combinados. Es importante que cualquiera que esté interesado en reintentos tenga un monitoreo de la tabla **OPERATION_REPROCESSING** y también debe asegurarse que los reintentos no sean infinitos y que todos ellos deben haber compensado cualquier proceso antes de reintentarse por si solo. Para mayor información revisar documentación de reintentos de operaciones. Este feature es recomendable cuando el reintento no pide compensación previa.

**Antes de configurar reintentos tenga en cuenta lo siguiente**:

Algunos **STAGES** especialmente los que usan al **Invoke** manejan un feature llamado. **INVOKE_COMPENSATION_BACKWARD** Y **INVOKE_COMPENSATION_FORWARD**.

**Los reintentos de toda la operación no siempre aplican**, por lo cual existen reintentos que se hacen a nivel de bloques asíncronos, si este es el caso. Estos reintentos tienen el objetivo de reintentarse solo, por sí mismo y no reintenta toda la operación sino solo una invocación exclusiva de un servicio de forma asíncrona. Y si este ultimo no da éxito después de completar su numero de reintentos, así también termina en una tabla llamada **INVOKE_REPROCESSING** la cual sirve para que un operador los reintente después de corregir cualquier eventualidad o haga una nueva reinyección de reintentos. Este caso solo aplica si realmente no nos interesa la respuesta o estamos seguros que siempre debe devolver éxito, o es el ultimo de los servicios. Después de esto, la otra opción es otro **feature** llamado **COMPENSACION HACIA ADELANTE** y **HACIA ATRAS**. **La compensación hacia ADELANTE solo aplica cuando el numero de RETRIES es mayor a 0 y el reintento se recupera en el siguiente reintento.**

**Cómo funciona?**

Si **retries** es mayor a 0, y se ha configurado los Invoke que compensan hacia atrás o adelante (opcional) y una columna **APPLY_COMPENSATION** a YES, entonces se ejecuta el **feature**.

Sin embargo, si **retries** es 0, solo se ejecutará **COMPENSACION HACIA ATRAS.**

La compensación hacia atrás determina que si el invocador que estoy llamando falla después de todos el numero reintentos configurados(**RETRIES**), este llama a un servicio que compensa hacia atrás. Este debe tener la compensación N-1 servicios. En caso de que el servicio después de reintentar responde exitosamente antes de completar el numero de reintentos configurado, este continua al servicio de compensación hacia adelante si es que existe. La razón de hacer esto es porque los servicios cuando aplican reintentos se separan y quien compensa hacia adelante debe hacer el resto. La transacción de **QuickWin** termina reportando que se hace una compensación hacia atrás o delante. Este error es reportado al usuario pero la compensación se hace de forma asíncrona por si sola.

No podemos garantizar que la compensación también sea exitosa. Este servicio debe ser a prueba de fuego o debe monitorearse.

**TIME_BETWEEN_RETRIES**: Es el tiempo que debe ocurrir entre reintentos después de responder con **FALLA**. Por defecto 1 segundo. No se soporta más aun así se configure hasta ahora.

**DESCRIPTION**: Descripción.

**RESPONSE_PROPERTIES**: Especifica un arreglo de variables que pueden ser devuelta como parte del response como datas de Sesión. Por defecto es []. Se puede configurar de la siguiente manera en caso de requerirse.

```
[
  {
    "responseAlias": "ORDEN",
    "subscriberPropertyId": "ORDER_ID"
  }
]
```

**responseAlias**: Opcional. Indica si se desea cambiar el nombre del SUBSCRIBER_PROPERTIES devuelto.

**subscriberPropertyId**: Id de SUSCRIBER_PROPERTYIES.

La respuesta vendrá dentro de **sessionData**.

```
{
   "activateMessageResponse":    {
    ...
      "sessionData": {"externalSubscriberProperties": [      {
         "id": "ORDEN",
         "value": ["10000221222545544474744"]
      }]}
   },
   "commonHeaderResponse": {"operationInfo":    {
      "transactionId": "ce6bb706-1d6c-413b-a000-ea2481d53986",
      "transactionDate": "13/05/2021 09:00:59",
      "operationId": "166"
   }},
   "userMessageResponse":    {
     ...
   }
}
```

~~**DINAMIC_LOGGING_XSLT**~~: Deprecado.

~~**DINAMIC_LOGGING_COLLECTION**~~: Deprecado.

~~**DINAMIC_LOGGING_MONGO_URL**~~: Deprecado.

# 11.- QUICKWIN_MDS.STAGES

Esta es una tabla Precargada. Define todos los **Stages** o bloques de **QuickWin** que ejecutan alguna funcionalidad.

| NAME | TYPE | NULLABLE | DEFAULT |
|---|---|---|---|
| STAGE_ID | VARCHAR2(255) | N | |
| DESCRIPTION | VARCHAR2(255) | Y | |
| VALID_FROM | DATE | N | sysdate |
| VALID_UNTIL | DATE | N | to_date('20/03/2987 23:59:59','DD/MM/YYYY HH24:MI:SS') |
| STATUS | VARCHAR2(1) | N | 'A' |

**STAGE_ID**: Id del STAGE.

**DESCRIPTION**: Descripción.

**VALID_FROM**: Desde cuando es aplica.

**VALID_UNTIL**:  Hasta cuando es aplica.

**STATUS**: Estado A de Activo, I de Inactivo.

# 12.- QUICKWIN_MDS.BEHAVIOUR_OPER_STAGES

Esta tabla define el flujo de **STAGES** a seguir para una operación de campaña.

| NAME | TYPE | NULLABLE | DEFAULT |
|---|---|---|---|
| BEHAVIOUR_OPER_STAGE_ID | NUMBER | N | |
| STAGE_ID | VARCHAR2(255) | Y | |
| ORDER_BY | NUMBER | Y | 10 |
| BEHAVIOUR_OPERATION_ID | NUMBER | Y | |
| VALID_FROM | DATE | Y | SYSDATE |
| VALID_UNTIL | DATE | Y | to_date('20/03/2987 23:59:59','DD/MM/YYYY HH24:MI:SS') |
| STATUS | VARCHAR2(1) | Y | 'A' |

**BEHAVIOUR_OPER_STAGE_ID**:  Id numérico. Este ID será utilizado en las configuraciones de BEHAVIOUR_BLOCKING, BEHAVIOUR_VALIDATION, BEHAVIOUR_ENRICHMENT, BEHAVIOUR_ENRICHMENT_PROPERTY, BEHAVIOUR_PRODUCT, BEHAVIOUR_OPER_INVOKE. La tabla que se configura va a estar determinada por **STAGE_ID**. Si es DYNAMIC_BLOCKING será la tabla **BEHAVIOUR_BLOCKING**.

**ORDER_BY**: Orden numérico.

**BEHAVIOUR_OPERATION_ID**: Id de behaviour operación.

**VALID_FROM**: Desde cuando es aplica.

**VALID_UNTIL**:  Hasta cuando es aplica.

**STATUS**: Estado A de Activo, I de Inactivo.

Recuerde que, cada STAGE que se elija necesitara configurarse en sus respectivas tablas.

| STAGE_ID | TABLA DE CONFIGURACIÓN |
|---|---|
| CHARGE | BEHAVIOUR_PRICE |
| DYNAMIC_BLOCKING | BEHAVIOUR_BLOCKINGS |
| DYNAMIC_VALIDATION | BEHAVIOUR_VALIDATIONS |
| EXCLUSION | BEHAVIOUR_EXCLUSIONS |
| NOTIFICATION | BEHAVIOUR_NOTIFICATION |
| OFFER_ADDONS | BEHAVIOUR_OFFER_ADDONS |
| OPERATION_INVOKE | BEHAVIOUR_OPER_INVOKE |
| PROVISIONING | BEHAVIOUR_PRODUCT, PRODUCT_TYPE_PROVISIONING |
| TRIGGERS | NO DEFINIDO |

## Ejemplos.

```
select * from BEHAVIOUR_OPER_STAGES T WHERE T.BEHAVIOUR_OPERATION_ID = 46 ORDER BY 3
```

| BEHAVIOUR_OPER_STAGE_ID | STAGE_ID | ORDER_BY | BEHAVIOUR_OPERATION_ID |
|---|---|---|---|
| 292 | DYNAMIC_BLOCKING | 10 | **46** |
| 293 | DYNAMIC_BLOCKING | 20 | **46** |
| 294 | DYNAMIC_VALIDATION | 30 | **46** |
| 295 | PROVISIONING | 40 | **46** |

Para el ejemplo anterior, vemos que la **operación 46** debe realizar lógicas de Bloqueo,Bloqueo, Validación y finalmente Aprovisionamiento. Vemos que en los registros **292 y 293 usan el mismo STAGE_ID**. Esto es válido cuando se requiere que entre un STAGE y OTRO que puede ser del mismo tipo, ocurra un llamado a un servicio web o una inicialización de variables a través de alguna función. Cuando se requiere hacer eso, se deben configurar en las tablas **BEHAVIOUR_ENRICHMENT, y BEHAVIOUR_ENRICHMENT_PROPERTY** según sea el caso. A este proceso se lo conoce como enriquecer datos y ocurre antes de cada **STAGE**, siempre y cuando se lo configure. Para mayor información se debe revisar Configuración de STAGES.

# CREACION DE CAMPAÑA CON UN STAGE PLSQL

**Nota**: En este ejemplo solo estamos configurando UN solo **STAGE_ID** llamado **OPERATION_INVOKE** para UNA SOLA OPERACION DE CAMPAÑA (**BEHAVIOUR_OPERATION**), pero en la vida real, se necesitaran configurar varios en un orden especifico de ejecución.

No se continúa con la configuración en sus respectivas tablas. En este caso revisar la documentación respectiva de configuración de **STAGES**.

En este ejemplo Ud. puede modificar el **PLSQL** y agregar mas **STAGES** si así lo desea y configurar cada **STAGE** por separado en otro **PLSQL** o en el mismo. En esta documentación proveemos el **PLSQL** de como configurarlos en la sesión de **Configuración de STAGES**.

Antes de continuar echemos un vistazo a los **STAGES** disponibles. Los tachados están descontinuados.

# QUICKWIN_MDS.STAGES

| STAGE_ID | DESCRIPTION |
|---|---|
| NOTIFICATION | Notifications |
| OFFER_ADDONS | Offer Addons |
| **OPERATION_INVOKE** | Invoke any Operation |
| DYNAMIC_BLOCKING | Dynamic Blocking |
| DYNAMIC_VALIDATION | Dynamic Validation |
| ~~EXCLUSION~~ | ~~Exclusion~~ |
| PROVISIONING | Provisioning |
| ~~CHARGE~~ | ~~Charging~~ |
| ~~TRIGGERS~~ | ~~Triggers~~ |

Ahora procedemos a configurar la campaña.
**FAQ**: En caso de recibir el siguiente error, lea la instrucción.

```
ORA-00001: unique constraint (QUICKWIN_MDS.PK_PRODUCT_TYPE_ID) violated
-- No olvide cambiar a otro sufijo que no está siendo usado PR_SUFFIX
  VARCHAR2(11) := 'TEST_QW';
```

# SCRIPT PLSQL

## CREAR OPERACION DE CAMPAÑA CON UN STAGE DE OPERATION_INVOKE

```
-- Created on 18/12/2019 by MGARCIAR@CLARO.COM.EC
declare

  LN_BEHAVIOUR_ID          NUMBER;
  LN_BEH_OPERATION_ID      NUMBER;
  LN_BEH_OPER_STAGE_ID1    NUMBER;
  LN_CAMPAIGN_ID           NUMBER;
  LN_OFFER_ID1             NUMBER;
  PRODUCT_TYPE_ID1         VARCHAR2(200);
  PRODUCT_ID1              VARCHAR2(200);
```

```
  LN_OPERATION_TYPE_ID1   VARCHAR2(200) := 'activateOffer';
  -- PR_SUFFIX CAMBIAR PARA PRUEBAS
  PR_SUFFIX               VARCHAR2(11) := 'TEST1_QW';
  -- AGREGAR EL STAGE QUE SE DESEA
  lv_description_campaign varchar(255) := '[QWV1-MGR] ';
  LV_STAGE                VARCHAR2(200) := 'OPERATION_INVOKE';
  LV_DESCRIPTION          VARCHAR2(200) := 'TEST QUICKWIN';

begin


  SELECT NVL(MAX(B.BEHAVIOUR_ID),0) + 1 INTO LN_BEHAVIOUR_ID FROM BEHAVIOUR B;

  insert into behaviour (BEHAVIOUR_ID,DESCRIPTION) values (LN_BEHAVIOUR_ID,
lv_description_campaign||LV_DESCRIPTION);


  SELECT NVL(MAX(C.CAMPAIGN_ID),0) + 1 INTO LN_CAMPAIGN_ID FROM CAMPAIGN C;

  /*campaign*/
  insert into campaign (CAMPAIGN_ID, SHORT_DESCRIPTION, LONG_DESCRIPTION,
BEHAVIOUR_ID)
  values (LN_CAMPAIGN_ID, LV_DESCRIPTION,lv_description_campaign||
LV_DESCRIPTION,LN_BEHAVIOUR_ID);

  /*product_types*/
  PRODUCT_TYPE_ID1 := 'PRT_'||PR_SUFFIX;
  insert into product_types (PRODUCT_TYPE_ID, DESCRIPTION)
  values (PRODUCT_TYPE_ID1, lv_description_campaign || LV_DESCRIPTION);

  /*products*/
  PRODUCT_ID1 := 'PR_'|| PR_SUFFIX;
  insert into products (PRODUCT_ID, DESCRIPTION, PRODUCT_TYPE_ID) values
(PRODUCT_ID1, LV_DESCRIPTION, PRODUCT_TYPE_ID1);

  SELECT NVL(MAX(O.OFFER_ID),0) + 1 INTO LN_OFFER_ID1 FROM OFFERS O;
  -- OFFERS
  INSERT INTO OFFERS(OFFER_ID, SHORT_DESCRIPTION, LONG_DESCRIPTION,
OFFER_TYPE_ID) VALUES(LN_OFFER_ID1, LV_DESCRIPTION,lv_description_campaign ||
LV_DESCRIPTION,'');

  --CAMPAÑA OFERTAS
  INSERT INTO CAMPAIGN_OFFERS(CAMPAIGN_ID,OFFER_ID)
VALUES(LN_CAMPAIGN_ID,LN_OFFER_ID1);

  -- OFFER_PRODUCTS
  insert into OFFER_PRODUCTS(OFFER_ID,PRODUCT_ID,ORDER_BY)
values(LN_OFFER_ID1,PRODUCT_ID1,10);


  -- BEHAVIOUR_OPERATION
  SELECT NVL(MAX(BO.BEHAVIOUR_OPERATION_ID),0) + 1 INTO LN_BEH_OPERATION_ID FROM
BEHAVIOUR_OPERATION BO;

  -- AGREGAR TODOS LOS CAMPOS QUE SE NECESITEN
  insert into behaviour_operation (BEHAVIOUR_OPERATION_ID, BEHAVIOUR_ID,
OPERATION_TYPE_ID, DESCRIPTION)
```

```
  values (LN_BEH_OPERATION_ID, LN_BEHAVIOUR_ID, LN_OPERATION_TYPE_ID1,
LV_DESCRIPTION);

  -- Creacion del esqueleto o stages
  SELECT NVL(MAX(BS.BEHAVIOUR_OPER_STAGE_ID),0) + 1 INTO LN_BEH_OPER_STAGE_ID1
FROM BEHAVIOUR_OPER_STAGES BS;
  insert into behaviour_oper_stages (BEHAVIOUR_OPER_STAGE_ID, STAGE_ID,
ORDER_BY, BEHAVIOUR_OPERATION_ID)
  values (LN_BEH_OPER_STAGE_ID1, LV_STAGE,10,LN_BEH_OPERATION_ID);


 -- commit;
  dbms_output.put_line('ln_beh_operation_id number :=
'||LN_BEH_OPERATION_ID||';');

  dbms_output.put_line('---------------REFRESH---------------------------------
----');

dbms_output.put_line('http://192.168.37.146:8101/quickWin/clearCampaignById/'||L
N_CAMPAIGN_ID);
  dbms_output.put_line('---------------GUARDE ESTE SCRIPT POR MAYOR SEGURIDAD---
---------------------------------');

  dbms_output.put_line('---------------SELECT----------------------------------
---');
  dbms_output.put_line('SELECT * FROM CAMPAIGN T WHERE T.CAMPAIGN_ID =
'''||LN_CAMPAIGN_ID||''';');
  dbms_output.put_line('SELECT * FROM CAMPAIGN_OFFERS T WHERE T.CAMPAIGN_ID =
'''||LN_CAMPAIGN_ID||''';');
  dbms_output.put_line('SELECT * FROM BEHAVIOUR_OPERATION T WHERE
T.BEHAVIOUR_OPERATION_ID = '''||LN_BEH_OPERATION_ID||''';');
  dbms_output.put_line('SELECT * FROM BEHAVIOUR_OPER_STAGES T WHERE
T.BEHAVIOUR_OPERATION_ID = '''||LN_BEH_OPERATION_ID||''';');
  dbms_output.put_line('SELECT * FROM BEHAVIOUR_PRODUCT T WHERE
T.BEHAVIOUR_OPER_STAGE_ID = '''||LN_BEH_OPER_STAGE_ID1||''';');
  dbms_output.put_line('SELECT * FROM offer_products T WHERE  T.offer_id =
'''||LN_OFFER_ID1||''';');
  dbms_output.put_line('SELECT * FROM campaign_offers T WHERE   T.offer_id =
'''||LN_OFFER_ID1||''';');
  dbms_output.put_line('SELECT * FROM PRODUCT_TYPES T WHERE  T.PRODUCT_TYPE_ID =
'''||PRODUCT_TYPE_ID1||''';');
  dbms_output.put_line('SELECT * FROM PRODUCTS T WHERE   T.PRODUCT_TYPE_ID =
'''||PRODUCT_TYPE_ID1||''';');

  dbms_output.put_line('--------------UPDATE STATUS----------------------------
----------');
  dbms_output.put_line('update offer_products t set status = ''I'' where
t.offer_id = '''||LN_OFFER_ID1||''';');
  dbms_output.put_line('update campaign_offers t set status = ''I'' where
t.offer_id = '''||LN_OFFER_ID1||''';');
  dbms_output.put_line('update offers t set status = ''I'' where t.offer_id =
'''||LN_OFFER_ID1||''';');
  dbms_output.put_line('update products p set status = ''I'' where
p.product_type_id = '''||PRODUCT_TYPE_ID1||''';');
  dbms_output.put_line('update product_types p set status = ''I'' where
p.product_type_id = '''||PRODUCT_TYPE_ID1||''';');
  dbms_output.put_line('update campaign c set status = ''I'' where
c.behaviour_id = '''||LN_BEHAVIOUR_ID||''';');
```

```
  dbms_output.put_line('update behaviour a set status = ''I'' where
a.behaviour_id ='''||LN_BEHAVIOUR_ID||''';');


 dbms_output.put_line('--------------DELETE-------------------------------------
--');
 dbms_output.put_line('delete behaviour_oper_stages b where
b.behaviour_operation_id = '''||LN_BEH_OPERATION_ID||''';');
 dbms_output.put_line('delete behaviour_operation b where b.behaviour_id =
'''||LN_BEHAVIOUR_ID||''';');
 dbms_output.put_line('delete offer_products t where t.offer_id =
'''||LN_OFFER_ID1||''';');
 dbms_output.put_line('delete campaign_offers t where t.offer_id =
'''||LN_OFFER_ID1||''';');
 dbms_output.put_line('delete offers t where t.offer_id =
'''||LN_OFFER_ID1||''';');
 dbms_output.put_line('delete products p where p.product_type_id =
'''||PRODUCT_TYPE_ID1||''';');
 dbms_output.put_line('delete product_types p where p.product_type_id =
'''||PRODUCT_TYPE_ID1||''';');
 dbms_output.put_line('delete campaign c where c.behaviour_id =
'''||LN_BEHAVIOUR_ID||''';');
 dbms_output.put_line('delete behaviour a where a.behaviour_id
='''||LN_BEHAVIOUR_ID||''';');

 dbms_output.put_line('---------------REQUEST-----------------------------------
-----------');

 dbms_output.put_line('http://192.168.37.146:8101/quickWin/getActivateRequest');

 dbms_output.put_line('---------------POST REST---------------------------------
-----');

  dbms_output.put_line('{
  "commonHeaderRequest": {
    "channelInfo": {
      "channelId": "622",
      "mediaDetailId": "BES.15.101",
      "mediaId": "CRM.6"
    },
    "consumerInfo": {
      "companyId": "CLARO",
      "consumerType": "CLARO",
      "consumerId": "SPRXSLT",
      "terminal": "HUB PLATINUM"
    },
    "geolocationInfo": {
      "accuracy": "",
      "location": {
        "latitude": "",
        "longitude": ""
      }
    },
    "operationInfo": {
      "externalOperation": "activateOffer",
      "externalTransactionDate": "11/10/2018 10:07:05",
      "externalTransactionId": "123",
      "operationId": "'||LN_OPERATION_TYPE_ID1||'",
```

```
      "processingMethod": "SY",
      "compensationTransactionId": "XXXX"
    },
    "securityInfo": {
    "authorizationId": "12345"
  }
  },
  "body": {
    "activationInfo": {
      "expectedExecutionDate": "",
      "offerId": "'||LN_OFFER_ID1||'",
      "offerDetail": "",
      "externalOfferId": "",
      "remarks": "Para Pruebas de Quickwin",
      "subscription": {
        "type": "CEL",
        "value": "5930969249902"
      },
    "paymentInfo": {
    "paymentMethod": {
      "paymentAmount": "500",
      "paymentMethodId": "CARD",
      "paymentMethodType": "CARDTYPE"
    },
    "paymentDate": "11/10/2018 10:07:05",
    "currency": {
      "currencyId": "$",
      "description": "DOLAR"
    }
    },
      "campaignId": "'||LN_CAMPAIGN_ID||'",
      "productInfo":{
        "productId": "",
        "productName": "",
        "productOfferingPrice" : {
          "name":"",
          "productAmount":""
        }
      },
      "period": "",
      "periodQuantity": "",
      "quantity": "",
      "targetSubscription": {
        "type": "",
        "value": ""
      },
      "unit": "",
      "amount": "",
      "sessionData": {
        "externalSubscriberProperties": [
          {
            "id": "DUMMY",
            "value": ["TEST"]
          }
        ]
      }
    }
  }
```

```
}');

--chequear los erroes
exception when others then
    rollback;
    dbms_output.put_line(sqlerrm);
end;
```

**EJEMPLO AVANZADO BEHAVIOUR OPERATION.**

Permite que la operacion devuelva los subscriber properties definidos en **RESPONSE_PROPERTIES**

```
select nvl(max(t.BEHAVIOUR_OPERATION_ID), 0) + 1 into ln_behaviour_operation_290
from BEHAVIOUR_OPERATION t;    insert into BEHAVIOUR_OPERATION
(BEHAVIOUR_OPERATION_ID, BEHAVIOUR_ID, OPERATION_TYPE_ID,
DESCRIPTION,RESPONSE_PROPERTIES,STATUS) values
(ln_behaviour_operation_290,ln_behaviour_260,'activateOffer','QUICKWIN-IPTV
Orquestador Provisioning','[
  {
    "subscriberPropertyId": "COD_RESPUESTA"
  },
  {
    "subscriberPropertyId": "MESSAGE"
  },
  {
    "responseAlias": "EXTERNAL_TRANSACTION",
    "subscriberPropertyId": "$EXTERNALTRANSACTIONID$"
  },
  {
    "subscriberPropertyId": "QUICKWIN_TRANSACTION_ID"
  },
  {
    "subscriberPropertyId": "QUICKWIN_OPERATION_ID"
  }
]','A');
```

# REQUEST QUICKWIN

Al ejecutar este Script genera un **request**.

POST http://192.168.37.146:8101/quickWin/getActivateRequest HTTP/1.1

```
{
    "commonHeaderRequest": {
        "channelInfo": {
            "channelId": "622",
            "mediaDetailId": "BES.15.101",
            "mediaId": "CRM.6"
        },
        "consumerInfo": {
            "companyId": "CLARO",
            "consumerType": "CLARO",
            "consumerId": "SPRXSLT",
            "terminal": "HUB PLATINUM"
```

```
        },
        "geolocationInfo": {
            "country": "ECUADOR",
            "state": "GUAYAS",
            "city": "GUAYAQUIL",
            "locality": {
                "localityId": "LOC1",
                "description": "LOC DESCR"
            },
            "accuracy": "123456",
            "location": {
                "latitude": "123477",
                "longitude": "5566655"
            }
        },
        "operationInfo": {
            "externalOperation": "activateOffer",
            "externalTransactionDate": "11/10/2018 10:07:05",
            "externalTransactionId": "123",
            "operationId": "activateOffer",
            "processingMethod": "SY",
            "compensationTransactionId": "XXXX"
        },
        "securityInfo": {
            "authorizationId": "12345"
        }
    },
    "body": {
        "activationInfo": {
            "expectedExecutionDate": "",
            "offerId": "260",
            "offerDetail": "",
            "externalOfferId": "",
            "remarks": "Para Pruebas de Quickwin",
            "subscription": {
                "type": "CEL",
                "value": "5930969249902"
            },
            "paymentInfo": {
                "paymentMethod": {
                    "paymentAmount": "500",
                    "paymentMethodId": "CARD",
                    "paymentMethodType": "CARDTYPE"
                },
                "paymentDate": "11/10/2018 10:07:05",
                "currency": {
                    "currencyId": "$",
                    "description": "DOLAR"
                }
            },
            "campaignId": "248",
            "productInfo": {
                "productId": "",
                "productName": "",
                "productOfferingPrice": {
                    "name": "",
                    "productAmount": ""
                }
```

```
            },
            "period": "",
            "periodQuantity": "",
            "quantity": "",
            "targetSubscription": {
                "type": "",
                "value": ""
            },
            "unit": "",
            "amount": "",
            "sessionData": {
                "externalSubscriberProperties": [
                    {
                        "id": "DUMMY",
                        "value": [
                            "TEST"
                        ]
                    }
                ]
            }
        }
    }
}
```

## DESCRIPCION DE PARAMETROS DE CONTRATO DE QUICKWIN CON SUBSCRIBER PROPERTIES

A continuación se detallan los parámetros de **QuickWin** y los **SUBSCRIBER_PROPERTIES** que se inicializan.

Los parámetros que son obligatorios se marcan con *.

| Parámetro | Descripción | SUBSCRIBER_PROPERTIES |
|---|---|---|
| commonHeaderRequest | Objeto que contien informacion de metadata, para tracking. | |
| channelInfo | Informacion del Canal | |
| **channelInfo.channelId** | **RECOMENDABLE**: Id de Canal que llama la transaccion. | $CHANNELID$ |
| channelInfo.mediaDetailId | Código del medio o canal que consume el servicio para activación | $MEDIADETAILID$ |
| channelInfo.mediaId | Código del tipo de medio que se conecta al servicio | $MEDIAID$ |
| consumerInfo | Información del consumidor que está haciendo uso del servicio | |
| consumerInfo.companyId | Código de la compañía o entidad que consume el servicio. Ej. : CONECEL,AMX, 80, etc. | $COMPANYID$ |
| consumerInfo.consumerType | Tipo de consumidor que accede al servicio. Ej. : Retail, Dealer, DA, etc. | $CONSUMERTYPE$ |
| consumerInfo.consumerId | ID unívoco del consumir del servicio. Ej.: DealerID, Fybeca-001 | $CONSUMERID$ |
| consumerInfo.terminal | Terminal desde la cuál se realiza el consumo | $TERMINAL$ |
| geolocationInfo | Información de la localización geográfica del consumidor | |
| geolocationInfo.country | País del consumidor | $COUNTRY$ |
| geolocationInfo.state | Estado o provincia de ubicación del consumidor | $STATE$ |
| geolocationInfo.city | Ciudad o cantón de ubicación del consumidor | $CITY$ |
| geolocationInfo.locality | Información de la localidad del consumidor | |
| locality.localityID | Código de la localidad | $LOCALITYID$ |
| locality.description | Descripción de la localidad | $LOCALITYDESCRIPTION$ |
| geolocationInfo.accuracy | Porcentaje de precisión de la información de geo posicional | $ACCURACY$ |
| geolocationInfo.location | Localización de geográfica mediante geo referencia | |
| location.latitude | Coordenada de latitud | $LATITUDE$ |
| location.longitude | Coordenada de longitud | $LONGITUDE$ |
| operationInfo | Información de la operación que ejecuta el consumidor del servicio | |
| operationInfo.externalOperation | Identificador de la operación que se está ejecutando del lado del consumidor | $EXTERNALOPERATION$ |
| operationInfo.externalTransactionDate | Fecha del consumidor por motivos de rastreos debido a que puede haber disparidad del hora de sistemas del equipo del consumidor y servicio | $EXTERNALTRANSACTIONDATE$ |
| **operationInfo.externalTransactionId** | **RECOMENDABLE**: ID externo de transacción de referencia para el consumidor. **Esto con fines de trazabilidad** | $EXTERNALTRANSACTIONID$ |
| operationInfo.compensationTransactionId | ID de Transaccion de compensacion | |

| Parámetro | Descripción | SUBSCRIBER_PROPERTIES |
|---|---|---|
| operationInfo.operationId | ID de la operación que está deseando ejecutar el consumidor hacia el servicio | $OPERATIONID$ |
| ***operationInfo.async** | **OBLIGATORIO**. Sirve para que el consumidor pueda indicar si la transacción de se ejecuta manera sincrónica o asincrónica. **SY - SINCRONO- AS Asíncrona**. Esto es configurable en caso que se desee bloquear el consumo ASÍNCRONO. | $ASYNC$ |
| operationInfo.securityInfo | Información de seguridad que debe llenar el consumidor, siempre y cuando se requerido por la operación que se desee ejecutar | |
| securityInfo.authorizationId | Id de Autorización | $AUTHORIZATIONID$ |
| body | Objeto que contiene información de la oferta y datos adicionales | |
| body.activationInfo | Contiene información de la activación que desea realizar | |
| activationInfo.expectedExecutionDate | Fecha desea de la activación | $EXPECTEDEXECUTIONDATE$ |
| ***activationInfo.offerId** | **OBLIGATORIO**: ID de la oferta que se desea activar | $OFFERID$ |
| activationInfo.offerDetail | Detalle adicional de la oferta que se desee agregar | $OFFERDETAIL$ |
| activationInfo.remarks | Observación que se desea agregar como etiqueta a la activación de la oferta. | $REMARKS$ |
| activationInfo.externalOfferID | Identificador externo de la oferta, es decir su homólogo del lado del sistema o ente que realizar el consumo del servicio | $EXTERNAL_OFFER_ID$ |
| activationInfo.subscription | Contiene la información básica de la subscripción que realiza la activación de la oferta | |
| ***subscription.type** | **OBLIGATORIO**: Tipo de identificador usado para la subscripción | $SUBSCRIBERTYPE$ |
| ***subscription.value** | **OBLIGATORIO**: Identificador unívoco de la subscripción | $SUBSCRIBERID$ |
| subscription.paymentPlan | Identifica cuál es el plan de pago que usa la subscripción. Ej.: Prepaid, PostPaid, Charge | SUBSCRIBERPAYMENTPLAN$ |
| activationInfo.paymentInfo | Contiene la información de pago del a oferta que desea adquirir | |
| paymentInfo.paymentMethod | Información del método de pago usado | |
| paymentMethod.paymentAmount | Monto del pago expresado en unidades monetarias | $PAYMENTAMOUNT$ |
| paymentMethod.paymentMethodId | Id del método de pago utilizado | $PAYMENTMETHODID$ |
| paymentMethod.paymentMethodType | Identificador del tipo de método de pago. Ej. Bancario, Contra factura, Saldo, etc. | $PAYMENTMETHODTYPE$ |
| paymentInfo.paymentDate | Fecha en la que realiza el pago | $PAYMENTDATE$ |
| paymentInfo.currency | Unidad monetaria utilizada para el pago de la transacción | |

| Parámetro | Descripción | SUBSCRIBER_PROPERTIES |
|---|---|---|
| currency.currencyId | ID del unidad monetaria utilizada para el pago de la transacción | $CURRENCYID$ |
| currency.description | Descripción de la unidad monetaria de la transacción | $CURRENCYDESCRIPTION$ |
| ***activationInfo.campaignId** | **OBLIGATORIO**: ID de la campaña a la que pertenece la oferta que se desea activar | $CAMPAIGNID$ |
| activationInfo.productInfo | Información comercial del producto que se desea adquirir mediante la activación de la oferta | |
| productInfo.productId | ID del producto | $PRODUCTID$ |
| productInfo.productName | Nombre del producto | $PRODUCTNAME$ |
| productInfo.productOfferingPrice | Información del precio mercado del producto | |
| productOfferingPrice.name | Nombre de la presentación del producto. Ej. : Tarjeta de $3, voucher claro video, etc. | $PRODUCTOFFERINGPRICENAME$ |
| productOfferingPrice.productAmount | Monto en unidades monetarias de la denominación del producto. | $PRODUCTOFFERINGPRICEAMOUNT$ |
| activationInfo.period | Periodo en el cuál se realiza la activación. Ej. Days si se desea que se active en días, HOURS si se desea que se active en horas. | $PERIOD$ |
| activationInfo.periodQuantity | Números de periodos en los que se desea que se haga la acreditación de la oferta | $PERIODQUANTITY$ |
| activationInfo.targetSubscription | Subscripción destino a la que se le activa la oferta, en caso de no tener información se le activa la oferta a la subscripción que requiere dicha activación | |
| targetSubscription.type | Tipo del target de Subcribpcion | $TARGETSUBSCRIPTIONTYPE$ |
| targetSubscription.value | Valor del target de Subcribpcion | $TARGETSUBSCRIPTIONVALUE$ |
| activationInfo.quantity | Unidades que se desea comprar de la oferta | $QUANTITY$ |
| activationInfo.amount | Monto de la oferta, en caso que se desee comprar por monto monetario. Ej. Cinco dólares de tiempo aire, diez dólares de megas, etc. | $AMOUNT$ |
| activationInfo.unit | Unidad en la que se encuentra expresada la compra de la oferta | $UNIT$ |
| activationInfo.sessionData | Información adicional que desee agregar el consumidor mediante el mecanismo de registros tipo clave-valor | |
| sessionData.externalSubscriberProperties | Registro de información adicional que quiere agregar el consumidor. Recibe una lista de Subscriber Properties Definidos por el USUARIO | |
| externalSubscriberProperties[0].id | ID o clave de un dato | DUMMY |
| externalSubscriberProperties[0].value | Valor SUBSCRBIER_PROPERTIES | |

Al ejecutar el **request** anterior no hay errores actualmente, sin embargo; se requiere configurar el STAGE en la tabla **BEHVIOUR_OPER_INVOKE**:

```
LV_STAGE                VARCHAR2(200) := 'OPERATION_INVOKE';
```

Para la configuración adicional de estos **STAGES**. Ver la sesión de Bloques de Quickwin. Recuerde que todo cambio adicional después de la primer ejecución requiere actualizar la memoria.

http://192.168.37.146:8101/quickWin/clearCampaignById/248

# RESPONSE BASICO DE QUICKWIN

Salida 1.


```
{
   "activateMessageResponse":    {
      "message": "Operacion Ejecutada Exitosamente",
      "code": "0",
      "sessionData": {"externalSubscriberProperties": []}
   },
   "commonHeaderResponse": {"operationInfo":    {
      "transactionId": "9c494826-34ef-4908-b310-ee7676c50f50",
      "transactionDate": "12/05/2021 16:01:08",
      "operationId": "305"
   }},
   "userMessageResponse": null
}
```


## DESCRIPCION PARAMETROS RESPONSE

**activateMessageResponse.message**: Devuelve mensajes de Éxitos y Errores. Cuando es Éxito devuelve "**Operacion Ejecutada Exitosamente**"

**activateMessageResponse.code**: Devuelve **0** cuando es Éxito. **0** implica que la transacción de **QuickWin** se completó exitosamente  o no presentó ninguna problema de validación o conexión a servicios webs. **Código Mayor a 0** indica que la transacción terminó por alguna validación o bloqueo. Este estado es CANCELADO. (C). **Menores a 0** se refiere a errores de configuración u otro tipo de errores que se requieran reportar como negativos. Este estado es ERROR. (E). **Reportar un error menor que 0 nos guarda el request de QuickWin en la tabla de OPERATION_HISTORY con la finalidad de poder reintentarlo**. Normalmente los reintentos en **QuickWin** están por defectos cuando estos  códigos de errores son negativos. Sin embargo los reintentos pueden caerse repetidamente y el ultimo reintento caído se almacenará en **OPERATION_REPROCESSING**, siempre y cuando se haya configurado reintentos en Quickwin. Para mas detalles de reintentos en Quickwin, revisar la documentación de reintentos.

**activateMessageResponse.sessionData.externalSubscriberProperties**[]: Devuelve una lista de SUBSCRIBER_PROPERTIES configurados para la operación en el campo  CLOB **BEHAVIOUR_OPERATION.RESPONSE_PROPERTIES**.

**commonHeaderResponse.operationInfo.transactionId**: Es el ID de la transacción de **QuickWin**. Se utiliza para buscar la transacción en los logs.

```
select * from OPERATION_HISTORY t where t.operation_history_id = '9c494826-34ef-4908-b310-ee7676c50f50';
```

**commonHeaderResponse.operationInfo.transactionDate:** La fecha de la transaccion realizada por QuickWin.

**commonHeaderResponse.operationInfo.operationId**: Representa el ID de la Operación de la campaña configurada. Esta está conformada por **BEHAVIOUR** y **OPERATION_TYPE**.

**userMessageResponse:** Representa mensajes de errores humanizados en caso de haberlos. Para condiciones e invocadores cuando estos se caen o fallan por alguna razón.

# RESPONSE AVANZADO DE QUICKWIN

En este ejemplo vemos una salida mas compleja.

En **userMessageResponse** vemos que  hay mucho mas detalle del error.

**userMessageResponse.block**: Que tipo de STAGE produjo la falla. PROVISIONING, BLOCKING, VALIDATION.

**userMessageResponse.type:** Que tipo de componente. Condición, Invoke o Schedule.

**userMessageResponse.id:** El id del componente, sea Condición,Invoke o Schedule.  .

**userMessageResponse.success:** true exitoso, false falla.

**userMessageResponse.code:** El error que produjo el componente.

**userMessageResponse.message**: Menajes personalizado del Usuario.

**Nota**: Estos mensajes y códigos pueden ser configurables con el **feature** de Interpolación de **String** de Quickwin. Pueden mostrarse errores personalizados combinado con el error del mismo invocador si así se desea. Dado que en este ejemplo el componente es un INVOKE se puede tener mas datos en la tabla **INVOKE_HISTORY**. En todo caso  también puede ver detalle del flujo de INVOKES y CONDITIONS en la tabla de **OPERTATION_HISTORY**.


```
{
   "activateMessageResponse":    {
      "message": "[ PROVISIONING ] [APPLY_COMPENSATION] Error al consumir
servicio de activacion de Huawei",
      "code": "1000",
      "sessionData": {"externalSubscriberProperties": [      {
         "id": "ORDEN",
         "value": [""]
      }]}
   },
   "commonHeaderResponse": {"operationInfo":    {
      "transactionId": "ce6bb706-1d6c-413b-a000-ea2481d53986",
      "transactionDate": "13/05/2021 09:00:59",
      "operationId": "166"
   }},
   "userMessageResponse":    {
      "block": "PROVISIONING",
      "type": "Invoke",
      "id": 323,
      "success": false,
      "code": 14,
      "message": "Error al consumir servicio de activacion de Huawei"
   }
}
```

### DESCRIPCION PARAMETROS RESPONSE Y SUSBCRIBER PROPERTIES

Algunos datos de esta respuesta están disponibles a través de SUBSCRIBER_PROPERTIES.

| PROPIEDAD | SUBSCRIBER_PROPERTIES | EJEMPLO |
|---|---|---|
| commonHeaderResponse.operationInfo.**transactionId** | $INTERNALTRANSACTIONID$ | **1a7707f3-93e1-4b5a-984c-07081db7a2e7** |
| commonHeaderResponse.operationInfo.**transactionDate** | $TRANSACTIONDATE$ | **12/05/2021 18:38:29** |
| commonHeaderResponse.operationInfo.**transactionDate** | $TRANSACTIONDATE_WS$ | **2021-05-12T18:38:29-05:00** |
| commonHeaderResponse.operationInfo.**operationId** | $BEHVIOUROPERATIONID$ | **166** |

# QUICKWIN REQUEST ASINCRONO.

Para llama a **QuickWin** asíncrono se usa el mismo **Request** de **Quickwin** pero antes deben cumplirse ciertas condiciones:

En la tabla **BEHAVIOUR_OPERATION** debe configurase en la columna  MEP el valor de **AS** o **CD**, **CD** es el valor por defecto. Cuando es **AS**, obliga que el requerimiento siempre se procese asíncrono y **processingMethod** se ignora. Si es CD **processingMethod**  debe ser "**AS**". **As** sirve cuando por cualquier motivo se quiere forzar que todas las campañas consuman de forma asíncrona. CD significa CUSTOMER DEFINED y es decir el usuario decide como lo desea consumir a través de **processingMethod**.

```
{
    "commonHeaderRequest": {
        ....
        "operationInfo": {
             ..
            "processingMethod": "AS"
        }
    },
    "body": {
    ...
        }
}
```

# RESPONSE QUICKWIN ASINCRONO

## CON ERRORES.

Antes de encolar la transacción siempre existe una validación básica. Los códigos errores de configuración se muestran con códigos negativos.

```
{
   "activateMessageResponse":    {
      "message": "[ VALIDACION_INTRINSECA_ASYNC ] No se encontro ningun
campaignId, puede estar inactiva o caducada: 7",
      "code": "-4"
   },
   "commonHeaderResponse": {"operationInfo":    {
      "transactionId": "152672f7-3a13-4928-b461-e659906ae1e6",
      "transactionDate": null,
      "operationId": null
   }},
   "userMessageResponse": null
}
```


## EXITOSO

Cuando la validación es exitosa el código de Error es 0.

Nótese que **commonHeaderResponse.operationInfo.operationId** en este punto es **null**. Este debería ser determinado con el id de behaviour de campaña y tipo de operación.


```
{
    "activateMessageResponse": {
        "message": "Mensaje Encolado Exitosamente",
        "code": "0"
    },
    "commonHeaderResponse": {
        "operationInfo": {
            "transactionId": "91315ca4-d286-43b0-a354-bac2b38c1e58",
            "transactionDate": null,
            "operationId": null
        }
    },
    "userMessageResponse": null
}
```


# LOG DE CONSUMO DE CAMPAÑA

Los logs de campaña de **QuickWin** se almacenan en files y a nivel de base de datos.

Para ver la configuración de logs es importante ver la ruta de desligues. Primero se debe localizar el archivo  de despliegue. En nuestro caso en el servidor 192.168.37.146  usuario msacloud. **/dockerappsvol/dockerappdata/quickwin**

## ARCHIVO DE DESPLIEGUE

```
version: "3"
services:
  quickwin-gestor-promo:
    container_name: quickwin-gestor-promo
    restart: always
    image: arqsis/quickwin:1.0
    logging:
```

```
      driver: "json-file"
      options:
        max-size: "900k"
        max-file: "3"
    ports:
      - 8101:8086
    volumes:
      - ./bin:/quickwin/bin
      - ./config:/quickwin/config
      - /dockerlogsvol/logs/quickwin/:/quickwin/logs
    depends_on:
          - rabbitmq-service
          - redis-service
    networks:
      - quickwin_net
  rabbitmq-service:
    container_name: quickwin-rabbitmq-service
    restart: always
    logging:
      driver: none
    ports:
      - 8672:5672
      - 8673:15672
    image: rabbitmq:3-management
    networks:
      - quickwin_net
  redis-service:
    container_name: quickwin-redis-service
    restart: always
    logging:
      driver: none
    image: redis:latest
    ports:
      - 8100:6379
    command: redis-server --appendonly yes
    networks:
      - quickwin_net
networks:
  quickwin_net:
```

# RUTA DE LOGS

## SHELL SISTEMA OPERATIVO

En esta linea:"**-/dockerlogsvol/logs/quickwin/:/quickwin/logs**". Podemos ver que el log del contenedor está mapeado a un volumen  en la ruta /dockerlogsvol/logs/quickwin. Por lo tanto.

```
[08:36:18]msacloud-dgdcmsa01:/dockerlogsvol/logs/quickwin$ls -ltr
/dockerlogsvol/logs/quickwin
total 145796
-rw-r--r-- 1 root root 48546535 Jan 27 09:40 quickwin_04213874e4d2_current.log
-rw-r--r-- 1 root root 52428880 Apr  8 10:13 quickwin_a8edd0c1e15f-2021-04-08-
1.log
-rw-r--r-- 1 root root 31565215 May 14 08:35 quickwin_a8edd0c1e15f_current.log
```

```
[08:41:49]msacloud-dgdcmsa01:/dockerlogsvol/logs/quickwin$tail -f
quickwin_a8edd0c1e15f_current.log | grep 5a841563-9b61-447b-8ff1-93a10925dbdc
[20210514084032.593][INFO][com.conecel.claro.general.service.DossierService] [
5a841563-9b61-447b-8ff1-93a10925dbdc ] [ subscId: 5930969249902 ] [ campId: 248
] [ campId: 248 ][ ipCliente: 10.35.12.65 ] [ ipServer: 10.35.12.66 ] [ puerto:
8086 ]- [ pnerrorOut: 0 ] [ pverrorOut: Operacion Ejecutada Exitosamente ]
```

## DESCRIPCION DE PARAMETROS DE ARCHIVO

**[ 5a841563-9b61-447b-8ff1-93a10925dbdc ]** : id de transacción generada en **QuickWin** corresponde a **transactionId**-

**[ subscId: 5930969249902 ]** :Id de subscriptor enviado a **QuickWin** corresponde **subscriber body.activationInfo.subscription.value**.

**[ campId: 248 ]:** Id de Campaña

**[ campId: 248 ]:** id de Oferta.

**[ ipCliente: 10.35.12.65 ]** Ip que solicita.

**[ ipServer: 10.35.12.66 ]** Ip servidor que atiende.

**[ puerto: 8086 ]** : Puerto del servicio.

**[ pverrorOut: Operacion Ejecutada Exitosamente ]**: Mensaje de éxito o Error.

**Nota**: Esta pendiente agregar el parámetro de **exernalTransaction**.

A continuación veremos como ver la data en los logs.

# LOGS DE BASE DE DATOS

## QUICKWIN_MDS.OPERATION_HISTORY

| NAME | TYPE | NULLABLE | COMMENTS |
|---|---|---|---|
| OPERATION_HISTORY_ID | VARCHAR2(2000) | N | |
| SUBSCRIBER_ID | VARCHAR2(255) | Y | |
| STARTING_DATE | DATE | Y | |
| ENDING_DATE | DATE | Y | |
| ELAPSED_TIME | NUMBER | Y | |
| SHOPPINGCART_ID | VARCHAR2(255) | Y | |
| CAMPAIGN_ID | NUMBER | Y | |
| OFFER_ID | NUMBER | Y | |
| PRODUCT_ID | VARCHAR2(255) | Y | |
| RESPONSE_CODE | NUMBER | Y | |
| RESPONSE_MESSAGE | VARCHAR2(2000) | Y | |
| RESPONSE_STATUS | VARCHAR2(1) | Y | |
| RESPONSE_RESULT | CLOB | Y | |
| LOCAL_IP | VARCHAR2(255) | Y | |
| LOCAL_PORT | NUMBER | N | |
| REMOTE_IP | VARCHAR2(255) | Y | |
| OPERATION_NAME | VARCHAR2(255) | Y | |
| SESSION_DATA | CLOB | Y | |
| EXTERNAL_TRANSACCION_ID | VARCHAR2(2000) | Y | |
| REQUEST_XML | CLOB | Y | |
| REQUEST_JSON | CLOB | Y | |
| CONDITIONS_DATA | CLOB | Y | |
| TRACKING_OPER_HISTORY_ID | VARCHAR2(255) | Y | Id de Tracking de Operacion Id, para reintentos |
| RETRIES_COUNT | NUMBER | Y | Cantidad de reintentos realizados para un TRACKING_OPER_HISTORY_ID |
| CONDITIONS_GROUP_DATA | CLOB | Y | |
| COMPENSATION_OPER_HISTORY_ID | VARCHAR2(255) | Y | Id de seguimiento para compensacion |
| COMPENSATION_DATA | CLOB | Y | Datos para compensar una transaccion |

## DESCRIPCION DE COLUMNAS DE OPERATION_HISTORY

```
select * from OPERATION_HISTORY t where t.operation_history_id = '5a841563-9b61-447b-8ff1-93a10925dbdc';
```

| COLUMNS | VALUES |
| --- | --- |
| OPERATION_HISTORY_ID | 5a841563-9b61-447b-8ff1-93a10925dbdc |
| SUBSCRIBER_ID | 5,93097E+12 |
| STARTING_DATE | 14/05/2021 8:40 |
| ENDING_DATE | 14/05/2021 8:40 |
| ELAPSED_TIME | 17 |
| SHOPPINGCART_ID | |
| CAMPAIGN_ID | 248 |
| OFFER_ID | 260 |
| PRODUCT_ID | |
| RESPONSE_CODE | 0 |
| RESPONSE_MESSAGE | Operacion Ejecutada Exitosamente |
| RESPONSE_STATUS | F |
| RESPONSE_RESULT | {"invokeDataResponse":[]} |
| LOCAL_IP | 10.35.12.66 |
| LOCAL_PORT | 8086 |
| REMOTE_IP | 10.35.12.65 |
| OPERATION_NAME | activateOffer |

| COLUMNS | VALUES |
| --- | --- |
| SESSION_DATA | {"externalSubscriberProperties":[{"id":"$TRANSACTIONDATE$","value":["14/05/2021 08:40:32"]},{"id":"$TRANSACTIONDATE_WS$","value":["2021-05-14T08:40:32-05:00"]},{"id":"$CHANNELID$","value":["622"]},{"id":"$MEDIAID$","value":["CRM.6"]},{"id":"$MEDIADETAILID$","value":["BES.15.101"]},{"id":"$COMPANYID$","value":["CLARO"]},{"id":"$CONSUMERTYPE$","value":["CLARO"]},{"id":"$CONSUMERID$","value":["SPRXSLT"]},{"id":"$TERMINAL$","value":["HUB PLATINUM"]},{"id":"$COUNTRY$","value":["ECUADOR"]},{"id":"$STATE$","value":["GUAYAS"]},{"id":"$CITY$","value":["GUAYAQUIL"]},{"id":"$LOCALITYID$","value":["LOC1"]},{"id":"$LOCALITYDESCRIPTION$","value":["LOC DESCR"]},{"id":"$ACCURACY$","value":["123456"]},{"id":"$LATITUDE$","value":["123477"]},{"id":"$LONGITUDE$","value":["5566655"]},{"id":"$EXTERNALOPERATION$","value":["activateOffer"]},{"id":"$EXTERNALTRANSACTIONDATE$","value":["11/10/2018 10:07:05"]},{"id":"$EXTERNALTRANSACTIONID$","value":["123"]},{"id":"$INTERNALCOMPENSATIONTRANSACTIONID$","value":["123"]},{"id":"$OPERATIONID$","value":["activateOffer"]},{"id":"$ASYNC$","value":["SY"]},{"id":"$AUTHORIZATIONID$","value":["12345"]},{"id":"$EXPECTEDEXECUTIONDATE$","value":[""]},{"id":"$OFFERID$","value":["260"]},{"id":"$OFFERDETAIL$","value":[""]},{"id":"$REMARKS$","value":["Para Pruebas de Quickwin"]},{"id":"$EXTERNAL_OFFER_ID$","value":["null"]},{"id":"$SUBSCRIBERTYPE$","value":["CEL"]},{"id":"$SUBSCRIBERID$","value":["5930969249902"]},{"id":"$SUBSCRIBERPAYMENTPLAN$","value":[]},{"id":"$PAYMENTAMOUNT$","value":["500"]},{"id":"$PAYMENTMETHODID$","value":["CARD"]},{"id":"$PAYMENTMETHODTYPE$","value":["CARDTYPE"]},{"id":"$PAYMENTDATE$","value":["11/10/2018 10:07:05"]},{"id":"$CURRENCYID$","value":["$"]},{"id":"$CURRENCYDESCRIPTION$","value":["DOLAR"]},{"id":"$CAMPAIGNID$","value":["248"]},{"id":"$PRODUCTID$","value":[""]},{"id":"$PRODUCTNAME$","value":[""]},{"id":"$PRODUCTOFFERINGPRICENAME$","value":[""]},{"id":"$PRODUCTOFFERINGPRICEAMOUNT$","value":[""]},{"id":"$PERIOD$","value":[""]},{"id":"$PERIODQUANTITY$","value":[""]},{"id":"$TARGETSUBSCRIPTIONTYPE$","value":[""]},{"id":"$TARGETSUBSCRIPTIONVALUE$","value":[""]},{"id":"$QUANTITY$","value":[""]},{"id":"$AMOUNT$","value":[""]},{"id":"$UNIT$","value":[""]},{"id":"$INTERNALTRANSACTIONID$","value":["5a841563-9b61-447b-8ff1-93a10925dbdc"]},{"id":"DUMMY","value":["TEST"]},{"id":"$BEHAVIOURID$","value":["268"]},{"id":"$TIMEBETWEENRETRIES$","value":["1000"]},{"id":"$RETRIES$","value":["0"]},{"id":"$BEHVIOUROPERATIONID$","value":["305"]},{"id":"$MEP$","value":["CD"]},{"id":"$BEHAVIOUROPERSTAGEID$","value":["1011"]},{"id":"$IS_OPERATION_SUCCESS$","value":["YES"]}]} |
| EXTERNAL_TRANSACCION_ID | 123 |
| REQUEST_XML | |
| REQUEST_JSON | |
| CONDITIONS_DATA | {"conditionRequestResponses":[]} |
| TRACKING_OPER_HISTORY_ID | 5a841563-9b61-447b-8ff1-93a10925dbdc |

| COLUMNS | VALUES |
| --- | --- |
| RETRIES_COUNT | 0 |
| CONDITIONS_GROUP_DATA | {"conditionGroupRequestResponses":[]} |
| COMPENSATION_OPER_HISTORY_ID | XXXX |
| COMPENSATION_DATA | {} |

Mencionaremos algunas columnas que vale la pena mencionar.

**OPERATION_HISTORY_ID**: Id de operación de Quickwin.

**TRACKING_OPER_HISTORY_ID**: Representa el ID de la operación inicial. En la primera transacción **OPERATION_HISTORY_ID** y **TRACKING_OPER_HISTORY_ID** siempre serán los mismos. Pero si la operación de **QuickWin** aplica reintentos entonces **TRACKING_OPER_HISTORY_ID** se mantiene como referencia de cual fue la transacción inicial que causó reintento y **RETRIES_COUNT** mantiene el numero de reintentos ejecutados.

**EXTERNAL_TRANSACCION_ID:** Id de transacción externa. Id de tracking o trazabilidad.

**OPERATION_NAME**: Operación que se ejecuta.

**CAMPAIGN_ID**: Id de Campaña.

**OFFER_ID**: Oferta de Campaña.

**STARTING_DATE**: Fecha de inicio de transacción.

**ENDING_DATE**: Fecha fin de la transacción.

**ELAPSED_TIME**: Tiempo transcurrido o cuanto demoró la transacción en completar. Resta entre **STARTING_TIME** y **ENDING_DATE**.

**RETRIES_COUNT**: Numero de reintentos de esta transacción. Aplica cuando se ha configurado reintentos para la operación.

**RESPONSE_CODE**: 0 es Éxito (F). Menor a 0 errores (E). Mayor a 0 cancelado. (C).

**RESPONSE_STATUS**: Representa el estado de la transacción F de finalizado con Éxito,C de cancelado por algún error > 0, E de error por código menor a 0.

**SESSION_DATA**: Contiene todos los SUBSCRIBER_PROPERTIES que interactúan en la transacción en el orden de ejecución. Solo existe un único SUBSCRIBER_PROPERTIES, y en caso de sobrescribir valores, esté se mostrará con el ultimo valor.

**RESPONSE_RESULT**: Contiene todos los **Invoke** o Invocadores que fueron llamados en la transacción de **Quickwin** en el orden de ejecución.

**CONDITIONS_DATA**: Contiene el resultado de todas las condiciones en el orden que se llamaron. Por lo tanto aquí puede ver que condición se cumplió o no y sus mensajes.

**CONDITIONS_GROUP_DATA**: Contiene el resultado de todas los grupos de condiciones ejecutados.

**REQUEST_XML** y **REQUEST_JSON**: Genera un contrato para **SOAP** y **REST** cuando **RESPONSE_STATUS** tiene el valor de '**E**' . Su objetivo es que un operador humano pueda reintentar el requerimiento. **Nota**: Por cuestión de espacio, no se guarda en éxito.

**COMPENSATION_DATA**: Dato de compensación.

# SCRIPT PLSQL

## DELETE OPERACION DE CAMPAÑA CON TODOS LOS STAGES Y BLOQUES CONFIGURADOS

```
-- Created on 13/05/2021 by MGARCIAR
declare
lv_description_campaign varchar(255) := '[QWV1-MGR]%';
deleted_cant number := 0;
 -- ELIMINAR

begin



SELECT COUNT(T.CAMPAIGN_ID) INTO  deleted_cant  FROM CAMPAIGN_OFFERS T
  WHERE T.CAMPAIGN_ID IN
        (SELECT C.CAMPAIGN_ID
           FROM CAMPAIGN C
          WHERE C.LONG_DESCRIPTION LIKE lv_description_campaign);


dbms_output.put_line('cantidad de CAMPAIGN_OFFERS a eliminar: ' || deleted_cant
);

 -- POR OFFERTAS...
 DELETE FROM CAMPAIGN_OFFERS T
  WHERE T.CAMPAIGN_ID IN
        (SELECT C.CAMPAIGN_ID
           FROM CAMPAIGN C
          WHERE C.LONG_DESCRIPTION LIKE lv_description_campaign);



 -- ELIMINAR OFFER --PRODUCTS

  SELECT COUNT(T.PRODUCT_ID) INTO deleted_cant  FROM OFFER_PRODUCTS T
  WHERE T.OFFER_ID IN
        (SELECT O.OFFER_ID
           FROM OFFERS O
          WHERE O.LONG_DESCRIPTION LIKE lv_description_campaign);

 dbms_output.put_line('cantidad de OFFER_PRODUCTS a eliminar: ' || deleted_cant
);


 DELETE FROM OFFER_PRODUCTS T
  WHERE T.OFFER_ID IN
        (SELECT O.OFFER_ID
           FROM OFFERS O
          WHERE O.LONG_DESCRIPTION LIKE lv_description_campaign);

  -- DELETE PRODUCT_TYPE_PROVISIONING
```

```
  SELECT COUNT(T.PRODUCT_TYPE_PROVISIONING_ID) INTO deleted_cant  FROM
PRODUCT_TYPE_PROVISIONING T
   WHERE T.PRODUCT_TYPE IN
         (SELECT T.PRODUCT_TYPE_ID
            FROM PRODUCT_TYPES T
           WHERE T.DESCRIPTION LIKE lv_description_campaign);


  dbms_output.put_line('cantidad de PRODUCT_TYPE_PROVISIONING a eliminar: ' ||
deleted_cant );


  DELETE FROM PRODUCT_TYPE_PROVISIONING T
   WHERE T.PRODUCT_TYPE IN
         (SELECT T.PRODUCT_TYPE_ID
            FROM PRODUCT_TYPES T
           WHERE T.DESCRIPTION LIKE lv_description_campaign);
  -- DELETE PRODUCT..

  SELECT COUNT(T.PRODUCT_ID) INTO deleted_cant  FROM PRODUCTS T
  WHERE T.Product_Type_Id IN
        (SELECT T.PRODUCT_TYPE_ID
           FROM PRODUCT_TYPES T
          WHERE T.DESCRIPTION LIKE lv_description_campaign);

 dbms_output.put_line('cantidad de PRODUCTS a eliminar: ' || deleted_cant );

 DELETE FROM PRODUCTS T
  WHERE T.Product_Type_Id IN
        (SELECT T.PRODUCT_TYPE_ID
           FROM PRODUCT_TYPES T
          WHERE T.DESCRIPTION LIKE lv_description_campaign);

 -- DELETE PRODUCT TYPE
 SELECT COUNT(T.PRODUCT_TYPE_ID) INTO deleted_cant  FROM PRODUCT_TYPES T WHERE
T.DESCRIPTION LIKE lv_description_campaign;

 dbms_output.put_line('cantidad de PRODUCT_TYPES a eliminar: ' || deleted_cant
);


 DELETE FROM PRODUCT_TYPES T WHERE  T.DESCRIPTION LIKE lv_description_campaign;


 SELECT COUNT(O.OFFER_ID) INTO  deleted_cant FROM OFFERS O WHERE
O.LONG_DESCRIPTION LIKE lv_description_campaign;

 dbms_output.put_line('cantidad de OFFERS a eliminar: ' || deleted_cant );
 DELETE FROM OFFERS O WHERE O.LONG_DESCRIPTION LIKE lv_description_campaign;

 -- AQUI SE DEBEN ELIMINAR TODOS LOS BLOQUES...

SELECT COUNT(T.BEHAVIOUR_ENRICHMENT_ID) INTO deleted_cant from
behaviour_enrichment t
 where t.behaviour_oper_stage_id in
       (select t.behaviour_oper_stage_id
          from behaviour_oper_stages t
         where t.behaviour_operation_id in (SELECT T.BEHAVIOUR_OPERATION_ID
```

```
          FROM BEHAVIOUR_OPERATION T, BEHAVIOUR B
         WHERE T.BEHAVIOUR_ID = B.BEHAVIOUR_ID
           AND B.DESCRIPTION LIKE lv_description_campaign));

dbms_output.put_line('cantidad de behaviour_enrichment a eliminar: ' ||
deleted_cant );

DELETE from behaviour_enrichment t
 where t.behaviour_oper_stage_id in
       (select t.behaviour_oper_stage_id
          from behaviour_oper_stages t
         where t.behaviour_operation_id in (SELECT T.BEHAVIOUR_OPERATION_ID
          FROM BEHAVIOUR_OPERATION T, BEHAVIOUR B
         WHERE T.BEHAVIOUR_ID = B.BEHAVIOUR_ID
           AND B.DESCRIPTION LIKE lv_description_campaign));



SELECT COUNT(T.BEHAVIOUR_ENRICHE_PROPERTY_ID) INTO deleted_cant from
behaviour_enrichment_property t
 where t.behaviour_oper_stage_id in
       (select t.behaviour_oper_stage_id
          from behaviour_oper_stages t
         where t.behaviour_operation_id in (SELECT T.BEHAVIOUR_OPERATION_ID
          FROM BEHAVIOUR_OPERATION T, BEHAVIOUR B
         WHERE T.BEHAVIOUR_ID = B.BEHAVIOUR_ID
           AND B.DESCRIPTION LIKE lv_description_campaign));

dbms_output.put_line('cantidad de behaviour_enrichment_property a eliminar: ' ||
deleted_cant );

DELETE from behaviour_enrichment_property t
 where t.behaviour_oper_stage_id in
       (select t.behaviour_oper_stage_id
          from behaviour_oper_stages t
         where t.behaviour_operation_id in (SELECT T.BEHAVIOUR_OPERATION_ID
          FROM BEHAVIOUR_OPERATION T, BEHAVIOUR B
         WHERE T.BEHAVIOUR_ID = B.BEHAVIOUR_ID

           AND B.DESCRIPTION LIKE lv_description_campaign));

 SELECT COUNT(T.BEHAVIOUR_VALIDATION_ID) INTO deleted_cant from
behaviour_validations t
 where t.behaviour_oper_stage_id in
       (select t.behaviour_oper_stage_id
          from behaviour_oper_stages t
         where t.stage_id = 'DYNAMIC_VALIDATION' AND t.behaviour_operation_id in
(SELECT T.BEHAVIOUR_OPERATION_ID
          FROM BEHAVIOUR_OPERATION T, BEHAVIOUR B
         WHERE T.BEHAVIOUR_ID = B.BEHAVIOUR_ID
           AND B.DESCRIPTION LIKE lv_description_campaign));

dbms_output.put_line('cantidad de behaviour_validations a eliminar: ' ||
deleted_cant );

 DELETE from behaviour_validations t
 where t.behaviour_oper_stage_id in
       (select t.behaviour_oper_stage_id
```

```
          from behaviour_oper_stages t
         where t.stage_id = 'DYNAMIC_VALIDATION' AND t.behaviour_operation_id in
(SELECT T.BEHAVIOUR_OPERATION_ID
          FROM BEHAVIOUR_OPERATION T, BEHAVIOUR B
         WHERE T.BEHAVIOUR_ID = B.BEHAVIOUR_ID
           AND B.DESCRIPTION LIKE lv_description_campaign));

SELECT COUNT(T.BEHAVIOUR_BLOCKING_ID) INTO deleted_cant  from
BEHAVIOUR_BLOCKINGS t
 where t.behaviour_oper_stage_id in
       (select t.behaviour_oper_stage_id
          from behaviour_oper_stages t
         where t.stage_id = 'DYNAMIC_BLOCKING' AND t.behaviour_operation_id in
(SELECT T.BEHAVIOUR_OPERATION_ID
          FROM BEHAVIOUR_OPERATION T, BEHAVIOUR B
         WHERE T.BEHAVIOUR_ID = B.BEHAVIOUR_ID
           AND B.DESCRIPTION LIKE lv_description_campaign));

dbms_output.put_line('cantidad de BEHAVIOUR_BLOCKINGS a eliminar: ' ||
deleted_cant );

 DELETE from BEHAVIOUR_BLOCKINGS t
 where t.behaviour_oper_stage_id in
       (select t.behaviour_oper_stage_id
          from behaviour_oper_stages t
         where t.stage_id = 'DYNAMIC_BLOCKING' AND t.behaviour_operation_id in
(SELECT T.BEHAVIOUR_OPERATION_ID
          FROM BEHAVIOUR_OPERATION T, BEHAVIOUR B
         WHERE T.BEHAVIOUR_ID = B.BEHAVIOUR_ID
           AND B.DESCRIPTION LIKE lv_description_campaign));

SELECT COUNT(T.BEHAVIOUR_OPER_INVOKE_ID) INTO deleted_cant from
BEHAVIOUR_OPER_INVOKE t
 where t.behaviour_oper_stage_id in
       (select t.behaviour_oper_stage_id
          from behaviour_oper_stages t
         where t.stage_id = 'OPERATION_INVOKE' AND t.behaviour_operation_id in
(SELECT T.BEHAVIOUR_OPERATION_ID
          FROM BEHAVIOUR_OPERATION T, BEHAVIOUR B
         WHERE T.BEHAVIOUR_ID = B.BEHAVIOUR_ID
           AND B.DESCRIPTION LIKE lv_description_campaign));

dbms_output.put_line('cantidad de BEHAVIOUR_OPER_INVOKE a eliminar: ' ||
deleted_cant );

  DELETE from BEHAVIOUR_OPER_INVOKE t
 where t.behaviour_oper_stage_id in
       (select t.behaviour_oper_stage_id
          from behaviour_oper_stages t
         where t.stage_id = 'OPERATION_INVOKE' AND t.behaviour_operation_id in
(SELECT T.BEHAVIOUR_OPERATION_ID
          FROM BEHAVIOUR_OPERATION T, BEHAVIOUR B
         WHERE T.BEHAVIOUR_ID = B.BEHAVIOUR_ID
           AND B.DESCRIPTION LIKE lv_description_campaign));

 -- FIN DE LA ELIMINACION DEL OS BLOQUES

 -- ELIMINAR LOS OPER STAGES
```

```
 SELECT COUNT(T.BEHAVIOUR_OPER_STAGE_ID) INTO deleted_cant FROM
BEHAVIOUR_OPER_STAGES T
 WHERE T.BEHAVIOUR_OPERATION_ID IN
       (SELECT T.BEHAVIOUR_OPERATION_ID
          FROM BEHAVIOUR_OPERATION T, BEHAVIOUR B
         WHERE T.BEHAVIOUR_ID = B.BEHAVIOUR_ID
           AND B.DESCRIPTION LIKE lv_description_campaign);

 dbms_output.put_line('cantidad de BEHAVIOUR_OPER_STAGES a eliminar: ' ||
deleted_cant );

 DELETE FROM BEHAVIOUR_OPER_STAGES T
 WHERE T.BEHAVIOUR_OPERATION_ID IN
       (SELECT T.BEHAVIOUR_OPERATION_ID
          FROM BEHAVIOUR_OPERATION T, BEHAVIOUR B
         WHERE T.BEHAVIOUR_ID = B.BEHAVIOUR_ID
           AND B.DESCRIPTION LIKE lv_description_campaign);

 -- ELMINAR OPERATION....

SELECT COUNT(T.BEHAVIOUR_OPERATION_ID) INTO deleted_cant FROM
BEHAVIOUR_OPERATION T WHERE T.BEHAVIOUR_OPERATION_ID IN (SELECT
T.BEHAVIOUR_OPERATION_ID
        FROM BEHAVIOUR_OPERATION T, BEHAVIOUR B
       WHERE T.BEHAVIOUR_ID = B.BEHAVIOUR_ID
         AND B.DESCRIPTION LIKE lv_description_campaign);

 dbms_output.put_line('cantidad de BEHAVIOUR_OPERATION a eliminar: ' ||
deleted_cant );

 DELETE FROM BEHAVIOUR_OPERATION T WHERE T.BEHAVIOUR_OPERATION_ID IN (SELECT
T.BEHAVIOUR_OPERATION_ID
          FROM BEHAVIOUR_OPERATION T, BEHAVIOUR B
         WHERE T.BEHAVIOUR_ID = B.BEHAVIOUR_ID
           AND B.DESCRIPTION LIKE lv_description_campaign);


 SELECT COUNT(C.CAMPAIGN_ID) INTO  deleted_cant FROM CAMPAIGN C WHERE
C.LONG_DESCRIPTION LIKE lv_description_campaign;
 dbms_output.put_line('cantidad de CAMPAIGN a eliminar: ' || deleted_cant );


 DELETE  FROM CAMPAIGN C WHERE C.LONG_DESCRIPTION LIKE lv_description_campaign;

 SELECT COUNT(B.BEHAVIOUR_ID) INTO deleted_cant FROM BEHAVIOUR B WHERE
B.DESCRIPTION LIKE lv_description_campaign;

 dbms_output.put_line('cantidad de BEHAVIOUR a eliminar: ' || deleted_cant );


 DELETE  FROM BEHAVIOUR B WHERE B.DESCRIPTION LIKE lv_description_campaign;

exception when others then
    rollback;
    dbms_output.put_line(sqlerrm);
end
```

**OUPUT**. Asegúrese que sea la cantidad creada que se va a eliminar antes de dar COMMIT.

```
cantidad de CAMPAIGN_OFFERS a eliminar: 1
cantidad de OFFER_PRODUCTS a eliminar: 1
cantidad de PRODUCT_TYPE_PROVISIONING a eliminar: 0
cantidad de PRODUCTS a eliminar: 1
cantidad de PRODUCT_TYPES a eliminar: 1
cantidad de OFFERS a eliminar: 1
cantidad de behaviour_enrichment a eliminar: 0
cantidad de behaviour_enrichment_property a eliminar: 0
cantidad de behaviour_validations a eliminar: 0
cantidad de BEHAVIOUR_BLOCKINGS a eliminar: 0
cantidad de BEHAVIOUR_OPER_INVOKE a eliminar: 0
cantidad de BEHAVIOUR_OPER_STAGES a eliminar: 1
cantidad de BEHAVIOUR_OPERATION a eliminar: 1
cantidad de CAMPAIGN a eliminar: 1
cantidad de BEHAVIOUR a eliminar: 1
```

# Support

QukickWin  is a Claro's  project. It can grow thanks to feeback and support of other actors inside the company

# Stay in touch

- Leader Architec - TIC Manuel García
- Mail - mgarciar@claro.com.ec
- Sponsor - TIC Guillermo Proaño
- Mail - gproano@claro.com.ec
- Developers - Fernando Andrade
- Mail - fernando.andrade@gizlocorp.com