

INDICE

STAGES O BLOQUES DE COMPORTAMIENTOS

Los **Stages** o **Bloques** representan una funcionalidad definida en **QuickWin**. Los **Stages** pueden ser comparados a métodos o módulos de algún lenguaje de programación. A continuación mostramos los **STAGES** existentes a nivel de configuración o soportados.

QUICKWIN_MDS.STAGES

Nota: Los que están tachado no tienen soporte o están discontinuados.

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

STAGES CON SUS TABLAS DE CONFIGURACIONES

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

ENRICHMENT INVOKE Y ENRICHMENT PROPERTIES.

No representan ningún **STAGE** en la tabla anterior. Sin embargo se ejecutan antes que cualquier Configuración de **STAGE** (**BEHAVIOUR_OPER_STAGE_ID**) en este orden:

```
(PRE)ENRICHMENT_PROPERTIES -> ENRICHMENT_INVOKE -> (POST)ENRICHMENT_PROPERTIES ->
(STAGE)
```

Su propósito es enriquecer de datos a un BLOQUE LOGICO o Configuración de STAGE (**BEHAVIOUR_OPER_STAGE_ID**) antes de realizarse.

NOTIFICATION

Este **Stage** sirve para llamar al servicio de notificación. Las notificaciones normalmente también pueden configurarse a nivel de otros bloques y pueden ser configurables para ERROR, WARNING Y SUCCESS especialmente para errores en el sistema.

OFFER_ADDONS

Entrega de premios adicionales a una oferta después del provisioning. Se puede configurar lógica de prioridades de premio basados si un cliente aplica varias promociones.

OPERATION_INVOKE

Sirve para llamar a un **Invoke** o varios a través de funcionalidad LOOPING. Adicionales soporta **Compensation Fordward y Backard**. Soporta condiciones y grupos de condiciones. Asi como también negación de condición o grupo de condicion. No soporta Schedule hasta esta versión.

DYNAMIC_BLOCKING

Permite evaluar una condición y si es positiva bloquea.

DYNAMIC_VALIDATION

Permite evaluar una condición y si es negativa bloquea.

PROVISIONING

Teóricamente cada entrega de oferta termina con el aprovisionamiento de un producto. Es por lo cual el **Provisioning** al otorgar una oferta identifica que productos están asociadas a ellas y cual es su tipo de producto. El tipo de producto le indica al **APROVISIONADOR** de QUICKWIN que debe ejecutar una serie de invocaciones. Este proceso es repetido por cada producto con su tipo de la oferta a entregar.

El **PROVISIONING** requiere de **OFERTA, PRODUCTS, PRODUCT_TYPE** configurados para funcionar. Es el único STAGE que obliga esta configuración. Sin embargo a pesar que no se configure o no se requiera **PROVISIONING** es obligatorio la lista de configuraciones previamente mencionadas.

Si se desea configurar **Provisioning** este debe configurarse en la tabla **BEHAVIOUR_PRODUCT**. Esta tabla requiere una tabla adicional llamada **PRODUCT_TYPE_PROVISIONING**. Y como su nombre implica, esta tabla sirve para configurar un flujo de invocadores para el aprovisionamiento de un producto en específico. Cabe recalcar que este flujo de Invocadores también se lo puede lograr usando el **STAGE OPERATION_INVOKE** en ciertos escenarios.

Algunas de las características que tiene el aprovisionamiento es que puede entregar todos los productos de una oferta o algún producto en específico dependiendo de las condiciones. El aprovisionamiento o entrega de producto se la entiende como la llamada a un servicio web que ejecuta el **APROVISIONADOR DE QuickWin** de la oferta a entregar.

FUNCIONALIDADES DE STAGES

SESSION DATA

Session Data es un feature en línea que esta disponible casi en la mayoría de las configuraciones de los **STAGES** y su objetivo es proveer con Información de **Subscriber Properties**. La información que se pone aquí en caso de repetirse los mismos **Subscriber Properties**, se sobrescribe.

Su forma de inicializa es:

id: Representa el **Subscriber Properties**.

value: Valores del **Subscriber Properties**.

```
[
  {
    "id": "SERVICE_NUMBER",
    "value": [
      "985200097"
    ]
  },
  {
    "id": "OFFERING_ID",
    "value": [
      "100038136"
    ]
  },
  {
    "id": "BUSINESS_ACTION",
    "value": [
      "6"
    ]
  }
]
```

```
]
    }
]
```

Cuando no se requieren **Subscriber Properties** es necesario inicializar con una lista vacía [], caso contrario tendríamos una respuesta de validación.

En este ejemplo **activateMessageResponse.message** indica que es un ENRICHMENT_PROPERTIES antes de llamar a un STAGE_DYNAMIC_BLOCKING. En este punto no indica cual es, sin embargo es posible que pueda revisar todos los STAGES haciendo un Query a todos los configurados a través de **commonHeaderResponse.operationInfo.operationId**.

ERROR DE CONFIGURACION CON SESSION DATA

```
{
  "activateMessageResponse": {
    "message": "[ ENRICHMENT_PROPERTIES_PRE_DYNAMIC_BLOCKING ] Dato de Session de Enriquecimiento es obligatorio",
    "code": "-30",
    "sessionData": {"externalSubscriberProperties": []}
  },
  "commonHeaderResponse": {"operationInfo": {
    "transactionId": "90dbae5a-06c1-40cb-b3c9-fa25a77bc9c5",
    "transactionDate": "19/05/2021 09:53:04",
    "operationId": "205"
  }},
  "userMessageResponse": null
}
```

Aquí se debe revisar cuales son los STAGE_DYNAMIC_BLOCKING que existen.

```
select * from BEHAVIOUR_OPER_STAGES t where behaviour_operation_id = 205;
```

BEHAVIOUR_OPER_STAGE_ID	STAGE_ID	ORDER_BY	BEHAVIOUR_OPERATION_ID	VALID_FROM	VALID_UNTIL	STATUS
680	DYNAMIC_BLOCKING	10	205	29/07/2020 14:12:13	20/03/2987 23:59:59	A
681	PROVISIONING	20	205	29/07/2020 14:12:13	20/03/2987 23:59:59	A

En este caso, para BEHAVIOUR_OPER_STAGE_ID 680 debería revisarse que SESSION_DATA en BEHAVIOUR_ENRICHMENT_PROPERTIES este configurado y no NULL o VACIO.

COPIAR SUSCRIBER A OTRO SUBSCRIBER.

El valor configurado en **SESSION_DATA** inicializa variables de **Subscriber Properties** e incluso con múltiples valores, sin embargo ocurre que en ocasiones se requiere llamar a un mismo invocador seguido varias veces en una misma configuración ya sea un Sage de Operaciones u otros; este toma por defecto el índice cero del arreglo de las variables. En este caso el usuario repite la configuración del **Subscriber Properties** con el nuevo valor en el índice 0 sobrescribiendo el anterior y de esta forma lograr el cometido.

Dado que en ocasiones se requiere hacer esto de forma no estática sino dinámica para invocadores, se incluye una nueva funcionalidad que permite copiar o mover valores de un **Subscriber Properties** a otros subscriber properties basados en ciertos parámetros.

Esta funcionalidad pregunta si se quiere copiar un valor antes de llamar a un invocador, o después de llamar a un invocador, o ambas.

Da la opción a sobrescribir el valor de algún **Subscriber Properties** o agregarle valores recogidos productos de la ejecución de los invocadores esto es depuse de ejecutar el invoke.

En este escenario **COPY_FROM_SUBS_TO_SUBS** es útil sin embargo esta funcionalidad solo esta útil para el STAGE de OPERATION_INVOKE.

COPY_FROM_SUBS_TO_SUBS puede funcionar de forma estática o dinámica. Cuando es estática el usuario provee índices a copiar y cuando es dinámica depende de otro componente llamado **LOOPING_OPERATION** que es quien genera la iteración. Looping también es soportado en el STAGE de **OPERATION_INVOKE**.

Existen dos configuraciones: **afterInvoke** y **beforeInvoke** indica cuando quiere hacerse el copiado de información "despues del invoke - afterInvoke o Antes del Invoke - beforeInvoke". Una vez que las variables son copiadas pasan a formar parte del las variables Subscriber properties de **Session Data** de **QuickWin**.

Si existe información en **SESSION_DATA** en las mismas configuraciones donde se este usando, este valor se carga primero a las variables de **Session** de **QuickWin** y de allí se continúa copiando.

"**afterInvoke**": [] representa que no se utilizará nada antes del llamar al INVOKE, y lo mismo aplica para **beforeInvoke** . En caso de no requerirse, no es necesario especificar ambas con arreglos vacíos, sino mas bien esta configuración es opcional. Puede ser null o vacia.

```
{
  "beforeInvoke": [
    {
      "subscriberPropertyIdTo": "QUANTITY_NEGATIVE",
      "subscriberPropertyIdFrom": "QUANTITY_NEGATIVE_ARR",
      "copyStrategy": "ITERATOR",
      "index": "FIRST"
    }, {
      "subscriberPropertyIdTo": "FREE_UNIT_INST_ID",
      "subscriberPropertyIdFrom": "FREE_UNIT_INST_ID_ARR",
      "copyStrategy": "ITERATOR",
      "index": "FIRST"
    }, {
      "subscriberPropertyIdTo": "BILLING_ACCOUNT",
      "subscriberPropertyIdFrom": "BILLING_ACCOUNT_ARR",
      "copyStrategy": "ITERATOR",
      "index": "FIRST"
    }
  ],
  "afterInvoke": [
  ]
}
```

subscriberPropertyIdFrom: Subscriber Properties origen.

subscriberPropertyIdTo: Subscriber Properties destino.

copyStrategy: ITERATOR, SET y ADD.

Cuando **copyStrategy**: es **ITERATOR**, indica que la iteración es dinámica y esta dada por el feature de **OPERATION_LOOPING** que es quien provee el índice. Por defecto usa el **index: FIRST** e ignora cualquier otra configuración con index.

Cuando **copyStrategy**: es **SET**, indica que la iteración es fija y depende del index:

index: FIRST: Copia de un **Subscriber Property** al otro el primer valor de la lista de elementos. Puede ser 0.

index: LAST: Copia de un **Subscriber Property** al otro el último valor de la lista de elementos. Aquí el usuario debe el tamaño de la lista y restarle 1.

index: ALL: Copia de un **Subscriber Property** todos los valores de la lista de elementos a otros subscriber.

y en caso contrario **index** es cualquier valor entre 0 y el tamaño de la lista a copiar - 1.

Cuando **copyStrategy**: es **ADD** simplemente hace un MERGE de **Subscriber Properties**. Por lo tanto **index** por defecto es "**index**": "**ALL**". No hay soporte para índices de número, primero o último.

```
{
  "beforeInvoke": [
    {
      "subscriberPropertyIdTo": "REPOSITARIOS",
      "subscriberPropertyIdFrom": "REPO_A_ITERAR",
      "copyStrategy": "ITERATOR",
      "index": "FIRST"
    }
  ],
  "afterInvoke": [
    {
      "subscriberPropertyIdTo": "LISTA_SALDOS",
      "subscriberPropertyIdFrom": "SALDOS",
      "copyStrategy": "ADD",
      "index": "ALL"
    },
    {
      "subscriberPropertyIdTo": "LISTA_FECHAS",
      "subscriberPropertyIdFrom": "FECHAS",
      "copyStrategy": "ADD",
      "index": "ALL"
    }
  ]
}
```

LOOPING OPERATION

Permite configurar un lazo dinámico o fijo que sirve para iterar usando el feature de Copiar Subscribers a otras Subscribers.

NAME	TYPE	NULLABLE	DEFAULT	COMMENTS
LOOPING_OPERATION_ID	NUMBER	N		
MAX_NUM_ITERATION_TYPE	VARCHAR2(15)	Y	'FV'	FV - VALOR FIJO; SP_ALL - Todos los valores de SP - SP_DISTINCT - Todos los valores eliminando repetidos
MAX_NUM_ITERATION	VARCHAR2(20)	Y	1	Se puede poner un SP, dependiendo del tipo se puede extraer un valor fijo o dinamico, cuando es FV se espera un valor numerico....
COPY_FROM_SUBS_TO_SUBS	CLOB	Y		
STATUS	VARCHAR2(1)	Y	'A'	
LOOPING_SUBSCRIBERS_PREFIX	VARCHAR2(20)	Y		Indicar un prefijo para variables de varios grupo, como LOOPING_SP_DISTINCT_(PREFIX)

LOOPING_OPERATION_ID: Id numérico.

MAX_NUM_ITERATION_TYPE: FV - VALOR FIJO; SP - El valor esta dado en el **Subscriber Properties**, SP_ALL - Todos los valores de SP - SP_DISTINCT - Todos los valores eliminando repetidos.

MAX_NUM_ITERATION: Cuando el campo de **MAX_NUM_ITERATION_TYPE** es FV se espera un valor numérico positivo. Cuando es SP, SP_ALL, SP_DISTINCT se espera que se especifique un **Subscriber Properties**.

- Si **MAX_NUM_ITERATION_TYPE** es **SP**, entonces **MAX_NUM_ITERATION** es el valor de **Subscriber Properties**. El valor del índice 0 del **Subscriber Properties**.
- Si **MAX_NUM_ITERATION_TYPE** es **SP_ALL**, entonces **MAX_NUM_ITERATION** es la cantidad de elementos que contiene el **Subscriber Properties**.
- Si **MAX_NUM_ITERATION_TYPE** es **SP_DISTINCT**, entonces **MAX_NUM_ITERATION** es la cantidad de elementos que contiene el **Subscriber Properties** per no considera duplicados.

COPY_FROM_SUBS_TO_SUBS: Aplica cuando a nivel de detalle donde no se configure, se tome esta configuración por defecto; siempre y cuando en el detalle este campo sea NULL o VACIO. Si la configuración existe a nivel de detalle, esta se ignora.

STATUS: Estado Activo 'A', o Inactivo 'I'.

LOOPING_SUBSCRIBERS_PREFIX: Dado que se pueden crear varios lazos en una misma campaña y para que no se sobrescriba el valor de \$**LOOPING_SP_DISTINCT**_\$ que es quien guarda la cantidad de la lista se agrega un prefijo ALFANUMERICO. **Esto es Opcional**.

Nota: Este feature esta disponible en el STAGE OPERATION_INVOKE.

COMPENSACION HACIA ADELANTE Y HACIA ATRAS

Algunos STAGES que usan invocadores usan este feature. Su objetivo principal es compensar una invocación. Como funciona. **APPLY_COMPENSATION** debe SER YES. Sin embargo debe cumplirse una codician para aplicar compensación hacia atrás y hacia adelante.

Si se agota el numero de RETRIES, entonces ejecuta **INVOKE_ID_COMP_BACKWARD**.

Si no se agota el numero de RETRIES y se ejecuta exitosamente entonces ejecuta **INVOKE_ID_COMP_FORWARD**.

Esto quiere decir, que si **RETRIES** es 0, entonces no daremos oportunidad a recuperarse por lo tanto solo se ejecuta **INVOKE_ID_COMP_BACKWARD**.

RETRIES: Numero de veces que se debe reintentar.

APPLY_COMPENSATION: YES O NO

INVOKE_ID_COMP_BACKWARD: Invocador que COMPENSA. Este invocador puede ser cualquier otro servicio que compensa la afectación y por ultimo pudiera reintentar después de compensar.

INVOKE_ID_COMP_FORWARD: Este invocador es una alternativa en el caso que si se recupera, pudiéramos llamar a un servicio que continua la operación.

La COMPENSACION es una actividad asíncrona ejecuta por **QuickWin**. Cuando el invocador se cae, se reporta al usuario que la transacción va a ser COMPENSADA hacia atrás o delante.

Que STAGES usan este feature?

1. **OPER_INVOKE**
2. **PRODUCT_TYPE_PROVISIONING**
3. **BEHAVIOUR_ENRICHMENT**.

EJEMPLO DE RESPUESTA DE COMPENSACION

En este ejemplo vemos que bloque produce la compensación y cual es el invocador Fallido.

Dado que el servicio de compensación ocurre asíncrono no veremos registro en el LOG de OPERATION HISTORY hasta que el servicio de compensación termine de Realizarse. Este se agregará al final de todas las operaciones realizadas por **QuickWin**.

```
{
  "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"
  }
}
```

En este ejmplo.


```
select * from OPERATION_HISTORY t  
where operation_history_id = 'ce6bb706-1d6c-413b-a000-ea2481d53986';
```

Ejemplo.

COLUMN	VALUE
OPERATION_HISTORY_ID	ce6bb706-1d6c-413b-a000-ea2481d53986
SUBSCRIBER_ID	LHJ000
STARTING_DATE	13/05/2021 9:00
ENDING_DATE	13/05/2021 9:01
ELAPSED_TIME	7820
SHOPPINGCART_ID	
CAMPAIGN_ID	133
OFFER_ID	144
PRODUCT_ID	
RESPONSE_CODE	1000
RESPONSE_MESSAGE	Operacion Ejecutada Parcialmente y Compensada Hacia Atras Con Exito
RESPONSE_STATUS	F
RESPONSE_RESULT	
LOCAL_IP	10.35.12.66
LOCAL_PORT	8086
REMOTE_IP	10.35.12.65
OPERATION_NAME	activateOffer
SESSION_DATA	
EXTERNAL_TRANSACCION_ID	123
REQUEST_XML	
REQUEST_JSON	
CONDITIONS_DATA	
TRACKING_OPER_HISTORY_ID	ce6bb706-1d6c-413b-a000-ea2481d53986
RETRIES_COUNT	0
CONDITIONS_GROUP_DATA	
COMPENSATION_OPER_HISTORY_ID	
COMPENSATION_DATA	

Vemos que compensación aplica código 1000.

y el mensaje es: "**Operación Ejecutada Parcialmente y Compensada Hacia Atrás Con Éxito**".

RESPONSE_RESULT:

Aquí vemos que el ultimo invocador fue el de compensación.

```
{
  "invokeDataResponse": [
    .....
    {
      "invokeId": 377,
      "invokerName": "COMPENSATION_BACKWARD:PROVISIONING",
      "success": true,
      "elapsedTime": 186,
      "code": 0,
      "message": "OK",
      "externalSubscriberProperties": [],
      "request": "\n{\n  \n  \"idRequest\": \"CFAQuotaRefund\", \n\n\n  \"params\": {\n\n\n    \"serviceNumber\": \"778662522\", \n\n\n    \"businessSerialNo\" : \"SEGURO20210513090059\", \n\n\n    \"status\" : \n\n\n    \"os80\" \n  } \n\n\n\n  \"response\": \"\n\n  {\"messageCode\": \"-1\", \"messageDescription\": \"Solicitud no encontrada\", \"stackMessages\": {\"stackMessage\": []}, \"headerResponse\": {\"externalTransactionDate\": null, \"externalTransactionId\": null, \"internalTransactionId\": null, \"responseTime\": 0.0, \"token\": null}, \"response\": {\"idResult\": 0.0, \"resultDescription\": \"successful transaction\", \"customFaultMessageObject\": null}}\", \n\n\n  \"responseHeader\": \"{ \"statusInfo\": [\"OK\"], \"Content-Length\": [\"347\"], \"Date\": [\"Thu, 13 May 2021 14:01:08 GMT\"], \"Content-Type\": [\"application/json; charset=UTF-8\"], \"status\": [\"200\"] }\", \n\n\n  \"ticket\": \"3658ca39-4b94-44f0-abf0-570a92b86e6b\", \n\n\n  \"startingDate\": \"May 13, 2021 09:01:07 AM\", \n\n\n  \"endingDate\": \"May 13, 2021 09:01:08 AM\", \n\n\n  \"externalTransactionId\": \"123\", \n\n\n  \"customerInvokeId\": \"ce6bb706-1d6c-413b-a000-ea2481d53986\", \n\n\n  \"retriesCount\": 0, \n\n\n  \"endpointUrl\": \n\n\n  \"http://192.168.37.146:8082/rest/microGateway/invoke\", \n\n\n  \"httpMethod\": \"POST\", \n\n\n  \"headers\": [] \n\n\n    }
  ]
}
```

COMPENSATION_DATA.

```
{
  "operationHistoryId": "ce6bb706-1d6c-413b-a000-ea2481d53986",
  "invoke": {
    "invokeId": 323,
    "sessionData": {
      "externalSubscriberProperties": [
        {
          "id": "$TRANSACTIONDATE$",
          "value": [
            "13/05/2021 09:00:59"
          ]
        }
      ]
    }
  }
}
```

```

        .....
    ]
},
"externalTransactionId": "123",
"invokerName": "PROVISIONING:PR_Claroup",
"cacheOptions": "0",
"sync": "YES",
"customerInvokerId": "ce6bb706-1d6c-413b-a000-ea2481d53986"
},
"retries": 0,
"compensationBackward": {
    "invokeId": 377,
    "externalResources": {
        "externalResourceId": 356,
        "externalResourceGroups": {
            "externalResourceGroupId": 4,
            "description": "Microgateway"
        },
    },
    "operationNameDescription": "Cancel cupo",
    "resourceDescription": "Cancel cupo",
    "requestPatternXslt": "<?xml version='1.0'
xmlns:xsl='http://www.w3.org/1999/XSL/Transform'>
<xsl:output method='text'
indent='yes' media-type='text/json' omit-xml-declaration='yes'/>
<xsl:template match='/'>
  <xsl:for-each select='//params'>
    <xsl:element name='idRequest' namespace='CFAQuotaRefund'>
      <xsl:for-each select='_params'>
        <xsl:element name='serviceNumber'>
          <xsl:value-of select='_serviceNumber' />
        </xsl:element>
        <xsl:element name='businessSerialNo'>
          <xsl:value-of select='_businessSerialNo' />
        </xsl:element>
        <xsl:element name='status'>
          <xsl:value-of select='_status' />
        </xsl:element>
      </xsl:for-each>
    </xsl:element>
  </xsl:for-each>
</xsl:template>
</xsl:stylesheet>
'
",
    "timeout": 30000,
    "status": "A",
    "extResType": "REST",
    "requestMethod": "POST",
    "contentType": "application/json",
    "dataType": "json"
  },
  "operationNameDescription": "Anulacion de Orden De Reduccion de Cupo CFA",
  "description": "Anulacion de Orden De Reduccion de Cupo CFA",
  "status": "A",
  "retries": 0,
  "timeBetweenRetries": 1000
},
"currentStage": "PROVISIONING"
}

```

Compensación a nivel de Operation Invoke

```

{
  "activateMessageResponse": {
    "message": "[ OPERATION_INVOKE ] [APPLY_COMPENSATION] Quickwin UPDATE SERIAL IPTV no se ejecuto exitosamente",
    "code": "1000",
    "sessionData": {
      "externalSubscriberProperties": [
        {
          "id": "COD_RESPUESTA",

```

```

        "value": ["100"]
    },
    {
        "id": "MESSAGE",
        "value": ["[ DYNAMIC_BLOCKING ] La oferta se ha bloqueado. CUSTOMER
NO VALIDO"]
    },
    {
        "id": "EXTERNAL_TRANSACTION",
        "value": ["8555885555"]
    },
    {
        "id": "QUICKWIN_TRANSACTION_ID",
        "value": ["d5eb3697-cced-4711-824d-d1638dd3503f"]
    },
    {
        "id": "QUICKWIN_OPERATION_ID",
        "value": ["296"]
    }
    ]}
},
"commonHeaderResponse": {"operationInfo": {
    "transactionId": "f548aace-2256-49ae-971e-cb98e6c0888e",
    "transactionDate": "19/05/2021 11:07:13",
    "operationId": "290"
}},
"userMessageResponse": {
    "block": "OPERATION_INVOKE",
    "type": "Invoke",
    "id": 718,
    "success": false,
    "code": -221,
    "message": "Quickwin UPDATE SERIAL IPTV no se ejecuto exitosamente"
}
}

```

NEGACION DE CONDICIONES O GRUPO DE CONDICIONES

Todos los STAGES utilizan condiciones. Sin embargo en ocasiones resulta que necesitamos hacer algo en CASO CONTRARIO a la CONDICION.

Por ejemplo. Se evalúa que el cliente este esté en una lista para dar un premio. **Caso contrario notificarle que no aplica premio.** En este caso no es lo optimo configurar una condición por caso contrario.

Cuando queremos que la misma condición se evalúe como FALSE, y no como TRUE por defecto, entonces usamos **WHEN_CONDITION** y su valor FALSE.

Hemos dicho que cuando no existe una condición configurada es como que la condicion siempre es verdadera. Sin configuramos **WHEN_CONDITION** con un valor de FALSE, esta nunca se hará.

TRUE es el valor por defecto, y si esta configurada aplica para el GRUPO DE CONDICONES o la CONDICION.

Que STAGES usan este feature?

1. **PROVISIONING**
2. **OPER_INVOKE.**

NEGACION DE SCHEDULE

Algunos STAGES utilizan Schedules. Sin embargo en ocasiones resulta que necesitamos hacer algo en CASO CONTRARIO a SCHEDULE.

Por ejemplo. Se evalúa que la promoción solo este activa los dias martes de 8am a 8pm. **Caso contrario dar la promocion normal.** En este caso no es lo optimo configurar una Schedule por caso contrario.

Cuando queremos que la misma Schedule se evalúe como FALSE, y no como TRUE por defecto, entonces usamos **WHEN_SCHEDULE** y su valor FALSE.

Nota. A pesar que algunos lo tienen sus configuraciones no está soportado, como es el caso de **OPERATION_INVOKE.**

Que STAGES usan este feature?

1. **PROVISIONING**
2. **OFFER_ADDONS**

GRUPO DE NOTIFICACIONES

Algunos **STAGES** utilizan **Notification Groups**. Es utilizado para enviar una notificación basado en un tipo de error por alguna llamada a un Invocador.

Se pueden configurar las notificaciones para tres escenarios.

SUCCESS: Fue exitoso.

INVOKE_FAILURE: Fallo por algún **timeout**.

INVOKE_REQUIRED: No cumplió con lo esperado.

Normalmente una campaña puedes estar operando, pero si una invocación a un servicio falla puede enviarse una notificación o varias. Este servicio esta acoplado al servicio de NOTIFICACIONES, pero también posee configuraciones extras que pueden adaptarse a cualquier otro servicio de notificaciones vía el INVOKE o Invocador de Servicios.

Actualmente en **QuickWin** puede lograrse este escenario usando condiciones y bloques de invocaciones.

VISTAZO GENERAL DE CONFIGURACIONES.

Para continuar con las configuraciones de **QuickWin** que involucran configurar STAGES es importante previamente haber revisado:

1. **Invokes.**
2. **Schedules.**
3. **Conditions y Groups.**
4. **Creacion de Campaign**


```

WHERE C.LONG_DESCRIPTION LIKE lv_description_campaign;

-- Si se conocen los id, inicializarlos directamente
SELECT BO.BEHAVIOUR_OPERATION_ID INTO ln_behaviour_oper_id FROM
BEHAVIOUR_OPERATION BO
WHERE BO.OPERATION_TYPE_ID = ln_operation_type_id1
and bo.behaviour_id = (SELECT MAX(C.BEHAVIOUR_ID)
FROM CAMPAIGN C
WHERE C.CAMPAIGN_ID = ln_campaign_id);

-- insertar
SELECT NVL(MAX(BS.BEHAVIOUR_OPER_STAGE_ID),0) + 1 INTO
ln_behaviour_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_behaviour_oper_stage_id1, 'DYNAMIC_BLOCKING',(SELECT
NVL(MAX(BS.ORDER_BY),0) + 10 FROM BEHAVIOUR_OPER_STAGES BS WHERE
BS.BEHAVIOUR_OPERATION_ID = ln_behaviour_oper_id) ,ln_behaviour_oper_id);

SELECT NVL(MAX(BS.BEHAVIOUR_OPER_STAGE_ID),0) + 1 INTO
ln_behaviour_oper_stage_id2 FROM BEHAVIOUR_OPER_STAGES BS;
insert into behaviour_oper_stages (BEHAVIOUR_OPER_STAGE_ID, STAGE_ID, ORDER_BY,
BEHAVIOUR_OPERATION_ID)
values (ln_behaviour_oper_stage_id2, 'DYNAMIC_VALIDATION',(SELECT
NVL(MAX(BS.ORDER_BY),0) + 10 FROM BEHAVIOUR_OPER_STAGES BS WHERE
BS.BEHAVIOUR_OPERATION_ID = ln_behaviour_oper_id) ,ln_behaviour_oper_id);

SELECT NVL(MAX(BS.BEHAVIOUR_OPER_STAGE_ID),0) + 1 INTO
ln_behaviour_oper_stage_id3 FROM BEHAVIOUR_OPER_STAGES BS;
insert into behaviour_oper_stages (BEHAVIOUR_OPER_STAGE_ID, STAGE_ID, ORDER_BY,
BEHAVIOUR_OPERATION_ID)
values (ln_behaviour_oper_stage_id3, 'PROVISIONING',(SELECT
NVL(MAX(BS.ORDER_BY),0) + 10 FROM BEHAVIOUR_OPER_STAGES BS WHERE
BS.BEHAVIOUR_OPERATION_ID = ln_behaviour_oper_id) ,ln_behaviour_oper_id);

dbms_output.put_line('ln_behaviour_oper_stage_id number :=
'||ln_behaviour_oper_stage_id2||');
dbms_output.put_line('ln_behaviour_oper_stage_id number :=
'||ln_behaviour_oper_stage_id1||');
dbms_output.put_line('ln_behaviour_oper_stage_id number :=
'||ln_behaviour_oper_stage_id3||');

dbms_output.put_line('-----REFRESH-----
--');
dbms_output.put_line('http://192.168.37.146:8101/quickwin/clearCampaignById/'||ln_campaign_id);

dbms_output.put_line('-----SELECT-----
-');
dbms_output.put_line('SELECT * FROM behaviour_oper_stages T WHERE
T.BEHAVIOUR_OPERATION_ID = '||ln_behaviour_oper_id||' order by t.order_by;');
dbms_output.put_line('SELECT * FROM behaviour_oper_stages T WHERE
T.BEHAVIOUR_OPER_STAGE_ID = '||ln_behaviour_oper_stage_id1||' order by
t.order_by;');

```

```

dbms_output.put_line('SELECT * FROM behaviour_oper_stages T WHERE
T.BEHAVIOUR_OPER_STAGE_ID = '||ln_behaviour_oper_stage_id2||' order by
t.order_by;');
dbms_output.put_line('SELECT * FROM behaviour_oper_stages T WHERE
T.BEHAVIOUR_OPER_STAGE_ID = '||ln_behaviour_oper_stage_id3||' order by
t.order_by;');

dbms_output.put_line('-----UPDATE STATUS-----
-----');
dbms_output.put_line('update behaviour_oper_stages t set status = ''I'' where
t.BEHAVIOUR_OPERATION_ID = '||ln_behaviour_oper_id||';');
dbms_output.put_line('update behaviour_oper_stages t set status = ''I'' where
t.BEHAVIOUR_OPER_STAGE_ID = '||ln_behaviour_oper_stage_id1||';');
dbms_output.put_line('update behaviour_oper_stages t set status = ''I'' where
t.BEHAVIOUR_OPER_STAGE_ID = '||ln_behaviour_oper_stage_id2||';');
dbms_output.put_line('update behaviour_oper_stages t set status = ''I'' where
t.BEHAVIOUR_OPER_STAGE_ID = '||ln_behaviour_oper_stage_id3||';');

dbms_output.put_line('-----DELETE-----
-');
dbms_output.put_line('delete behaviour_oper_stages b where
b.BEHAVIOUR_OPERATION_ID = '||ln_behaviour_oper_id||';');
dbms_output.put_line('delete behaviour_oper_stages b where
b.BEHAVIOUR_OPER_STAGE_ID = '||ln_behaviour_oper_stage_id1||';');
dbms_output.put_line('delete behaviour_oper_stages b where
b.BEHAVIOUR_OPER_STAGE_ID = '||ln_behaviour_oper_stage_id2||';');
dbms_output.put_line('delete behaviour_oper_stages b where
b.BEHAVIOUR_OPER_STAGE_ID = '||ln_behaviour_oper_stage_id3||';');

dbms_output.put_line('-----REQUEST-----
-----');
dbms_output.put_line('http://192.168.37.146:8101/quickwin/getActivateRequest');

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

```

Al ejecutar nos da un resultado como este.

```

ln_behaviour_oper_stage_id number := 1014;
ln_behaviour_oper_stage_id number := 1013;
ln_behaviour_oper_stage_id number := 1015;
-----REFRESH-----
http://192.168.37.146:8101/quickwin/clearCampaignById/249
-----SELECT-----
SELECT * FROM behaviour_oper_stages T WHERE T.BEHAVIOUR_OPERATION_ID = 306 order
by t.order_by;
SELECT * FROM behaviour_oper_stages T WHERE T.BEHAVIOUR_OPER_STAGE_ID = 1013
order by t.order_by;
SELECT * FROM behaviour_oper_stages T WHERE T.BEHAVIOUR_OPER_STAGE_ID = 1014
order by t.order_by;
SELECT * FROM behaviour_oper_stages T WHERE T.BEHAVIOUR_OPER_STAGE_ID = 1015
order by t.order_by;
-----UPDATE STATUS-----
update behaviour_oper_stages t set status = 'I' where t.BEHAVIOUR_OPERATION_ID =
306;

```

```

update behaviour_oper_stages t set status = 'I' where t.BEHAVIOUR_OPER_STAGE_ID
= 1013;
update behaviour_oper_stages t set status = 'I' where t.BEHAVIOUR_OPER_STAGE_ID
= 1014;
update behaviour_oper_stages t set status = 'I' where t.BEHAVIOUR_OPER_STAGE_ID
= 1015;
-----DELETE-----
delete behaviour_oper_stages b where b.BEHAVIOUR_OPERATION_ID = 306;
delete behaviour_oper_stages b where b.BEHAVIOUR_OPER_STAGE_ID = 1013;
delete behaviour_oper_stages b where b.BEHAVIOUR_OPER_STAGE_ID = 1014;
delete behaviour_oper_stages b where b.BEHAVIOUR_OPER_STAGE_ID = 1015;
-----REQUEST-----
http://192.168.37.146:8101/quickwin/getActivateRequest

```














Al consultar los datos.

SQL

Output

Statistics

```
SELECT t.behaviour_oper_stage_id FROM behaviour_oper_stages T WHERE T.BEHAVIOUR_OPERATION_ID = 306 order by t.order_by;
```

	BEHAVIOUR_OPER_STAGE_ID	STAGE_ID	ORDER_BY	BEHAVIOUR_OPERATION_ID	VALID_FROM	VALID_UNTIL	STATUS
1	1012	OPERATION_INVOKE	10	306	17/05/2021 15:37:27	20/03/2987 23:59:59	A
2	1013	DYNAMIC_BLOCKING	20	306	18/05/2021 10:24:54	20/03/2987 23:59:59	A
3	1014	DYNAMIC_VALIDATION	30	306	18/05/2021 10:24:54	20/03/2987 23:59:59	A
4	1015	PROVISIONING	40	306	18/05/2021 10:24:54	20/03/2987 23:59:59	A

En esta tabla podemos ver el orden de cada **Stage** que corresponde a una operación. (**BEHAVIOUR_OPERTAION_ID**). El orden de ejecución esta determinado por **ORDER_BY**.















Las tabla de configuración para Cada STAGE requieren del ID generado, en este caso **BEHAVIOUR_OPER_STAGE_ID**. Este mismo ID se requiere para configurar **Enrichment Invoke** o **Enrichment Properties** en caso de necesitarse. Las tablas son opcionales, sin embargo no tiene sentido declarar un STAGE si no se va a usar.

Se puede configurar el mismo **STAGE** seguido. Sin embargo este tipo de configuración aplica cuando se va a usar ENRICHMENT antes de entrar al otro **Stage** del mismo tipo. Si no se requiere enriquecer entonces solo se necesita un solo STAGE.

No es necesario crear múltiples STAGES a menos que entre STAGES se use ENRICHMENT. Por lo tanto el siguiente ejemplo es válido pero no recomendado si no hay enriquecimiento entre los **BEHAVIOUR_OPER_STAGE_ID** desde 243 al 249.

View Output Statistics

SELECT * FROM behaviour_oper_stages T
where behaviour_operation_id = 83

	BEHAVIOUR_OPER_STAGE_ID	STAGE_ID	ORDER_BY	BEHAVIOUR_OPERATION_ID	VALID_FROM	VALID_UNTIL	STATUS
1	241	DYNAMIC_BLOCKING ...	10	83	17/10/2019 11:26:21	20/03/2987 23:59:59	A
2	243	DYNAMIC_BLOCKING ...	20	83	17/10/2019 11:26:21	20/03/2987 23:59:59	A
3	245	DYNAMIC_BLOCKING ...	30	83	17/10/2019 11:26:21	20/03/2987 23:59:59	A
4	247	DYNAMIC_BLOCKING ...	40	83	17/10/2019 11:26:21	20/03/2987 23:59:59	A
5	249	DYNAMIC_BLOCKING ...	50	83	17/10/2019 11:26:21	20/03/2987 23:59:59	A
6	251	DYNAMIC_BLOCKING ...	60	83	17/10/2019 11:26:21	20/03/2987 23:59:59	A

Para eliminar todos los STAGES se puede filtrar por la operación. Pero recuerde que debe eliminar primero las configuraciones dependientes. Si desea eliminar todas las configuraciones de una campaña puede visitar la sesión de PLSQL de Champaign donde se provee el script de eliminación.

ELIMINAR TODOS LOS STAGES ASOCIADOS A UN BEHAVIOUR OPERATION

El siguiente **SCRIPT** aplica, si es que las configuraciones dependiente no han sido creadas o han sido eliminadas.

```
-- Created on 17/05/2021 by MGARCIAR
declare
lv_description_campaign varchar(255) := '[QWV1-MGR]';
ln_campaign_id number := null;
ln_operation_type_id1 VARCHAR2(200) := 'activateOffer';
ln_behaviour_oper_id number := null;
deleted_cant          number := 0;

begin

SELECT max(C.CAMPAIGN_ID) into ln_campaign_id
      FROM CAMPAIGN C
      WHERE C.LONG_DESCRIPTION LIKE lv_description_campaign;

-- Si se conocen los id, inicializarlos directamente
SELECT BO.BEHAVIOUR_OPERATION_ID INTO ln_behaviour_oper_id FROM
BEHAVIOUR_OPERATION BO
WHERE BO.OPERATION_TYPE_ID = ln_operation_type_id1
and bo.behaviour_id = (SELECT MAX(C.BEHAVIOUR_ID)
      FROM CAMPAIGN C
      WHERE C.CAMPAIGN_ID = ln_campaign_id);

dbms_output.put_line('ln_behaviour_oper_id que se desea limpiar ' ||
      ln_behaviour_oper_id);

SELECT count(t.stage_id) into deleted_cant FROM behaviour_oper_stages T WHERE
T.BEHAVIOUR_OPERATION_ID = ln_behaviour_oper_id;
dbms_output.put_line('cantidad de behaviour_oper_stages a eliminar: ' ||
      deleted_cant);
DELETE FROM behaviour_oper_stages T WHERE T.BEHAVIOUR_OPERATION_ID =
ln_behaviour_oper_id;

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

Tendría un resultado así.

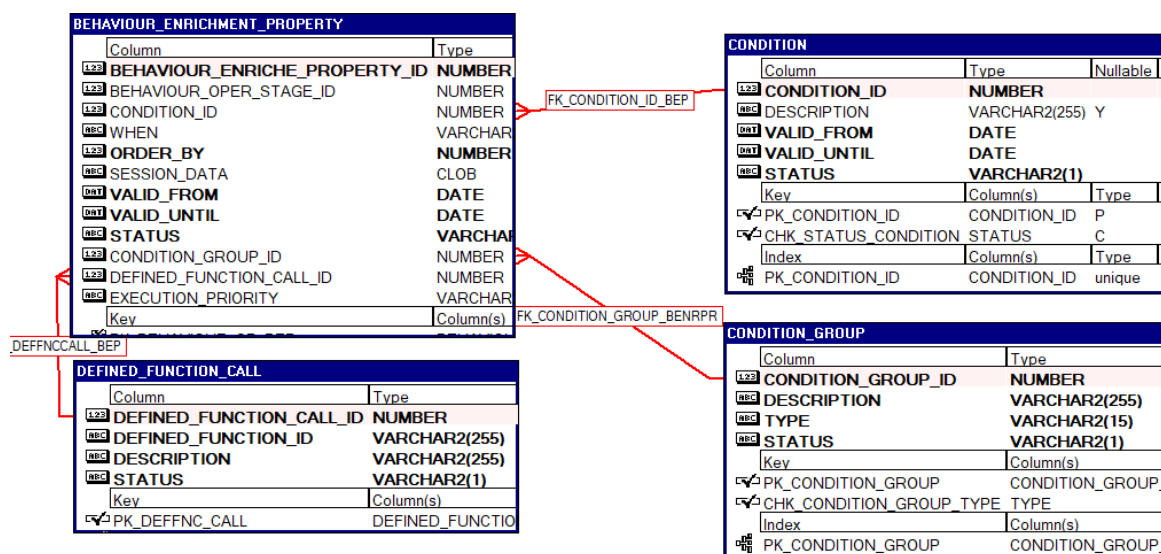
```
ln_behaviour_oper_id que se desea limpiar 306
cantidad de behaviour_oper_stages a eliminar: 4
```

ENREQUICIMIENTO DE DATOS.

Existen dos modos de enriquecer datos, los cuales veremos a continuación.

1.- ENRICHMENT Y ENRICHMENT PROPERTIES

Todo Enriquecimiento de Datos depende de un **BEHAVIOUR_OPER_STAGE_ID**. Este es el ID que precede al enriquecimiento.



Para poder configurar Enriquecimiento de datos es necesario checar los IDs de BEHAVIOUR_OPER_STAGE_ID para una BEHAVIOUR_OPERATION_ID dado. Por ejemplo.

```
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 (BEHAVIOUR_OPERATION_ID)** 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 usando enriquecimiento. Cuando se requiere hacer eso, se deben configurar en las tablas **BEHAVIOUR_ENRICHMENT**, y **BEHAVIOUR_ENRICHMENT_PROPERTY** según sea el caso el **BEHAVIOUR_OPERATION_ID** que se desea enriquecer.

2.- ENRICHMENT Y ENRICHMENT PROPERTIES PRE Y POST

Quickwin siempre llama a tres Enriquecimientos antes de ejecutar cada bloque de lógica. Esto es; llama primero a Enriquecer Propiedades, luego a Enriquecer por Invocador y finalmente llama otra vez a Enriquecer Propiedades. Esto quiere decir que enriquecer propiedades se ejecuta al inicio y fin. PRE y POST y en medio ocurre enriquecimiento por invocación a servicios web.

El **ENRICHMENT_PROPERTIES** a través de su columna "**WHEN**" siempre será el PRIMERO (PRE) y ultimo (POST) de **BEHAVIOUR_ENRICHMENT**. Al estar detrás o ser PRIMERO del **ENRICHMENT_INVOKE** le alimenta con **SUBSCRIBER_PROPERTIES** que este puede usar. El **Enrichment Propertie** siempre pregunta si es PRE o POST. Si no hay necesidad de llamar a un

BEHAVIOUR_ENRICHMENT antes o después del **BEHAVIOUR_ENRICHMENT_PROPERTY** , la configuración PRE y POST no tiene prioridad, pero el orden será **PRE** primero y luego por su columna **ORDER_BY** para las configuraciones hechas.

Ejemplo 1: Existe **Enrichment Invoke**

```
ENRICHMENT_PROPERTIES(PRE/STAGE1) -> ENRICHMENT_INVOKE(STAGE1) ->  
ENRICHMENT_PROPERTIES (POST/STAGE1) -> STAGE1
```

Ejemplo 2: No hay **Enrichment Invoke**

```
ENRICHMENT_PROPERTIES(PRE/STAGE1) -> NULL -> ENRICHMENT_PROPERTIES (POST/STAGE1)  
-> STAGE1
```

Ejemplo 3: No hay **Enrichment Properties** solo **Enrichment Invoke**

```
NULL -> ENRICHMENT_INVOKE(STAGE1) -> NULL -> STAGE1
```

En los ejemplos anterior STAGE1 representa el ID de BEHAVIOUR_OPER_STAGES.

CONFIGURACION DE BLOQUES O STAGES

BEVAHIOUR_ENRICHMENT_PROPERTIES

El **Stage** de **BEHAVIOUR_ENRICHMENT_PROPERTIES** Y **BEHAVIOUR_ENRICHMENT** siempre se ejecutarán antes de ejecutarse el **STAGE** definido por el **ID BEHAVIOUR_ENRICHE_PROPERTY_ID**. Si no se configura el **BEHAVIOUR_ENRICHE_PROPERTY_ID** previo a enriquecer, no existirá el enriquecimiento. Vale la pena recalcar, que todos los **STAGES** que ejecutan **INVOKES** o Invocadores tienen la capacidad también de enriquecer datos en el **SESSION_DATA** extraídos del resultado del **INVOKE** y **ADICIONAL** algunos que también usan **SESSION_DATA**.

El objetivo primordial de esta tabla es inicializar variables o ejecutar funciones definidas. Algunas inicializaciones o ejecución de funciones definidas pueden depender de alguna condición evaluada por éxito en esta tabla.

La inicialización de **Subscriber Properites** se configura en la columna **SESSION_DATA** y estos Subscriber Properties pasan a formar parte de todo el conjunto de **SESSION_DATA** de la transacción ejecutada por QUICKWIN llevada en memoria. Es una lista obligatoria de **Suscriber Properties**. Los **Subscriber Properties** no son validados en este punto pero la lista debe ser válida o vacía []. Internamente solo debe existir un solo **Subscriber Property** así que si se repite un **Subscriber Properties** en esta configuración, este se sobrescribirá o al menos a nivel de logs se presentará con el último valor inicializado. En algunos casos no se requiere usar esta tabla para inicializar a los **Subscriber Properties** sino más bien para llamar funciones definidas. En ambos casos se pueden aprovechar cualquiera de estas funcionalidades o combinadas.

Esta tabla evalúa siempre una condición (**CONDITION**) por defecto si es que existe y luego ejecuta la Función Definida configurada, pero en otras ocasiones podríamos necesitar lo opuesto, entonces para eso debemos cambiar este orden con **EXECUTION_PRIORITY** a **FUNCTION**. En este caso, se ejecuta primero la Función Definida y luego procedemos con la condición.

Este escenario normalmente se usa cuando la Función ejecuta alguna acción e inicializa un **Subscriber Properties** que es usado por la condición y finalmente de cumplirse inicializa alguna variable definida en el **SESSION_DATA**.

Como hemos dicho, por defecto, siempre se evalúa a la condición primero. Si no existe la condición configurada pero tampoco existe **CONDITION_GROUP_ID** como explicaremos luego, entonces es como que la condición es siempre **VERDADERA**.

CONDITION_GROUP_ID es una alternativa para evaluar mas de una condición con algún operador relacional (AND u OR). Cuando **CONDITION_GROUP_ID** se configura, entonces se ignora a la condición, sin embargo EXECUTION_PRIORITY cuando es CONDITION también aplica para el **CONDITION_GROUP_ID** por defecto si esté esta configurado.

Este **Stage** tiene dos momentos de ejecutarse. Previo al **Stage Enrichment de Invoke** y Después de un **Stage Enrichment de Invokes**. Cuando se requiere este escenario, se puede definir en la columna WHEN. (PRE Y POST). Esto es útil siempre y cuando el **Stage de enrichment de Invoke** requiera alguna lógica de validación e inicialización de variables antes de llamarse. Si no se requiere, PRE y POST no tiene sentido, sin embargo PRE se ejecutará primero en caso de configurarse.

A continuación veremos el detalle de las columnas.

NAME	TYPE	NULLABLE	DEFAULT
BEHAVIOUR_ENRICHE_PROPERTY_ID	NUMBER	N	
BEHAVIOUR_OPER_STAGE_ID	NUMBER	Y	
CONDITION_ID	NUMBER	Y	
WHEN	VARCHAR2(4)	Y	
ORDER_BY	NUMBER	N	10
SESSION_DATA	CLOB	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'
CONDITION_GROUP_ID	NUMBER	Y	
DEFINED_FUNCTION_CALL_ID	NUMBER	Y	
EXECUTION_PRIORITY	VARCHAR2(9)	Y	'CONDITION'

BEHAVIOUR_ENRICHE_PROPERTY_ID: Id numérico.

BEHAVIOUR_OPER_STAGE_ID: ID de **BEHAVIOUR_OPER_STAGE** que se desea enriquecer.

CONDITION_ID: Id condición opcional.

WHEN: PRE y POST. Antes o Después de Ejecutar **Enrichment de Invokes**.

ORDER_BY: Orden o prioridad de ejecución.

SESSION_DATA: [] representa una lista de **Subscriber Properties** con sus valores respectivos a inicializar.

VALID_FROM: Desde cuándo es aplica.

VALID_UNTIL: Hasta cuando es aplica.

STATUS: Estado A de Activo, I de Inactivo.

CONDITION_GROUP_ID: ID de Grupo de Condiciones.

DEFINED_FUNCTION_CALL_ID: ID de función definida.

EXECUTION_PRIORITY: Orden de ejecución de quien se ejecuta primero: **CONDITION** o **FUNCTION**.

SCRIPT PLSQL

CREATE BEHAVIOUR ENRICHMENT PROPERTIES

Nota: La campaña que aquí usamos es la que hemos venido creando en los ejemplos anteriores identificada en su descripción '[QWV1-MGR]%' y el tipo de Operación '**activateOffer**';

Para ver el script de creación de una campaña con un Stage visite la documentación de campaña o puede usar su propia campaña y operación.

```
declare

-- CONDITIONS

ln_condition_id number := null;
ln_operation_type_id1 VARCHAR2(200) := 'activateOffer';
ln_campaign_id number := null;
lv_description_campaign varchar(255) := '[QWV1-MGR]%;
ln_operation_type varchar2(255) := '';
-- obligatorio definir
lv_stage_id varchar2(255) := 'DYNAMIC_BLOCKING';
-- obligatorio definir
ln_behaviour_oper_id number := null;
ln_behaviour_oper_stage_id number := null;

--

begin

SELECT max(C.CAMPAIGN_ID) into ln_campaign_id
      FROM CAMPAIGN C
      WHERE C.LONG_DESCRIPTION LIKE lv_description_campaign;

-- Si se conocen los id, inicializarlos directamente
-- Si se conocen los id, inicializarlos directamente
SELECT MIN(T.BEHAVIOUR_OPER_STAGE_ID), MIN(T.BEHAVIOUR_OPERATION_ID) INTO
ln_behaviour_oper_stage_id, ln_behaviour_oper_id
      FROM BEHAVIOUR_OPER_STAGES T, BEHAVIOUR_OPERATION BO
      WHERE T.BEHAVIOUR_OPERATION_ID = BO.BEHAVIOUR_OPERATION_ID
      --AND T.STAGE_ID = lv_stage_id
      AND BO.OPERATION_TYPE_ID = ln_operation_type_id1
      and bo.behaviour_id = (SELECT MAX(C.BEHAVIOUR_ID)
      FROM CAMPAIGN C
```



```

WHERE C.CAMPAIGN_ID = ln_campaign_id);

-- insertar, sin condiciones.

insert into behaviour_enrichment_property (BEHAVIOUR_ENRICHE_PROPERTY_ID,
BEHAVIOUR_OPER_STAGE_ID, CONDITION_ID, WHEN, ORDER_BY, SESSION_DATA, STATUS,
CONDITION_GROUP_ID, DEFINED_FUNCTION_CALL_ID, EXECUTION_PRIORITY)
values ((SELECT NVL(MAX(BP.BEHAVIOUR_ENRICHE_PROPERTY_ID),0) + 1 FROM
BEHAVIOUR_ENRICHMENT_PROPERTY BP), ln_behaviour_oper_stage_id, ln_condition_id,
'PRE',(SELECT NVL(MAX(BP.ORDER_BY),10) +10 FROM BEHAVIOUR_ENRICHMENT_PROPERTY BP
WHERE BP.BEHAVIOUR_OPER_STAGE_ID = ln_behaviour_oper_stage_id) , '{"id":
"CANAL_OPERACION","value": [""]}',{id": "OFFERID_AMCO","value": [""]}'}', 'A',
'', null, 'CONDITION');
-- fin behaviour_enrichment_property

dbms_output.put_line('ln_behaviour_oper_stage_id number :=
'||ln_behaviour_oper_stage_id||';');

dbms_output.put_line('-----REFRESH-----
--');
dbms_output.put_line('http://192.168.37.146:8101/quickwin/clearCampaignById/'||
ln_campaign_id);

dbms_output.put_line('-----SELECT-----
-');
dbms_output.put_line('SELECT * FROM behaviour_oper_stages T WHERE
T.BEHAVIOUR_OPERATION_ID = '||ln_behaviour_oper_id||' order by t.order_by;');
dbms_output.put_line('SELECT * FROM behaviour_enrichment_property T WHERE
T.BEHAVIOUR_OPER_STAGE_ID = '||ln_behaviour_oper_stage_id||' order by
t.order_by;');

dbms_output.put_line('-----UPDATE STATUS-----
-----');
dbms_output.put_line('update behaviour_enrichment_property t set status = ''I''
where t.BEHAVIOUR_OPER_STAGE_ID = '||ln_behaviour_oper_stage_id||';');

dbms_output.put_line('-----DELETE-----
-');
dbms_output.put_line('delete behaviour_enrichment_property b where
b.BEHAVIOUR_OPER_STAGE_ID = '||ln_behaviour_oper_stage_id||';');

dbms_output.put_line('-----REQUEST-----
-----');
dbms_output.put_line('http://192.168.37.146:8101/quickwin/getActivateRequest');

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

El resultado dará algo así.

```

lv_behaviour_oper_stage_id number := 1012;
-----REFRESH-----
http://192.168.37.146:8101/quickwin/clearCampaignById/249
-----SELECT-----
SELECT * FROM behaviour_oper_stages T WHERE T.BEHAVIOUR_OPERATION_ID = 306 order
by t.order_by;
SELECT * FROM behaviour_enrichment_property T WHERE T.BEHAVIOUR_OPER_STAGE_ID =
1012 order by t.order_by;
-----UPDATE STATUS-----
update behaviour_enrichment_property t set status = 'I' where
t.BEHAVIOUR_OPER_STAGE_ID = 1012;
-----DELETE-----
delete behaviour_enrichment_property b where b.BEHAVIOUR_OPER_STAGE_ID = 1012;
-----REQUEST-----
http://192.168.37.146:8101/quickwin/getActivateRequest

```

Recuerde que todo cambio en configuraciones requiere refrescar cache. Por ejemplo.

<http://192.168.37.146:8101/quickWin/clearCampaignById/249>

BEHAVIOUR ENRICHMENT PROPERTIES

Dado que esta tabla es de detalles y no tiene **CONSTRAINTS** se debe tener mucho cuidado a la hora de borrar.

```

-- Created on 17/05/2021 by MGARCIAR
declare
lv_description_campaign varchar(255) := '[QWV1-MGR]%;
deleted_cant            number := 0;
begin

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.DESCRPTION 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.DESCRPTION LIKE lv_description_campaign));

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

```

El resultado dará algo así.

```

cantidad de behaviour_enrichment_property a eliminar: 1

```

BEVAHIOUR_ENRICHMENT

El **Stage** de **BEHAVIOUR_ENRICHMENT_PROPERTIES** Y **BEHAVIOUR_ENRICHMENT** siempre se ejecutarán antes de ejecutarse el **STAGE** definido por el **ID BEHAVIOUR_ENRICHE_PROPERTY_ID**. Si no se configura el **BEHAVIOUR_ENRICHE_PROPERTY_ID** previo a enriquecer, no existirá el enriquecimiento. Vale la pena recalcar, que todos los **STAGES** que ejecutan **INVOKES** o Invocadores tienen la capacidad también de enriquecer datos en el **SESSION_DATA** extraídos del resultado del INVOKE y ADICIONAL algunos que también usan **SESSION_DATA**.

El objetivo primordial de esta tabla es llamar a Invocadores. Normalmente son invocadores de consultas y estos invocadores enriquecen el **SESSION_DATA** interno de **QuickWin**.

Dado que en algunas ocasiones el Invocador a ejecutarse puede representar una alteración de datos y no necesariamente es un servicio de consulta, también se da soporte para compensar a estos servicios. **Para mayor información de compensación revisar la sesión de compensación al inicio de esta documentación.**

NAME	TYPE	NULLABLE	DEFAULT	COMMENTS
BEHAVIOUR_ENRICHMENT_ID	NUMBER	N		
BEHAVIOUR_OPER_STAGE_ID	NUMBER	Y		
CONDITION_ID	NUMBER	Y		
INVOKE_ID	NUMBER	Y		
ORDER_BY	NUMBER	Y	10	
REQUIRED	VARCHAR2(3)	Y	'YES'	
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'	
NOTIFICATION_GROUP_ID	NUMBER	Y		
WHEN_RESP_NOT_SUCCESS	VARCHAR2(10)	Y		Si la respuesta no es exitosa, marcarla como WARNING/ERROR
CONDITION_GROUP_ID	NUMBER	Y		
DEFINED_FUNCTION_CALL_ID	NUMBER	Y		
EXECUTION_PRIORITY	VARCHAR2(9)	Y	'CONDITION'	
RETRIES	NUMBER	Y	0	
APPLY_COMPENSATION	VARCHAR2(3)	Y	'NO'	Campo que indica si la linea de enriquecimiento aplica compensacion
INVOKE_ID_COMP_BACKWARD	NUMBER	Y		En caso de que los reintentos sean 0 o fallen todos los reintentos, se ejecuta este invoke el cual debe ser usado para hacer rollback
INVOKE_ID_COMP_FORWARD	NUMBER	Y		En el caso que el enriquecimiento tenga reintentos y este sea exitoso, se ejecuta esta invocacion la cual debe ser usada para continuar la transaccion.

BEHAVIOUR_ENRICHMENT_ID:

BEHAVIOUR_OPER_STAGE_ID: ID de **BEHAVIOUR_OPER_STAGE** que se desea enriquecer.

CONDITION_ID: Id condición opcional.

INVOKE_ID: Id de Invoke OBLIGATORIO.

ORDER_BY: Orden de ejecución.

REQUIRED: YES o NO. YES es obligatorio y si falla su evaluación de Invocador se devuelve el MENSAJE DE ERROR y CODIGO de ERROR configurado para ese invocador en el Response de **QuickWin**. NO, ejecuta de forma Asíncrona.

VALID_FROM: Desde cuándo es aplica.

VALID_UNTIL: Hasta cuando es aplica.

STATUS: Estado A de Activo, I de Inactivo.

NOTIFICATION_GROUP_ID: Id de la configuración de notificación.

WHEN_RESP_NOT_SUCCESS: Este feature hace que al evaluar una respuesta en el INVOKE sea reportada como ERROR. Esto permite aplicar reintentos con códigos negativos. Actualmente este feature esta DESCONTINUADO ya que las respuesta de los Invocadores pueden ahora incluir códigos negativos o positivos en sus evaluaciones.

CONDITION_GROUP_ID: ID de Grupo de Condiciones. Cuando se configura, se ignora a la condicion.

DEFINED_FUNCTION_CALL_ID: ID de función definida.

EXECUTION_PRIORITY: Orden de ejecución de quien se ejecuta primero: **CONDITION** o **FUNCTION**.

RETRIES: Numero de veces que se debe reintentar.

APPLY_COMPENSATION: YES O NO

INVOKE_ID_COMP_BACKWARD: Invocador que COMPENSA. Este invocador puede ser cualquier otro servicio que compensa la afectación y por ultimo pudiera reintentar después de compensar.

INVOKE_ID_COMP_FORWARD: Este invocador es una alternativa en el caso que si se recupera, pudiéramos llamar a un servicio que continua la operación.

SCRIPT PLSQL

PLSQL CREATE BEHAVIOUR ENRICHMENT INVOKE

En este ejemplo Ud. debe proveer los datos de la declaración. Especialmente ID de Invocadores y Condiciones. **Para este ejemplo estamos usando el ID de Invoke 757. Puesto que es prueba no importa cual usemos, sin embargo este ID debe existir al igual que la configuración de campaña con su STAGE.**

Nota: La campaña que aquí usamos es la que hemos venido creando en los ejemplos anteriores identificada en su descripción '[QWV1-MGR]%' y el tipo de Operación 'activateOffer';

Para ver el script de creación de una campaña con un Stage visite la documentación de campaña o puede usar su propia campaña y operación.

```
declare

-- CONDITIONS

-- USAR ID DE CONDICION QUE SE NECESITE
ln_condition_id number := null;
ln_operation_type_id1 VARCHAR2(200) := 'activateOffer';
ln_campaign_id number := null;
lv_description_campaign varchar(255) := '[QWV1-MGR]%;
ln_operation_type varchar2(255) := '';
-- USAR ID DE INVOCADOR QUE SE NECESITE
ln_invoke_id number := 757;
lv_stage_id varchar2(255) := 'DYNAMIC_BLOCKING';
-- obligatorio definir
ln_behaviour_oper_id number := null;
ln_behaviour_oper_stage_id number := null;
```

```

begin

SELECT max(C.CAMPAIGN_ID) into ln_campaign_id
      FROM CAMPAIGN C
      WHERE C.LONG_DESCRIPTION LIKE lv_description_campaign;

-- Si se conocen los id, inicializarlos directamente
SELECT MIN(T.BEHAVIOUR_OPER_STAGE_ID), MIN(T.BEHAVIOUR_OPERATION_ID) INTO
ln_behaviour_oper_stage_id, ln_behaviour_oper_id
      FROM BEHAVIOUR_OPER_STAGES T, BEHAVIOUR_OPERATION BO
      WHERE T.BEHAVIOUR_OPERATION_ID = BO.BEHAVIOUR_OPERATION_ID
      --AND T.STAGE_ID = lv_stage_id
      AND BO.OPERATION_TYPE_ID = ln_operation_type_id1
      and bo.behaviour_id = (SELECT MAX(C.BEHAVIOUR_ID)
                             FROM CAMPAIGN C
                             WHERE C.CAMPAIGN_ID = ln_campaign_id);

-- insertar

insert into behaviour_enrichment (BEHAVIOUR_ENRICHMENT_ID,
BEHAVIOUR_OPER_STAGE_ID, CONDITION_ID, INVOKE_ID, ORDER_BY, REQUIRED, STATUS,
NOTIFICATION_GROUP_ID, WHEN_RESP_NOT_SUCCESS, CONDITION_GROUP_ID,
DEFINED_FUNCTION_CALL_ID, EXECUTION_PRIORITY, RETRIES, APPLY_COMPENSATION,
INVOKE_ID_COMP_BACKWARD, INVOKE_ID_COMP_FORWARD)
values ((SELECT NVL(MAX(T.behaviour_enrichment_id),0) + 1 FROM
behaviour_enrichment T), ln_behaviour_oper_stage_id, ln_condition_id,
ln_invoke_id, (SELECT NVL(MAX(T.ORDER_BY),0) +10 FROM BEHAVIOUR_ENRICHMENT T
WHERE T.BEHAVIOUR_OPER_STAGE_ID = ln_behaviour_oper_stage_id), 'YES', 'A', null,
'', '', null, 'CONDITION', 0, 'NO', '', '');

dbms_output.put_line('ln_behaviour_oper_stage_id number :=
'||ln_behaviour_oper_stage_id||');

dbms_output.put_line('-----REFRESH-----
--');
dbms_output.put_line('http://192.168.37.146:8101/quickwin/clearCampaignById/'||
n_campaign_id);

dbms_output.put_line('-----SELECT-----
-');
dbms_output.put_line('SELECT * FROM behaviour_oper_stages T WHERE
T.BEHAVIOUR_OPERATION_ID = '||ln_behaviour_oper_id||' order by t.order_by;');
dbms_output.put_line('SELECT * FROM behaviour_enrichment T WHERE
T.BEHAVIOUR_OPER_STAGE_ID = '||ln_behaviour_oper_stage_id||' order by
t.order_by;');

dbms_output.put_line('-----UPDATE STATUS-----
-----');
dbms_output.put_line('update behaviour_enrichment t set status = 'I' where
t.BEHAVIOUR_OPER_STAGE_ID = '||ln_behaviour_oper_stage_id||');

dbms_output.put_line('-----DELETE-----
-');
dbms_output.put_line('delete behaviour_enrichment b where
b.BEHAVIOUR_OPER_STAGE_ID = '||ln_behaviour_oper_stage_id||');

```

```

dbms_output.put_line('-----REQUEST-----');
dbms_output.put_line('http://192.168.37.146:8101/quickwin/getActivateRequest');

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

```

Al ejecutar este bloque Ud. tendrá el siguiente resultado.

```

ln_behaviour_oper_stage_id number := 1012;
-----REFRESH-----
http://192.168.37.146:8101/quickwin/clearCampaignById/249
-----SELECT-----
SELECT * FROM behaviour_oper_stages T WHERE T.BEHAVIOUR_OPERATION_ID = 306 order
by t.order_by;
SELECT * FROM behaviour_enrichment T WHERE T.BEHAVIOUR_OPER_STAGE_ID = 1012
order by t.order_by;
-----UPDATE STATUS-----
update behaviour_enrichment t set status = 'I' where t.BEHAVIOUR_OPER_STAGE_ID =
1012;
-----DELETE-----
delete behaviour_enrichment b where b.BEHAVIOUR_OPER_STAGE_ID = 1012;
-----REQUEST-----
http://192.168.37.146:8101/quickwin/getActivateRequest

```

PLSQL DELETE BEHAVIOUR ENRICHMENT INVOKE

Dado que esta tabla es de detalles y no tiene **CONSTRAINTS** se debe tener mucho cuidado a la hora de borrar.

```

-- Created on 17/05/2021 by MGARCIAR
declare
    lv_description_campaign varchar(255) := '[QWV1-MGR]';
    deleted_cant            number := 0;
begin

    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.DESCRPTION 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.DESCRPTION LIKE lv_description_campaign));

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

Debe dar un resultado igual a este.

cantidad de behaviour_enrichment a eliminar: 1

EJEMPLO DE BEHAVIOUR_ENRICHMENT PROPERTY FALLIDO

En este ejemplo veremos básicamente información del Invocador devuelta en el Response de **QuickWin**. Es Código de Error y Mensaje de Error del Invocador cuando este falla por algún error general definido por **QuickWin** internamente. En muchas ocasiones este código puede ser positivo o negativo dependiendo de como se configure la evaluación de la respuesta de Invocador por FALLAS. Se recomienda que sea positivo cuando la transacción debe terminar o cancelarse, y negativo si se desea un reintento de toda la transacción. Recuerde algunos reintentos necesitan primero compensar, asegúrese que cuando aplique reintentos la compensación se haya hecho antes.

```

{
  "activateMessageResponse": {
    "message": "[ ENRICHMENT_PRE_DYNAMIC_BLOCKING ] Invoke ID: 711 Error
Invoke 711. ExternalResourceId. 778, ExternalResourceComponentId. 829, Url
http://192.168.37.146:8180/EAServices/EIS/request, Timeout. 3000, Error al
llamar POST Rest: http://192.168.37.146:8180/EAServices/EIS/request
java.net.SocketTimeoutException: Read timed out",
    "code": "-31",
    "sessionData": {"externalSubscriberProperties": [ {
      "id": "MESSAGE",
      "value": []
    }]}
  },
  "commonHeaderResponse": {"operationInfo": {
    "transactionId": "4fdf1883-083e-439b-9535-d1482efc51d6",
    "transactionDate": "13/05/2021 15:20:01",
    "operationId": "271"
  }},
  "userMessageResponse": {
    "block": "DYNAMIC_BLOCKING",
    "type": "Invoke",
    "id": 711,
    "success": false,
    "code": -31,
    "message": "Invoke ID: 711 Error Invoke 711. ExternalResourceId. 778,
ExternalResourceComponentId. 829, Url
http://192.168.37.146:8180/EAServices/EIS/request, Timeout. 3000, Error al
llamar POST Rest: http://192.168.37.146:8180/EAServices/EIS/request
java.net.SocketTimeoutException: Read timed out"
  }
}
```



```
}  
}
```

ERRORES INVOCADORES CON STAGES QUE LO USAN.

Cuando se presenta algún tipo de error con el Invocador. Tenemos en la salida del response de **QuickWin** un mensaje general, **activateMessageResponse.message** . La parte que esta entre corchete indica el [ACCION_EJECUTANDOSE y STAGE], para este ejemplo [ENRICHMENT_PRE_DYNAMIC_BLOCKING] indica que lo que esta ocurriendo es un **ENRICHMENT_PRE_DYNAMIC_BLOCKING** . Se está realizando un Enriquecimiento de datos antes de ejecutar un **DYNAMIC_BLOCKING**. (Antes de ejecutar un bloqueo dinámico). Se da detalle de la configuración de ese INVOKE y su tiempo de timeout configurado. Realmente el error es lo ultimo después de la url. **java.net.SocketTimeoutException: Read timed out**. Esto al ser un error negativo de Sistema habilita que se guarde el request de QuickWin. De esta forma si el operador corrige el error del invocador puede reintentarlo usando este request.

Y mucho mas detallado se encuentra el mensaje **userMessageResponse**. Y para este caso "block": "**DYNAMIC_BLOCKING**" indica que es el STAGE previo a EJECUTAR dado que los enriquecimiento ocurren antes de cada STAGE y no dentro de los STAGES. Dentro de esta sección hay mas detalle del componente indicando el tipo y su id etc. Esto es análogo también cuando ocurren problemas con **Conditions** o Condiciones.

```
{  
  "activateMessageResponse": {  
    "message": "[ ENRICHMENT_PRE_DYNAMIC_BLOCKING ] Invoke ID: 711 Error  
Invoke 711. ExternalResourceId. 778, ExternalResourceComponenteId. 829, Url  
http://192.168.37.146:8180/EAServices/EIS/request, Timeout. 3000, Error al  
llamar POST Rest: http://192.168.37.146:8180/EAServices/EIS/request  
java.net.SocketTimeoutException: Read timed out",  
    "code": "-31",  
    "sessionData": {"externalSubscriberProperties": [ {  
      "id": "MESSAGE",  
      "value": []  
    }  
  ]}  
  },  
  "commonHeaderResponse": {"operationInfo": {  
    "transactionId": "4fdf1883-083e-439b-9535-d1482efc51d6",  
    "transactionDate": "13/05/2021 15:20:01",  
    "operationId": "271"  
  }  
},  
  "userMessageResponse": {  
    "block": "DYNAMIC_BLOCKING",  
    "type": "Invoke",  
    "id": 711,  
    "success": false,  
    "code": -31,  
    "message": "Invoke ID: 711 Error Invoke 711. ExternalResourceId. 778,  
ExternalResourceComponenteId. 829, Url  
http://192.168.37.146:8180/EAServices/EIS/request, Timeout. 3000, Error al  
llamar POST Rest: http://192.168.37.146:8180/EAServices/EIS/request  
java.net.SocketTimeoutException: Read timed out"  
  }  
}
```

Para obtener el request lo puede recuperar con el id de transacción.

commonHeaderResponse.transactionId: **4fdf1883-083e-439b-9535-d1482efc51d6**.

```
select t.request_xml , t.request_json request_json from OPERATION_HISTORY t
where t.operation_history_id = '4fdf1883-083e-439b-9535-d1482efc51d6';
```

Nota: Para mayor información o detalle de los parámetros de request y response, revise la documentación de Campaign.

JSON.

Basta copiar y pegar, para ejecutarse de nuevo.

POST <http://10.31.32.15:8101/quickWin/getActivateRequest>

```
{
  "body": {
    "activationInfo": {
      "periodQuantity": "",
      "targetSubscription": {
        "value": "",
        "type": ""
      },
      "period": "",
      "campaignId": "221",
      "amount": "",
      "unit": "",
      "offerDetail": "",
      "productInfo": {
        "productId": "",
        "productName": "",
        "productOfferingPrice": {
          "name": "",
          "productAmount": ""
        },
        "paymentInfo": {
          "paymentMethod": {
            "paymentMethodId": "",
            "paymentMethodType": ""
          },
          "subscription": {
            "value": "11351781",
            "type": "SID",
            "expectedExecutionDate": "",
            "quantity": "",
            "offerId": "231",
            "sessionData": {
              "externalSubscriberProperties": [
                {
                  "id": "EMAIL",
                  "value": ""
                },
                {
                  "id": "CUSTOMER_ID",
                  "value": "593986327327"
                },
                {
                  "id": "SERIAL_NUMBER",
                  "value": ["67897"]
                },
                {
                  "id": "CHANNEL",
                  "value": ["SGA"]
                },
                {
                  "id": "EVENT_NAME",
                  "value": ["PROVISIONING_AMCO"]
                },
                {
                  "id": "CANONICAL_OFFER_ID",
                  "value": ["12345"]
                }
              ]
            },
            "commonHeaderRequest": {
              "geoLocationInfo": {
                "location": {
                  "longitude": "",
                  "latitude": ""
                },
                "accuracy": ""
              },
              "consumerInfo": {
                "consumerType": "CLARO",
                "terminal": "HUB PLATINUM",
                "companyId": "CLARO",
                "consumerId": "SPRXSLT"
              },
              "channelInfo": {
                "mediaId": "SGA",
                "mediaDetailId": "BES.15.101",
                "channelId": "622"
              },
              "operationInfo": {
                "externalTransactionId": "123",
                "externalOperation": "activateOffer",
                "externalTransactionDate": "11/10/2018 10:07:05",
                "operationId": "activateOffer",
                "processingMethod": "SY"
              }
            }
          }
        }
      }
    }
  }
}
```

XML.

Finalmente en caso de requerir usarse en formato SOAP de cumplir con el WSDL.

<http://192.168.37.146:8101/quickwin/ws/quickwin.wsdl>

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:quic="http://message.core.app.claro.conecel.com/quickwin">
  <soapenv:Header/>
  <soapenv:Body>
    <quic:activateRequest>
      <!--Optional:-->
      <quic:commonHeaderRequest>
        <!--Optional:-->
        <quic:channelInfo>
          <!--Optional:-->
          <quic:mediaId?></quic:mediaId>
          <!--Optional:-->
          <quic:mediaDetailId?></quic:mediaDetailId>
          <!--Optional:-->
        </quic:channelInfo>
      </quic:commonHeaderRequest>
    </quic:activateRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

```

        <quic:channelId>?</quic:channelId>
    </quic:channelInfo>
    <!--Optional:-->
    <quic:consumerInfo>
        <!--Optional:-->
        <quic:companyId>?</quic:companyId>
        <!--Optional:-->
        <quic:consumerType>?</quic:consumerType>
        <!--Optional:-->
        <quic:consumerId>?</quic:consumerId>
        <!--Optional:-->
        <quic:terminal>?</quic:terminal>
    </quic:consumerInfo>
    <!--Optional:-->
    <quic:geoLocationInfo>
        <!--Optional:-->
        <quic:country>?</quic:country>
        <!--Optional:-->
        <quic:state>?</quic:state>
        <!--Optional:-->
        <quic:city>?</quic:city>
        <!--Optional:-->
        <quic:locality>
            <!--Optional:-->
            <quic:localityId>?</quic:localityId>
            <!--Optional:-->
            <quic:description>?</quic:description>
        </quic:locality>
        <!--Optional:-->
        <quic:accuracy>?</quic:accuracy>
        <!--Optional:-->
        <quic:location>
            <!--Optional:-->
            <quic:latitude>?</quic:latitude>
            <!--Optional:-->
            <quic:longitude>?</quic:longitude>
        </quic:location>
    </quic:geoLocationInfo>
    <!--Optional:-->
    <quic:operationInfo>
        <!--Optional:-->
        <quic:externalOperation>?</quic:externalOperation>
        <!--Optional:-->
        <quic:externalTransactionDate>?</quic:externalTransactionDate>
        <!--Optional:-->
        <quic:externalTransactionId>?</quic:externalTransactionId>
        <!--Optional:-->
        <quic:operationId>?</quic:operationId>
        <!--Optional:-->
        <quic:processingMethod>?</quic:processingMethod>
        <!--Optional:-->
        <quic:compensationTransactionId>?
    </quic:compensationTransactionId>
    </quic:operationInfo>
    <!--Optional:-->
    <quic:securityInfo>
        <!--Optional:-->
        <quic:authorizationId>?</quic:authorizationId>
    </quic:securityInfo>
</quic:transaction>

```

```
</quic:securityInfo>
</quic:commonHeaderRequest>
<!--Optional:-->
<quic:body>
  <!--Optional:-->
  <quic:activationInfo>
    <!--Optional:-->
    <quic:expectedExecutionDate?></quic:expectedExecutionDate>
    <!--Optional:-->
    <quic:offerId?></quic:offerId>
    <!--Optional:-->
    <quic:externalOfferId?></quic:externalOfferId>
    <!--Optional:-->
    <quic:offerDetail?></quic:offerDetail>
    <!--Optional:-->
    <quic:remarks?></quic:remarks>
    <!--Optional:-->
    <quic:subscription>
      <!--Optional:-->
      <quic:type?></quic:type>
      <!--Optional:-->
      <quic:value?></quic:value>
      <!--Optional:-->
      <quic:paymentPlan?></quic:paymentPlan>
    </quic:subscription>
    <!--Optional:-->
    <quic:paymentInfo>
      <!--Optional:-->
      <quic:paymentMethod>
        <!--Optional:-->
        <quic:paymentAmount?></quic:paymentAmount>
        <!--Optional:-->
        <quic:paymentMethodId?></quic:paymentMethodId>
        <!--Optional:-->
        <quic:paymentMethodType?></quic:paymentMethodType>
      </quic:paymentMethod>
      <!--Optional:-->
      <quic:paymentDate?></quic:paymentDate>
      <!--Optional:-->
      <quic:currency>
        <!--Optional:-->
        <quic:currencyId?></quic:currencyId>
        <!--Optional:-->
        <quic:description?></quic:description>
      </quic:currency>
    </quic:paymentInfo>
    <!--Optional:-->
    <quic:campaignId?></quic:campaignId>
    <!--Optional:-->
    <quic:productInfo>
      <!--Optional:-->
      <quic:productId?></quic:productId>
      <!--Optional:-->
      <quic:productName?></quic:productName>
      <!--Optional:-->
      <quic:productOfferingPrice>
        <!--Optional:-->
        <quic:name?></quic:name>
```

```

        <!--Optional:-->
        <quic:productAmount?></quic:productAmount>
    </quic:productOfferingPrice>
</quic:productInfo>
<!--Optional:-->
<quic:period?></quic:period>
<!--Optional:-->
<quic:periodQuantity?></quic:periodQuantity>
<!--Optional:-->
<quic:targetSubscription>
    <!--Optional:-->
    <quic:type?></quic:type>
    <!--Optional:-->
    <quic:value?></quic:value>
</quic:targetSubscription>
<!--Optional:-->
<quic:quantity?></quic:quantity>
<!--Optional:-->
<quic:amount?></quic:amount>
<!--Optional:-->
<quic:unit?></quic:unit>
<!--Optional:-->
<quic:sessionData>
    <!--Zero or more repetitions:-->
    <quic:externalSubscriberProperties>
        <!--Optional:-->
        <quic:id?></quic:id>
        <!--Zero or more repetitions:-->
        <quic:value?></quic:value>
    </quic:externalSubscriberProperties>
</quic:sessionData>
</quic:activationInfo>
</quic:body>
</quic:activateRequest>
</soapenv:Body>
</soapenv:Envelope>

```

<http://192.168.37.146:8101/quickwin/ws.>

Nota: Este XML recuperado de base no contiene la parte SOAP, por lo tanto debe ser previamente ajustado al ejemplo anterior.

para este caso debemos agregar..

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:quic="http://message.core.app.claro.conecel.com/quickwin">
    <soapenv:Header/>
    <soapenv:Body>
        ..... XML DE BASE DE DATOS .....
    </soapenv:Body>
</soapenv:Envelope>

```

XML DE BASE DE DATOS

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
```

```

<quic:activateRequest
xmlns:quic="http://message.core.app.claro.conecel.com/quickwin">
  <quic:commonHeaderRequest>
    <quic:channelInfo>
      <quic:mediaId>SGA</quic:mediaId>
      <quic:mediaDetailId>BES.15.101</quic:mediaDetailId>
      <quic:channelId>622</quic:channelId>
    </quic:channelInfo>
    <quic:consumerInfo>
      <quic:companyId>CLARO</quic:companyId>
      <quic:consumerType>CLARO</quic:consumerType>
      <quic:consumerId>SPRXSLT</quic:consumerId>
      <quic:terminal>HUB PLATINUM</quic:terminal>
    </quic:consumerInfo>
    <quic:geolocationInfo>
      <quic:accuracy></quic:accuracy>
      <quic:location>
        <quic:latitude></quic:latitude>
        <quic:longitude></quic:longitude>
      </quic:location>
    </quic:geolocationInfo>
    <quic:operationInfo>
      <quic:externalOperation>activateOffer</quic:externalOperation>
      <quic:externalTransactionDate>11/10/2018
10:07:05</quic:externalTransactionDate>
      <quic:externalTransactionId>123</quic:externalTransactionId>
      <quic:operationId>activateOffer</quic:operationId>
      <quic:processingMethod>SY</quic:processingMethod>
    </quic:operationInfo>
  </quic:commonHeaderRequest>
  <quic:body>
    <quic:activationInfo>
      <quic:expectedExecutionDate></quic:expectedExecutionDate>
      <quic:offerId>231</quic:offerId>
      <quic:offerDetail></quic:offerDetail>
      <quic:subscription>
        <quic:type>SID</quic:type>
        <quic:value>11351781</quic:value>
      </quic:subscription>
      <quic:productInfo>
        <quic:productId></quic:productId>
        <quic:productName></quic:productName>
        <quic:productOfferingPrice>
          <quic:name></quic:name>
          <quic:productAmount></quic:productAmount>
        </quic:productOfferingPrice>
      </quic:productInfo>
      <quic:paymentInfo>
        <quic:paymentMethod>
          <quic:paymentMethodId></quic:paymentMethodId>
          <quic:paymentMethodType></quic:paymentMethodType>
        </quic:paymentMethod>
      </quic:paymentInfo>
      <quic:campaignId>221</quic:campaignId>
      <quic:period></quic:period>
      <quic:periodQuantity></quic:periodQuantity>
      <quic:targetSubscription>
        <quic:type></quic:type>

```

```

        <quic:value></quic:value>
    </quic:targetSubscription>
    <quic:quantity></quic:quantity>
    <quic:amount></quic:amount>
    <quic:unit></quic:unit>
    <quic:sessionData>
        <quic:externalSubscriberProperties>
            <quic:id>EMAIL</quic:id>
            <quic:value></quic:value>
        </quic:externalSubscriberProperties>
        <quic:externalSubscriberProperties>
            <quic:id>CUSTOMER_ID</quic:id>
            <quic:value>593986327327</quic:value>
        </quic:externalSubscriberProperties>
        <quic:externalSubscriberProperties>
            <quic:id>SERIAL_NUMBER</quic:id>
            <quic:value>67897</quic:value>
        </quic:externalSubscriberProperties>
        <quic:externalSubscriberProperties>
            <quic:id>CHANNEL</quic:id>
            <quic:value>SGA</quic:value>
        </quic:externalSubscriberProperties>
        <quic:externalSubscriberProperties>
            <quic:id>EVENT_NAME</quic:id>
            <quic:value>PROVISIONING_AMCO</quic:value>
        </quic:externalSubscriberProperties>
        <quic:externalSubscriberProperties>
            <quic:id>CANONICAL_OFFER_ID</quic:id>
            <quic:value>12345</quic:value>
        </quic:externalSubscriberProperties>
    </quic:sessionData>
</quic:activationInfo>
</quic:body>
</quic:activateRequest>

```

BEVAHIOUR_BLOCKING

El objetivo de usar esta configuración es que en caso de que una condición se evalué por verdadera el sistema aborta la transacción. Esta tabla es como el inverso de BEHAVIOUR_VALIDATION.

Adicional esta tabla involucra **Schedule**. Cuando Schedule esta configurado se debe cumplir la Condición y el Schedule. La **Condition** cuando es **NULL** es como una condición siempre verdadera y lo mismo aplica con Schedule, si no existe significa que se ejecuta a todas horas.

Cuando se configura **Condition Groups (CONDITION_GROUP_ID)**, se da prioridad al grupo de condiciones y se ignora a la condición configurada. En estos casos es preferible dejar a la condición como **NULL**.

Si no existen condiciones ni grupo de condiciones configurados, esto produce siempre un bloqueo. Es algo que no deberíamos hacer, sin embargo.

NAME	TYPE	NULLABLE	DEFAULT
BEHAVIOUR_BLOCKING_ID	NUMBER	N	
BEHAVIOUR_OPER_STAGE_ID	NUMBER	Y	
DESCRIPTION	VARCHAR2(255)	Y	
CONDITION_ID	NUMBER	Y	
ORDER_BY	NUMBER	N	10
SCHEDULE_ID	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'
NOTIFICATION_GROUP_ID	NUMBER	Y	
CONDITION_GROUP_ID	NUMBER	Y	

BEHAVIOUR_BLOCKING_ID: Id numérico.

BEHAVIOUR_OPER_STAGE_ID: ID de **BEHAVIOUR_OPER_STAGE** que aplica la configuración.

DESCRIPTION: Descripción del Bloqueo.

CONDITION_ID: Id condición opcional pero siempre debería existir, sino existe debería existir el Id de **Grupo de Condiciones**. Si la evaluación de la Condición o Grupo de Condición es TRUE, se devuelve el mensaje de EXITO y CODIGO de EXITO configurado para esa Condición o Condiciones en el Response de **QuickWin**.

ORDER_BY: Orden o prioridad de ejecución.

SCHEDULE_ID: Id de Schedule cuando aplica.

VALID_FROM: Desde cuándo es aplica.

VALID_UNTIL: Hasta cuando es aplica.

STATUS: Estado A de Activo, I de Inactivo.

NOTIFICATION_GROUP_ID: Id de la configuración de notificación.

CONDITION_GROUP_ID: ID de Grupo de Condiciones.

SCRIPT PLSQL

CREATE BEHAVIOUR BLOCKING

```

declare

-- CONDITIONS

-- USAR ID DE CONDICION QUE SE NECESITE
ln_condition_id number := 1075;
ln_operation_type_id1 VARCHAR2(200) := 'activateOffer';

```



```

ln_campaign_id number := null;
lv_description_campaign varchar(255) := '[QWV1-MGR]%;
ln_operation_type varchar2(255) := '';
-- USAR ID DE INVOCADOR QUE SE NECESITE
lv_stage_id varchar2(255) := 'DYNAMIC_BLOCKING';
-- obligatorio definir
ln_behaviour_oper_id number := null;
ln_behaviour_oper_stage_id number := null;

--

begin

SELECT max(C.CAMPAIGN_ID) into ln_campaign_id
      FROM CAMPAIGN C
      WHERE C.LONG_DESCRIPTION LIKE lv_description_campaign;

-- Si se conocen los id, inicializarlos directamente
SELECT MIN(T.BEHAVIOUR_OPER_STAGE_ID), MIN(T.BEHAVIOUR_OPERATION_ID) INTO
ln_behaviour_oper_stage_id, ln_behaviour_oper_id
      FROM BEHAVIOUR_OPER_STAGES T, BEHAVIOUR_OPERATION BO
      WHERE T.BEHAVIOUR_OPERATION_ID = BO.BEHAVIOUR_OPERATION_ID
      AND T.STAGE_ID = lv_stage_id
      AND BO.OPERATION_TYPE_ID = ln_operation_type_id1
      and bo.behaviour_id = (SELECT MAX(C.BEHAVIOUR_ID)
      FROM CAMPAIGN C
      WHERE C.CAMPAIGN_ID = ln_campaign_id);

-- insertar

insert into behaviour_blockings (BEHAVIOUR_BLOCKING_ID, BEHAVIOUR_OPER_STAGE_ID,
DESCRIPTION, CONDITION_ID, ORDER_BY, SCHEDULE_ID, STATUS, NOTIFICATION_GROUP_ID,
CONDITION_GROUP_ID)
values ((SELECT NVL(MAX(T.behaviour_blocking_id),0) + 1 FROM behaviour_blockings
T), ln_behaviour_oper_stage_id, 'Bloquea si el Serial es invalido',
ln_condition_id, (SELECT NVL(MAX(T.order_by),0) + 10 FROM behaviour_blockings T
where t.behaviour_oper_stage_id = ln_behaviour_oper_stage_id ), null, 'A', null,
'');

dbms_output.put_line('ln_behaviour_oper_stage_id number :=
'||ln_behaviour_oper_stage_id||');

dbms_output.put_line('-----REFRESH-----
--');
dbms_output.put_line('http://192.168.37.146:8101/quickwin/clearCampaignById/'||
n_campaign_id);

dbms_output.put_line('-----SELECT-----
-');
dbms_output.put_line('SELECT * FROM behaviour_oper_stages T WHERE
T.BEHAVIOUR_OPERATION_ID = '||ln_behaviour_oper_id||' order by t.order_by;');
dbms_output.put_line('SELECT * FROM behaviour_blockings T WHERE
T.BEHAVIOUR_OPER_STAGE_ID = '||ln_behaviour_oper_stage_id||' order by
t.order_by;');

dbms_output.put_line('-----UPDATE STATUS-----
-----');

```

```

dbms_output.put_line('update behaviour_blockings t set status = 'I' where
t.BEHAVIOUR_OPER_STAGE_ID = '||ln_behaviour_oper_stage_id||');

dbms_output.put_line('-----DELETE-----');
dbms_output.put_line('delete behaviour_blockings b where
b.BEHAVIOUR_OPER_STAGE_ID = '||ln_behaviour_oper_stage_id||');

dbms_output.put_line('-----REQUEST-----');
dbms_output.put_line('http://192.168.37.146:8101/quickwin/getActivateRequest');

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

```

Salida.

```

ln_behaviour_oper_stage_id number := 1013;
-----REFRESH-----
http://192.168.37.146:8101/quickwin/clearCampaignById/249
-----SELECT-----
SELECT * FROM behaviour_oper_stages T WHERE T.BEHAVIOUR_OPERATION_ID = 306 order
by t.order_by;
SELECT * FROM behaviour_blockings T WHERE T.BEHAVIOUR_OPER_STAGE_ID = 1013 order
by t.order_by;
-----UPDATE STATUS-----
update behaviour_blockings t set status = 'I' where t.BEHAVIOUR_OPER_STAGE_ID =
1013;
-----DELETE-----
delete behaviour_blockings b where b.BEHAVIOUR_OPER_STAGE_ID = 1013;
-----REQUEST-----
http://192.168.37.146:8101/quickwin/getActivateRequest

```

DELETE BEHAVIOUR BLOCKING

El siguiente script elimina todas las configuraciones del STAGE DYNAMIC_BLOCKING para una operación.

```

-- Created on 17/05/2021 by MGARCIAR
declare
    lv_description_campaign varchar(255) := '[QWV1-MGR]';
    deleted_cant            number := 0;
begin

    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.DESCRPTION 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.DESCRPTION LIKE lv_description_campaign));

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

```

Resultado.

```

cantidad de behaviour_blockings a eliminar: 1

```

EJEMPLO DE RESPUESTA DE BEHAVIOUR BLOCKING.

En este ejemplo veremos básicamente información de la Condición en el Response de **QuickWin**. Es Código de Éxito y Mensaje de Éxito.

Dado que el Código de Éxito es positivo, esta transacción no guarda el request por defecto y entiende que debe abortar la transacción. Si se desea que se guarde el **request** de **QuickWin** e incluso sea candidato a reintento toda la operación de **QuickWin** se pueden configurar los códigos de éxito a nivel de la condición para que estos sean negativos o incluso puede usar como otra alternativa el STAGE de Validación que usa código de Error y Mensaje de Error por defecto.

Dentro de **userMessageResponse** se da detalle de la condición. Nótese que el código y mensaje devueltos son los de éxito de la condición.

```

{
  "activateMessageResponse": {
    "message": "[ DYNAMIC_BLOCKING ] La oferta se ha bloqueado. No se reconoce el canal y se espera uno de estos SGA/HUAWEI/PDA",
    "code": "2",
    "sessionData": {"externalSubscriberProperties": [
      {
        "id": "CUSTOMER_ID",
        "value": []
      },
      {
        "id": "MESSAGE",

```

```

        "value": []
      }
    ]}
  },
  "commonHeaderResponse": {"operationInfo": {
    "transactionId": "0094d0b2-fabe-4231-b0ee-7fc14ce7ebac",
    "transactionDate": "19/05/2021 09:09:36",
    "operationId": "291"
  }},
  "userMessageResponse": {
    "block": "DYNAMIC_BLOCKING",
    "type": "Condition",
    "id": 960,
    "success": true,
    "code": 0,
    "message": "No se reconoce el canal y se espera uno de estos
SGA/HUAWEI/PDA"
  }
}

```

```

{
  "activateMessageResponse": {
    "message": "[ DYNAMIC_BLOCKING ] La oferta se ha bloqueado. CUSTOMER NO
VALIDO",
    "code": "2",
    "sessionData": {"externalSubscriberProperties": [ {
      "id": "MESSAGE",
      "value": []
    }]}
  },
  "commonHeaderResponse": {"operationInfo": {
    "transactionId": "87d9793d-4f4d-4add-9a27-7d8d91e170f9",
    "transactionDate": "19/05/2021 10:37:29",
    "operationId": "271"
  }},
  "userMessageResponse": {
    "block": "DYNAMIC_BLOCKING",
    "type": "Condition",
    "id": 1074,
    "success": true,
    "code": 0,
    "message": "CUSTOMER NO VALIDO"
  }
}

```

BEVAHIOUR_VALIDATION

El objetivo de usar esta configuración es que en caso de que una condición se evalúe sea falsa el sistema aborta la transacción. Esta tabla es como el inverso de BEHAVIOUR_BLOCKING.

Adicional esta tabla involucra **Schedule**. Cuando Schedule esta configurado se debe cumplir la Condición y el Schedule. La **Condition** cuando es **NULL** es como una condición siempre verdadera y lo mismo aplica con Schedule, si no existe significa que se ejecuta a todas horas.

Cuando se configura **Condition Groups (CONDITION_GROUP_ID)**, se da prioridad al grupo de condiciones y se ignora a la condición configurada. En estos casos es preferible dejar a la condición como **NULL**.

NAME	TYPE	NULLABLE	DEFAULT
BEHAVIOUR_VALIDATION_ID	NUMBER	N	
BEHAVIOUR_OPER_STAGE_ID	NUMBER	Y	
DESCRIPTION	VARCHAR2(255)	Y	
CONDITION_ID	NUMBER	Y	
ORDER_BY	NUMBER	N	
SCHEDULE_ID	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'
NOTIFICATION_GROUP_ID	NUMBER	Y	
CONDITION_GROUP_ID	NUMBER	Y	

BEHAVIOUR_VALIDATION_ID: Id numérico.

BEHAVIOUR_OPER_STAGE_ID: ID de **BEHAVIOUR_OPER_STAGE** que aplica la configuración.

DESCRIPTION: Descripción del Validación.

CONDITION_ID: Id condición opcional pero siempre debería existir, sino existe debería existir el Id de **Grupo de Condiciones**. Si la evaluación de la Condición o Grupo de Condición es FALSE, se devuelve el mensaje de ERROR y CODIGO de ERROR configurado para esa Condición o Condiciones en el Response de **QuickWin**.

ORDER_BY: Orden o prioridad de ejecución.

SCHEDULE_ID: Id de Schedule cuando aplica.

VALID_FROM: Desde cuándo es aplica.

VALID_UNTIL: Hasta cuando es aplica.

STATUS: Estado A de Activo, I de Inactivo.

NOTIFICATION_GROUP_ID: Id de la configuración de notificación.

CONDITION_GROUP_ID: ID de Grupo de Condiciones.

SCRIPT PLSQL

CREATE BEHAVIOUR VALIDATION

```
declare

-- CONDITIONS

-- USAR ID DE CONDICION QUE SE NECESITE
ln_condition_id number := 1075;
ln_operation_type_id1 VARCHAR2(200) := 'activateOffer';
ln_campaign_id number := null;
lv_description_campaign varchar(255) := '[QWV1-MGR]%';
ln_operation_type varchar2(255) := '';
-- USAR ID DE INVOCADOR QUE SE NECESITE
lv_stage_id varchar2(255) := 'DYNAMIC_VALIDATION';
-- obligatorio definir
ln_behaviour_oper_id number := null;
ln_behaviour_oper_stage_id number := null;

--

begin

SELECT max(C.CAMPAIGN_ID) into ln_campaign_id
      FROM CAMPAIGN C
      WHERE C.LONG_DESCRIPTION LIKE lv_description_campaign;

-- Si se conocen los id, inicializarlos directamente
SELECT MIN(T.BEHAVIOUR_OPER_STAGE_ID), MIN(T.BEHAVIOUR_OPERATION_ID) INTO
ln_behaviour_oper_stage_id, ln_behaviour_oper_id
      FROM BEHAVIOUR_OPER_STAGES T, BEHAVIOUR_OPERATION BO
      WHERE T.BEHAVIOUR_OPERATION_ID = BO.BEHAVIOUR_OPERATION_ID
      AND T.STAGE_ID = lv_stage_id
      AND BO.OPERATION_TYPE_ID = ln_operation_type_id1
      and bo.behaviour_id = (SELECT MAX(C.BEHAVIOUR_ID)
      FROM CAMPAIGN C
      WHERE C.CAMPAIGN_ID = ln_campaign_id);

-- insertar

insert into behaviour_validations (BEHAVIOUR_VALIDATION_ID,
BEHAVIOUR_OPER_STAGE_ID, DESCRIPTION, CONDITION_ID, ORDER_BY, SCHEDULE_ID,
STATUS, NOTIFICATION_GROUP_ID, CONDITION_GROUP_ID)
values ((SELECT NVL(MAX(T.behaviour_validation_id),0) + 1 FROM
behaviour_validations T), ln_behaviour_oper_stage_id, 'valida que el codigo
serial no este ingresado en la base', ln_condition_id, ((SELECT
NVL(MAX(T.Order_By),0) + 10 FROM behaviour_validations T where
t.behaviour_oper_stage_id = ln_behaviour_oper_stage_id )), null, 'A', null, '');

dbms_output.put_line('ln_behaviour_oper_stage_id number :=
'||ln_behaviour_oper_stage_id||';');

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

```

dbms_output.put_line('http://192.168.37.146:8101/quickwin/clearCampaignById/'||t
n_campaign_id);

dbms_output.put_line('-----SELECT-----
-');
dbms_output.put_line('SELECT * FROM behaviour_oper_stages T WHERE
T.BEHAVIOUR_OPERATION_ID = '||tn_behaviour_oper_id||' order by t.order_by;');
dbms_output.put_line('SELECT * FROM behaviour_validations T WHERE
T.BEHAVIOUR_OPER_STAGE_ID = '||tn_behaviour_oper_stage_id||' order by
t.order_by;');

dbms_output.put_line('-----UPDATE STATUS-----
-----');
dbms_output.put_line('update behaviour_validations t set status = 'I' where
t.BEHAVIOUR_OPER_STAGE_ID = '||tn_behaviour_oper_stage_id||';');

dbms_output.put_line('-----DELETE-----
-');
dbms_output.put_line('delete behaviour_validations b where
b.BEHAVIOUR_OPER_STAGE_ID = '||tn_behaviour_oper_stage_id||';');

dbms_output.put_line('-----REQUEST-----
-----');
dbms_output.put_line('http://192.168.37.146:8101/quickwin/getActivateRequest');

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

```

salida.

```

tn_behaviour_oper_stage_id number := 1014;
-----REFRESH-----
http://192.168.37.146:8101/quickwin/clearCampaignById/249
-----SELECT-----
SELECT * FROM behaviour_oper_stages T WHERE T.BEHAVIOUR_OPERATION_ID = 306 order
by t.order_by;
SELECT * FROM behaviour_validations T WHERE T.BEHAVIOUR_OPER_STAGE_ID = 1014
order by t.order_by;
-----UPDATE STATUS-----
update behaviour_validations t set status = 'I' where t.BEHAVIOUR_OPER_STAGE_ID
= 1014;
-----DELETE-----
delete behaviour_validations b where b.BEHAVIOUR_OPER_STAGE_ID = 1014;
-----REQUEST-----
http://192.168.37.146:8101/quickwin/getActivateRequest

```

DELETE BEHAVIOUR VALIDATION

El siguiente script elimina todas las configuraciones del STAGE **DYNAMIC_VALIDATION** para una operación.

```

-- Created on 17/05/2021 by MGARCIAR
declare

```

```

lv_description_campaign varchar(255) := '[QWV1-MGR]';
lv_stage_id varchar2(255) := 'DYNAMIC_VALIDATION';
deleted_cant          number := 0;
begin

    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 = lv_stage_id 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.DESCRPTION 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 = lv_stage_id 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.DESCRPTION LIKE lv_description_campaign));

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

```

SALIDA

```
cantidad de behaviour_validations a eliminar: 1
```

EJEMPLO DE RESPUESTA DE BEHAVIOUR VALIDATION

En este ejemplo veremos básicamente información de la Condición en el Response de **QuickWin**. Es Código de Error y Mensaje de Error por defecto.

Dado que el Código de Error es negativo, esta transacción guarda el request por defecto y entiende que es candidato a Reintenta toda la Operacion de Quickwin. Si se desea que no se guarde el **request** de **QuickWin** e incluso no sea candidato a reintento toda la operación de **QuickWin** se pueden configurar los códigos de Error a nivel de la condición para que estos sean positivos o incluso puede usar como otra alternativa el STAGE de **Blocking** que usa código de Éxito y Mensaje de Éxito por defecto.

```

{
  "activateMessageResponse": {
    "message": "[ DYNAMIC_VALIDATION ] No se cumplio la validacion: No se ha
provisto Nombres",
    "code": "1",

```



```

    "sessionData": {"externalSubscriberProperties": [
        {
            "id": "CUSTOMER_ID",
            "value": []
        },
        {
            "id": "MESSAGE",
            "value": []
        }
    ]}
},
"commonHeaderResponse": {"operationInfo": {
    "transactionId": "a73448a3-4436-4240-b898-c74d928fc08f",
    "transactionDate": "19/05/2021 11:18:10",
    "operationId": "291"
}},
"userMessageResponse": {
    "block": "DYNAMIC_VALIDATION",
    "type": "Condition",
    "id": 1092,
    "success": false,
    "code": -121,
    "message": "No se ha provisto Nombres"
}
}

```

BEVAHIOUR_OPER_INVOKE

El objetivo de usar esta configuración es poder llamar uno o varios Invokes de forma estáticas o dinámicas usando lazos. (Looping Operation). Lazos se explica al inicio de este documento.

NAME	TYPE	NULLABLE	DEFAULT	COMMENTS
BEHAVIOUR_OPER_INVOKE_ID	NUMBER	N		
BEHAVIOUR_OPER_STAGE_ID	NUMBER	Y		
WHEN	VARCHAR2(4)	Y	'PRE'	
CONDITION_ID	NUMBER	Y		
INVOKE_ID	NUMBER	Y		
ORDER_BY	NUMBER	Y	10	
REQUIRED	VARCHAR2(3)	Y	'YES'	
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'	
SYNC	VARCHAR2(3)	Y	'YES'	
NOTIFICATION_GROUP_ID	NUMBER	Y		
SESSION_DATA	CLOB	Y		
SCHEDULE_ID	NUMBER	Y		En que momento se debe ejecutar
WHEN_CONDITION	VARCHAR2(5)	Y		Si la condicion es TRUE, seguir, Si la condicion es FALSE seguir
WHEN_SCHEDULE	VARCHAR2(5)	Y		Si el Schedule es TRUE, seguir, Si el Schedule es FALSE seguir
WHEN_RESP_NOT_SUCCESS	VARCHAR2(10)	Y		Si la respuesta no es exitosa, marcarla como WARNING/ERROR
RETRIES	NUMBER	Y	0	
APPLY_COMPENSATION	VARCHAR2(3)	Y	'NO'	Campo que indica si la linea de invoke aplica compensacion
INVOKE_ID_COMP_BACKWARD	NUMBER	Y		En caso de que los reintentos sean 0 o fallen todos los reintentos, se ejecuta este invoke el cual debe ser usado para hacer rollback
INVOKE_ID_COMP_FORWARD	NUMBER	Y		En el caso que el invoke tenga reintentos y este sea exitoso, se ejecuta esta invocacion la cual debe ser usada para continuar la transaccion.
CONDITION_GROUP_ID	NUMBER	Y		
LOOPING_OPERATION_ID	NUMBER	Y		Para hacer llamadas recursivas
COPY_FROM_SUBS_TO_SUBS	CLOB	Y		Para copiar variables

BEHAVIOUR_OPER_INVOKE_ID: Id numérico.

BEHAVIOUR_OPER_STAGE_ID: Stage

CONDITION_ID: Id condición opcional.

INVOKE_ID: Id de Invoke o Invocador.

ORDER_BY: Orden de Ejecución.

REQUIRED: YES o NO. YES es obligatorio y si falla su evaluación de Invocador se devuelve el mensaje de error configurado para ese invocador en el Response de QuickWin. NO, ejecuta de forma Asíncrona.

VALID_FROM: Desde cuándo es aplica.

VALID_UNTIL: Hasta cuando es aplica.

STATUS: Estado A de Activo, I de Inactivo.

NOTIFICATION_GROUP_ID: Id de la configuración de notificación.

CONDITION_GROUP_ID: ID de Grupo de Condiciones.

SESSION_DATA: [] Representa una lista de **Subscriber Properties** con sus valores respectivos a inicializar.

SCHEDULE_ID: ID de Schedule. Si el Schedule está configurado y la condicion tambien lo está, ambos deben cumplirse. **NOTA:** En esta versión no está habilitado.

WHEN_CONDITION: Evaluar una condición por **FALSE** o **TRUE**. **TRUE** es por defecto. Aplica tambien negar grupo de condiciones.

WHEN_SCHEDULE: Evaluar Schedule por **FALSE** o **TRUE**. **TRUE** es por defecto. **NOTA:** En esta versión no está habilitado.

EJEMPLO DE BEHAVIOR OPER INVOKE FALLIDO.

En este ejemplo veremos básicamente información del Invocador devuelta en el Response de **QuickWin**. Es Código de Error y Mensaje de Error del Invocador cuando este falla por algún error general definido por **QuickWin** internamente. En muchas ocasiones este código puede ser positivo o negativo dependiendo de como se configure la evaluación de la respuesta de Invocador por FALLAS. Se recomienda que sea positivo cuando la transacción debe terminar o cancelarse, y negativo si se desea un reintento de toda la transacción. Recuerde algunos reintentos necesitan primero compensar, asegúrese que cuando aplique reintentos la compensación se haya hecho antes.

```
{
  "activateMessageResponse": {
    "message": "[ Operacion POST OPERATION_INVOKE ] WS de OPERATION es obligatorio: -> SRE de FEEDBACK SGA no se ejecuto exitosamente",
    "code": "-221",
    "sessionData": {"externalSubscriberProperties": [
      {
        "id": "COD_RESPUESTA",
        "value": ["400"]
      },
      {
        "id": "MESSAGE",
        "value": ["Ninguna Operacion Realizada. Asegurese que la operacion sea PROVISINING_AMCO o UPDATE_SERIAL"]
      },
      {
        "id": "EXTERNAL_TRANSACTION",
        "value": ["8555885555"]
      }
    ]}
  }
}
```

```

    },
    {
      "id": "QUICKWIN_TRANSACTION_ID",
      "value": []
    },
    {
      "id": "QUICKWIN_OPERATION_ID",
      "value": []
    }
  ]}
},
"commonHeaderResponse": {"operationInfo": {
  "transactionId": "4b404021-d9a0-431a-9605-17d461856d11",
  "transactionDate": "19/05/2021 09:08:46",
  "operationId": "290"
}},
"userMessageResponse": {
  "block": "OPERATION_INVOKE",
  "type": "Invoke",
  "id": 710,
  "success": false,
  "code": -221,
  "message": "SRE de FEEDBACK SGA no se ejecuto exitosamente"
}
}

```

CODIGO POSITIVO DE INVOCADOR

```

{
  "activateMessageResponse": {
    "message": "[ Operacion POST OPERATION_INVOKE ] WS de OPERATION es obligatorio: -> No se pudo activar la oferta",
    "code": "14",
    "sessionData": {"externalSubscriberProperties": [
      {
        "id": "ORDEN",
        "value": [""]
      },
      {
        "id": "ERRORMESSAGE",
        "value": ["The amount of payment can't be empty."]
      }
    ]}
  },
  "commonHeaderResponse": {"operationInfo": {
    "transactionId": "181ab3b5-1879-410c-bdae-eb4bea2e2623",
    "transactionDate": "19/05/2021 10:44:48",
    "operationId": "87"
  }},
  "userMessageResponse": {
    "block": "OPERATION_INVOKE",
    "type": "Invoke",
    "id": 149,
    "success": false,
    "code": 14,
    "message": "No se pudo activar la oferta"
  }
}

```

```
}
```

SCRIPT PLSQL

CREATE BEHAVIOUR OPER INVOKE

```
-- Created on 21/05/2021 by MGARCIAR
declare
-- USAR ID DE CONDICION QUE SE NECESITE
ln_condition_id number := 1075;
-- usar los invoke reales
ln_invoke_id number := 721;
ln_operation_type_id1 VARCHAR2(200) := 'activateOffer';
ln_campaign_id number := null;
lv_description_campaign varchar(255) := '[QWV1-MGR]%;
ln_operation_type varchar2(255) := '';
-- USAR ID DE INVOCADOR QUE SE NECESITE
lv_stage_id varchar2(255) := 'OPERATION_INVOKE';
-- obligatorio definir
ln_behaviour_oper_id number := null;
ln_behaviour_oper_stage_id number := null;

--

begin

SELECT max(C.CAMPAIGN_ID) into ln_campaign_id
      FROM CAMPAIGN C
      WHERE C.LONG_DESCRIPTION LIKE lv_description_campaign;

-- si se conocen los id, inicializarlos directamente
SELECT MIN(T.BEHAVIOUR_OPER_STAGE_ID), MIN(T.BEHAVIOUR_OPERATION_ID) INTO
ln_behaviour_oper_stage_id, ln_behaviour_oper_id
      FROM BEHAVIOUR_OPER_STAGES T, BEHAVIOUR_OPERATION BO
      WHERE T.BEHAVIOUR_OPERATION_ID = BO.BEHAVIOUR_OPERATION_ID
      AND T.STAGE_ID = lv_stage_id
      AND BO.OPERATION_TYPE_ID = ln_operation_type_id1
      and bo.behaviour_id = (SELECT MAX(C.BEHAVIOUR_ID)
      FROM CAMPAIGN C
      WHERE C.CAMPAIGN_ID = ln_campaign_id);

-- insertar
insert into BEHAVIOUR_OPER_INVOKE (BEHAVIOUR_OPER_INVOKE_ID,
BEHAVIOUR_OPER_STAGE_ID, WHEN, CONDITION_ID, INVOKE_ID, ORDER_BY, REQUIRED,
STATUS, SYNC, NOTIFICATION_GROUP_ID, SESSION_DATA, SCHEDULE_ID, WHEN_CONDITION,
WHEN_SCHEDULE, WHEN_RESP_NOT_SUCCESS, RETRIES, APPLY_COMPENSATION,
INVOKE_ID_COMP_BACKWARD, INVOKE_ID_COMP_FORWARD, CONDITION_GROUP_ID,
LOOPING_OPERATION_ID, COPY_FROM_SUBS_TO_SUBS)
values ((SELECT NVL(MAX(T.behaviour_oper_invoke_id),0) + 1 FROM
BEHAVIOUR_OPER_INVOKE T), ln_behaviour_oper_stage_id, 'PRE', ln_condition_id,
ln_invoke_id, (SELECT NVL(MAX(T.Order_By),0) + 10 FROM BEHAVIOUR_OPER_INVOKE T
where t.behaviour_oper_stage_id = ln_behaviour_oper_stage_id ), 'YES', 'A',
'YES', null, '', null, 'FALSE', '', '', 0, 'NO', '', '', '', null, '');

dbms_output.put_line('ln_behaviour_oper_stage_id number :=
'||ln_behaviour_oper_stage_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('-----SELECT-----
-');
dbms_output.put_line('SELECT * FROM behaviour_oper_stages T WHERE
T.BEHAVIOUR_OPERATION_ID = '||ln_behaviour_oper_id||' order by t.order_by;');
dbms_output.put_line('SELECT * FROM BEHAVIOUR_OPER_INVOKE T WHERE
T.BEHAVIOUR_OPER_STAGE_ID = '||ln_behaviour_oper_stage_id||' order by
t.order_by;');

dbms_output.put_line('-----UPDATE STATUS-----
-----');
dbms_output.put_line('update BEHAVIOUR_OPER_INVOKE t set status = 'I' where
t.BEHAVIOUR_OPER_STAGE_ID = '||ln_behaviour_oper_stage_id||';');

dbms_output.put_line('-----DELETE-----
-');
dbms_output.put_line('delete BEHAVIOUR_OPER_INVOKE b where
b.BEHAVIOUR_OPER_STAGE_ID = '||ln_behaviour_oper_stage_id||';');

dbms_output.put_line('-----REQUEST-----
-----');
dbms_output.put_line('http://192.168.37.146:8101/quickwin/getActivateRequest');

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

```

EJEMPLO AVANZADO

```

insert into BEHAVIOUR_OPER_INVOKE (BEHAVIOUR_OPER_INVOKE_ID,
BEHAVIOUR_OPER_STAGE_ID, WHEN, CONDITION_ID, INVOKE_ID, ORDER_BY, REQUIRED,
STATUS, SYNC, NOTIFICATION_GROUP_ID, SESSION_DATA, SCHEDULE_ID, WHEN_CONDITION,
WHEN_SCHEDULE, WHEN_RESP_NOT_SUCCESS, RETRIES, APPLY_COMPENSATION,
INVOKE_ID_COMP_BACKWARD, INVOKE_ID_COMP_FORWARD, CONDITION_GROUP_ID,
LOOPING_OPERATION_ID, COPY_FROM_SUBS_TO_SUBS)
values ((SELECT NVL(MAX(T.behaviour_oper_invoke_id),0) + 1 FROM
BEHAVIOUR_OPER_INVOKE T), ln_behaviour_oper_stage_964, 'PRE', '', ln_invoke_712,
20, 'YES', 'A', 'YES', null, '{"id": "COD_RESPUESTA_TMP","value": ["200"]}',
null, 'TRUE', '', '', 0, 'YES', ln_invoke_710, '', ln_condition_group_id_95,
null, '{
    "afterInvoke": [
        {
            "subscriberPropertyIdTo": "COD_RESPUESTA",
            "subscriberPropertyIdFrom": "COD_RESPUESTA_TMP",
            "copyStrategy": "SET",
            "index": "FIRST"
        }
    ],
    "beforeInvoke": [
    ]
}')');

```

RESULTADO

```
lv_behaviour_oper_stage_id number := 1012;
-----REFRESH-----
http://192.168.37.146:8101/quickwin/clearCampaignById/249
-----SELECT-----
SELECT * FROM behaviour_oper_stages T WHERE T.BEHAVIOUR_OPERATION_ID = 306 order
by t.order_by;
SELECT * FROM BEHAVIOUR_OPER_INVOKE T WHERE T.BEHAVIOUR_OPER_STAGE_ID = 1012
order by t.order_by;
-----UPDATE STATUS-----
update BEHAVIOUR_OPER_INVOKE t set status = 'I' where t.BEHAVIOUR_OPER_STAGE_ID
= 1012;
-----DELETE-----
delete BEHAVIOUR_OPER_INVOKE b where b.BEHAVIOUR_OPER_STAGE_ID = 1012;
-----REQUEST-----
http://192.168.37.146:8101/quickwin/getActivateRequest
```

DELETE BEHAVIOR OPER INVOKE

```
-- Created on 21/05/2021 by MGARCIAR
declare
lv_description_campaign varchar(255) := '[QWV1-MGR]';
deleted_cant             number := 0;
lv_stage_id varchar2(255) := 'OPERATION_INVOKE';

begin

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 = lv_stage_id 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.DESCRPTION 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_invoke_id in
(select t.behaviour_oper_stage_id
from behaviour_oper_stages t
where t.stage_id = lv_stage_id 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.DESCRPTION LIKE lv_description_campaign));

--chequear los errores
exception when others then
```

La lógica de Aprovisionamiento debe resolverse en la BEHAVIOUR_PRODUCT que es quien define que se debe aprovisionar, ya sea un PRODUCTO de una OFERTA X o todos los PRODUCTOS de una OFERTA. El campo PROVISIONING_BY indica que si lo que quiero entregar es una OFERTA o un PRODUCTO (OFFER/PRODUCT). Entiéndase que al entregar una oferta, realmente estoy entregando TODOS los productos que son los que pertenecen a dicha oferta. De todos estos productos que se deben entregar se obtiene su tipo de producto y este tipo de producto debe ser consultado en la tabla **PRODUCT TYPE PROVISIONING**.

PRODUCT_TYPE_PROVISIONING no es solo es una tabla de consulta también aplica lógica de procesos sobre un Invocador. Puede hacer lógica de compensación y reintentos. Adicional evaluar condiciones o grupo de condiciones así como también Schedule, notificaciones e incluso inicializar **Subscriber Properties** usando la columna de **SESSION_DATA**.

QUICKWIN_MDS.PRODUCT_TYPE_PROVISIONING

NAME	TYPE	NULLABLE	DEFAULT	COMMENTS
PRODUCT_TYPE_PROVISIONING_ID	NUMBER	N		
PRODUCT_TYPE	VARCHAR2(255)	Y		
INVOKE_ID	NUMBER	N		
ORDER_BY	NUMBER	Y	10	
SYNC	VARCHAR2(3)	Y	'YES'	
CONDITION_ID	NUMBER	Y		
SESSION_DATA	CLOB	Y		
NOTIFICATION_GROUP_ID	NUMBER	Y		
REQUIRED	VARCHAR2(3)	Y	'YES'	
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'	
EXTERNAL_OFFER_ID	VARCHAR2(255)	Y		
WHEN_CONDITION	VARCHAR2(5)	Y		Si la condicion es TRUE, seguir, Si la condicion es FALSE seguir
WHEN_RESP_NOT_SUCCESS	VARCHAR2(10)	Y		Si la respuesta no es exitosa, marcarla como WARNING/ERROR
SYNC_RETRIES	NUMBER	Y	1	Numero de veces que un servicio se reintenta en caso de fallo en el bloque de PROVISIONING en modo Sincrono. En Asincrono utilizar los reintentos del invocador
APPLY_COMPENSATION	VARCHAR2(3)	Y	'NO'	Campo que indica si la linea de aprovisionamiento aplica compensacion
INVOKE_ID_COMP_BACKWARD	NUMBER	Y		En caso de que los reintentos sean 0 o fallen todos los reintentos, se ejecuta este invoke el cual debe ser usado para hacer rollback
INVOKE_ID_COMP_FORWARD	NUMBER	Y		En el caso que el aprovisionamiento tenga reintentos y este sea exitoso, se ejecuta esta invocacion la cual debe ser usada para continuar la transaccion.

PRODUCT_TYPE_PROVISIONING_ID:

PRODUCT_TYPE: Tipo de Producto

INVOKE_ID: Id de Invocador.

ORDER_BY: Orden de ejecución

SYNC: **YES** es sincrónico, **NO** es asíncrono.

CONDITION_ID: Id de condición a Evaluarse.

SESSION_DATA: Lista de Subscriber Properties.

NOTIFICATION_GROUP_ID: Id de grupo de notificaciones. Se utiliza para enviar notificaciones con respecto a las respuestas de los invocadores en caso que los invocadores respondan con códigos de **WARNING, ERROR o SUCCESS**.

REQUIRED: **YES** obligatorio en caso que la evaluación del invoke no es **SUCCESS**. **NO**, No considerar evaluación y si hay errores continuar.

SCHEDULE_ID: Id de Schedule cuando aplica. Si el Schedule está configurado y la condición también lo está, ambos deben cumplirse.

VALID_FROM: Desde cuándo es aplica.

VALID_UNTIL: Hasta cuando es aplica.

STATUS: Estado A de Activo, I de Inactivo.

EXTERNAL_OFFER_ID: No se usa.

WHEN_RESP_NOT_SUCCESS: Sirve para cambiar el código de evaluación del response por uno negativo. **DESCONTINUADO** puesto que los invocadores ahora manejan sus propios códigos personalizados.

WHEN_CONDITION: Evaluar una condición por **FALSE** o **TRUE**. **TRUE** es por defecto. Aplica también negar grupo de condiciones.

WHEN_SCHEDULE: Evaluar Schedule por **FALSE** o **TRUE**. **TRUE** es por defecto.

CONDITION_GROUP_ID: ID de Grupo de Condiciones. Cuando se configura, se ignora a la condición.

QUICKWIN_MDS.BEHAVIOUR_PRODUCT

NAME	TYPE	NULLABLE	DEFAULT	COMMENTS
BEHAVIOUR_PRODUCT_ID	NUMBER	N		
BEHAVIOUR_OPER_STAGE_ID	NUMBER	Y		
SHORT_DESCRIPTION	VARCHAR2(255)	Y		
PROVISION_BY	VARCHAR2(7)	Y	'OFFER'	
OFFER_ID	NUMBER	Y		
PRODUCT_ID	VARCHAR2(255)	Y		
CONDITION_ID	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'	
ORDER_BY	NUMBER	Y	10	
SCHEDULE_ID	NUMBER	Y		En que momento se debe ejecutar
WHEN_CONDITION	VARCHAR2(5)	Y		Si la condicion es TRUE, seguir, Si la condicion es FALSE seguir
WHEN_SCHEDULE	VARCHAR2(5)	Y		Si el Schedule es TRUE, seguir, Si el Schedule es FALSE seguir
OFFER_RECURRENCY_ID	NUMBER	Y		Indica el ID de recurrencia
DATE_TO_EVALUATE	VARCHAR2(255)	Y	'[\$TRANSACTIONDATE\$]'	Key del subscriber properties a pasar al calendarizador. Si es nulo o el valor por defecto en el campo date_format el valor debe ser nulo o su valor por defecto.
DATE_FORMAT	VARCHAR2(255)	Y	'dd/MM/yyyy HH:mm:ss'	Formato de la fecha
CHILD_CALENDAR_PARAMS	CLOB	Y		Parametros utilizados en el calendario
CONDITION_GROUP_ID	NUMBER	Y		Grupo de Condiciones que debe cumplirse para aprovisionar producto, este campo es exclusivo con CONDITION_ID. Este campo tiene mayor precedencia

BEHAVIOUR_PRODUCT_ID: ID numérico.

BEHAVIOUR_OPER_STAGE_ID: ID de **BEHAVIOUR_OPER_STAGE** que aplica la configuración.

SHORT_DESCRIPTION: Descripcion corta.

PROVISION_BY: Indica que el tipo de aprovisionamiento va estar dado por **OFFER** o **PRODUCT**. Cuando se especifica **PRODUCT**, se espera que el **PRODUCT_ID** exista en la **OFFER_ID** especificada. En este caso el producto puede ser de cualquier oferta. Si se especifica la oferta entonces ese producto debe existir en esa oferta. Sin el producto no esta asociado a ninguna

oferta, o no se desea validar contra la oferta entonces OFFER_ID es NULL o VACIO. Es opcional ubicar la oferta en **OFFER_ID**.

OFFER_ID: Null o la oferta del producto. Es obligatorio cuando en **PROVISION_BY** es **OFFER**. Recuerde que al aprovisionar por OFFER estamos dando todos los productos de la oferta. Si la oferta tiene un solo producto, da lo mismo aprovisionarlo por la oferta o por producto.

PRODUCT_ID: Es obligatorio cuando **PROVISION_BY** es **PRODUCT**. Se ignora cuando **PROVISION_BY** es **OFFER**

CONDITION_ID: Id de condicion a Evaluarse.

SCHEDULE_ID: Id de Schedule cuando aplica. Si el Schedule está configurado y la condicion tambien lo está, ambos deben cumplirse.

VALID_FROM: Desde cuándo es aplica.

VALID_UNTIL: Hasta cuando es aplica.

STATUS: Estado A de Activo, I de Inactivo.

WHEN_CONDITION: Evaluar una condición por **FALSE** o **TRUE**. **TRUE** es por defecto. Aplica tambien negar grupo de condiciones.

WHEN_SCHEDULE: Evaluar Schedule por **FALSE** o **TRUE**. **TRUE** es por defecto.

CONDITION_GROUP_ID: ID de Grupo de Condiciones. Cuando se configura, se ignora a la condicion.

OFFER_RECURRENCY_ID: Para feature de recurrencia, basado en calendario. Aun no liberado.

DATE_TO_EVALUATE: Para feature de recurrencia, basado en calendario. Aun no liberado.

DATE_FORMAT: Para feature de recurrencia, basado en calendario. Aun no liberado.

CHILD_CALENDAR_PARAMS: Para feature de recurrencia, basado en calendario. Aun no liberado.

ERROR DE PROVISIONING

En este ejemplo veremos básicamente información del Invocador devuelta en el Response de **QuickWin**. Es Código de Error y Mensaje de Error del Invocador cuando este falla por algún error general definido por **QuickWin** internamente. En muchas ocasiones este código puede ser positivo o negativo dependiendo de como se configure la evaluación de la respuesta de Invocador por FALLAS. Se recomienda que sea positivo cuando la transacción debe terminar o cancelarse, y negativo si se desea un reintento de toda la transacción. Recuerde algunos reintentos necesitan primero compensar, asegúrese que cuando aplique reintentos la compensación se haya hecho antes.

```
{
  "activateMessageResponse": {
    "message": "[ PROVISIONING ] Error al consumir servicio de activacion de Huawei",
    "code": "14",
    "sessionData": {"externalSubscriberProperties": []}
  },
  "commonHeaderResponse": {"operationInfo": {
    "transactionId": "2786a977-b16b-465a-8f16-ae863ff185bb",
    "transactionDate": "19/05/2021 10:21:44",
    "operationId": "203"
  }}
},
```

```

    "userMessageResponse": {
        "block": "PROVISIONING",
        "type": "Invoke",
        "id": 323,
        "success": false,
        "code": 14,
        "message": "Error al consumir servicio de activacion de Huawei"
    }
}

```

PLSQL

CREAR BEHAVIOUR PRODUCT Y PRODUCT TYPE PROVISIONING

```

-- Created on 21/05/2021 by MGARCIAR
declare
-- USAR ID DE CONDICION QUE SE NECESITE
ln_condition_id number := 1075;
-- usar los invoke reales
ln_invoke_id number := 715;
ln_operation_type_id1 VARCHAR2(200) := 'activateOffer';
ln_campaign_id number := null;
lv_description_campaign varchar(255) := '[QWV1-MGR]';
ln_operation_type varchar2(255) := '';
-- USAR ID DE INVOCADOR QUE SE NECESITE
lv_product_id varchar2(255) := '';
lv_product_type_id varchar2(255) := '';

-- obligatorio definir
lv_stage_id varchar2(255) := 'PROVISIONING';
ln_behaviour_oper_id number := null;
ln_behaviour_oper_stage_id number := null;

--

begin

SELECT max(C.CAMPAIGN_ID) into ln_campaign_id
      FROM CAMPAIGN C
      WHERE C.LONG_DESCRIPTION LIKE lv_description_campaign;

-- Si se conocen los id, inicializarlos directamente
SELECT MIN(T.BEHAVIOUR_OPER_STAGE_ID), MIN(T.BEHAVIOUR_OPERATION_ID) INTO
ln_behaviour_oper_stage_id, ln_behaviour_oper_id
      FROM BEHAVIOUR_OPER_STAGES T, BEHAVIOUR_OPERATION BO
      WHERE T.BEHAVIOUR_OPERATION_ID = BO.BEHAVIOUR_OPERATION_ID
      AND T.STAGE_ID = lv_stage_id
      AND BO.OPERATION_TYPE_ID = ln_operation_type_id1
      and bo.behaviour_id = (SELECT MAX(C.BEHAVIOUR_ID)
                             FROM CAMPAIGN C
                             WHERE C.CAMPAIGN_ID = ln_campaign_id);
dbms_output.put_line('ln_behaviour_oper_stage_id number :=
'||ln_behaviour_oper_stage_id||');

SELECT DISTINCT P.PRODUCT_TYPE_ID, P.PRODUCT_ID into
lv_product_type_id, lv_product_id
      FROM OFFER_PRODUCTS OP, PRODUCTS P

```

```

WHERE P.PRODUCT_ID = OP.PRODUCT_ID
AND OP.OFFER_ID IN
(SELECT CO.OFFER_ID FROM CAMPAIGN_OFFERS CO WHERE CO.CAMPAIGN_ID =
ln_campaign_id);
dbms_output.put_line('lv_product_type_id VARCHAR2(200) :=
'||lv_product_type_id||');
dbms_output.put_line('lv_product_id VARCHAR2(200) := '||lv_product_id||');

-- insertar product_type_provisioning
insert into product_type_provisioning (PRODUCT_TYPE_PROVISIONING_ID,
PRODUCT_TYPE, INVOKE_ID, ORDER_BY, SYNC, CONDITION_ID, SESSION_DATA, REQUIRED,
STATUS, EXTERNAL_OFFER_ID, WHEN_CONDITION, WHEN_RESP_NOT_SUCCESS, SYNC_RETRIES,
APPLY_COMPENSATION, INVOKE_ID_COMP_BACKWARD, INVOKE_ID_COMP_FORWARD)
values ((SELECT NVL(MAX(PP.PRODUCT_TYPE_PROVISIONING_ID),0) + 1 FROM
PRODUCT_TYPE_PROVISIONING PP), lv_product_type_id, ln_invoke_id, (SELECT
NVL(MAX(PP.Order_By),0) + 1 FROM PRODUCT_TYPE_PROVISIONING PP WHERE
PP.PRODUCT_TYPE = lv_product_type_id ), 'YES', ln_condition_id, '', 'YES', 'A',
'', '', 0, 'NO', null, null);

-- INSERTAR BEHAVIOUR_PRODUCT

insert into behaviour_product (BEHAVIOUR_PRODUCT_ID, BEHAVIOUR_OPER_STAGE_ID,
SHORT_DESCRIPTION, PROVISION_BY, OFFER_ID, PRODUCT_ID, CONDITION_ID,ORDER_BY,
SCHEDULE_ID, WHEN_CONDITION, WHEN_SCHEDULE, OFFER_RECURRENCY_ID,
DATE_TO_EVALUATE, DATE_FORMAT, CHILD_CALENDAR_PARAMS, CONDITION_GROUP_ID)
values ((SELECT NVL(MAX(t.behaviour_product_id),0) + 1 FROM behaviour_product
t), ln_behaviour_oper_stage_id, 'Provisioning IPTV', 'PRODUCT', null,
lv_product_id, null, (SELECT NVL(MAX(t.order_by),0) + 10 FROM behaviour_product
t where t.behaviour_oper_stage_id = ln_behaviour_oper_stage_id ), null, '', '',
null, '[$TRANSACTIONDATE$]', 'dd/MM/yyyy HH:mm:ss', null, null);

dbms_output.put_line('-----REFRESH-----
--');
dbms_output.put_line('http://192.168.37.146:8101/quickwin/clearCampaignById/'||
n_campaign_id);

dbms_output.put_line('-----SELECT-----
-');
dbms_output.put_line('SELECT * FROM behaviour_oper_stages T WHERE
T.BEHAVIOUR_OPERATION_ID = '||ln_behaviour_oper_id||' order by t.order_by;');
dbms_output.put_line('SELECT * FROM BEHAVIOUR_PRODUCT T WHERE
T.BEHAVIOUR_OPER_STAGE_ID = '||ln_behaviour_oper_stage_id||' order by
t.order_by;');
dbms_output.put_line('SELECT * FROM product_type_provisioning T WHERE
T.PRODUCT_TYPE = ''||lv_product_type_id||'' order by t.order_by;');

dbms_output.put_line('-----UPDATE STATUS-----
-----');
dbms_output.put_line('update BEHAVIOUR_PRODUCT t set status = 'I' where
t.BEHAVIOUR_OPER_STAGE_ID = '||ln_behaviour_oper_stage_id||');
dbms_output.put_line('update product_type_provisioning t set status = 'I'
where t.PRODUCT_TYPE = ''||lv_product_type_id||'';');

dbms_output.put_line('-----DELETE-----
-');
dbms_output.put_line('delete BEHAVIOUR_PRODUCT b where b.BEHAVIOUR_OPER_STAGE_ID
= '||ln_behaviour_oper_stage_id||');

```

```

dbms_output.put_line('delete product_type_provisioning b where b.PRODUCT_TYPE =
'''||lv_product_type_id||''';');

dbms_output.put_line('-----REQUEST-----');
dbms_output.put_line('http://192.168.37.146:8101/quickwin/getActivateRequest');

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

```

SALIDA.

Los delete borran todas las configuraciones asociado a ese bloque de stage.

```

lv_behaviour_oper_stage_id number := 1015;
lv_product_type_id VARCHAR2(200) := PRT_TEST1_QW;
lv_product_id VARCHAR2(200) := PR_TEST1_QW;
-----REFRESH-----
http://192.168.37.146:8101/quickwin/clearCampaignById/249
-----SELECT-----
SELECT * FROM behaviour_oper_stages T WHERE T.BEHAVIOUR_OPERATION_ID = 306 order
by t.order_by;
SELECT * FROM BEHAVIOUR_PRODUCT T WHERE T.BEHAVIOUR_OPER_STAGE_ID = 1015 order
by t.order_by;
SELECT * FROM product_type_provisioning T WHERE T.PRODUCT_TYPE = 'PRT_TEST1_QW'
order by t.order_by;
-----UPDATE STATUS-----
update BEHAVIOUR_PRODUCT t set status = 'I' where t.BEHAVIOUR_OPER_STAGE_ID =
1015;
update product_type_provisioning t set status = 'I' where t.PRODUCT_TYPE =
'PRT_TEST1_QW';
-----DELETE-----
delete BEHAVIOUR_PRODUCT b where b.BEHAVIOUR_OPER_STAGE_ID = 1015;
delete product_type_provisioning b where b.PRODUCT_TYPE = 'PRT_TEST1_QW';
-----REQUEST-----
http://192.168.37.146:8101/quickwin/getActivateRequest

```

ELIMINAR BEHAVIOUR PRODUCT Y PRODUCT TYPE PROVISIONING

```

-- Created on 21/05/2021 by MGARCIAR
declare
    lv_description_campaign varchar(255) := '[QWV1-MGR]';
    deleted_cant            number := 0;
    lv_stage_id varchar2(255) := 'PROVISIONING';
    lv_product_id varchar2(255) := '';

begin

    SELECT count(t.Behaviour_Product_Id) into deleted_cant from BEHAVIOUR_PRODUCT
t
    where t.behaviour_oper_stage_id in
        (select t.behaviour_oper_stage_id
         from behaviour_oper_stages t

```

```

        where t.stage_id = lv_stage_id 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.DESCRPTION LIKE lv_description_campaign));

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

    SELECT t.Product_Id into lv_product_id from BEHAVIOUR_PRODUCT t
    where t.behaviour_oper_stage_id in
        (select t.behaviour_oper_stage_id
        from behaviour_oper_stages t
        where t.stage_id = lv_stage_id 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.DESCRPTION LIKE lv_description_campaign));

    dbms_output.put_line('productId a eliminar: ' || lv_product_id);
    DELETE from BEHAVIOUR_PRODUCT t
    where t.behaviour_oper_stage_id in
        (select t.behaviour_oper_stage_id
        from behaviour_oper_stages t
        where t.stage_id = lv_stage_id 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.DESCRPTION LIKE lv_description_campaign));

    SELECT count(T.PRODUCT_TYPE_PROVISIONING_ID) INTO deleted_cant FROM
PRODUCT_TYPE_PROVISIONING T WHERE T.PRODUCT_TYPE IN (SELECT P.PRODUCT_TYPE_ID
FROM PRODUCTS P WHERE P.PRODUCT_ID = lv_product_id );

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

    -- DELETE PRODUCT_TYPE_PROVISIONING
    DELETE FROM PRODUCT_TYPE_PROVISIONING T WHERE T.PRODUCT_TYPE IN (SELECT
P.PRODUCT_TYPE_ID FROM PRODUCTS P WHERE P.PRODUCT_ID = lv_product_id );

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

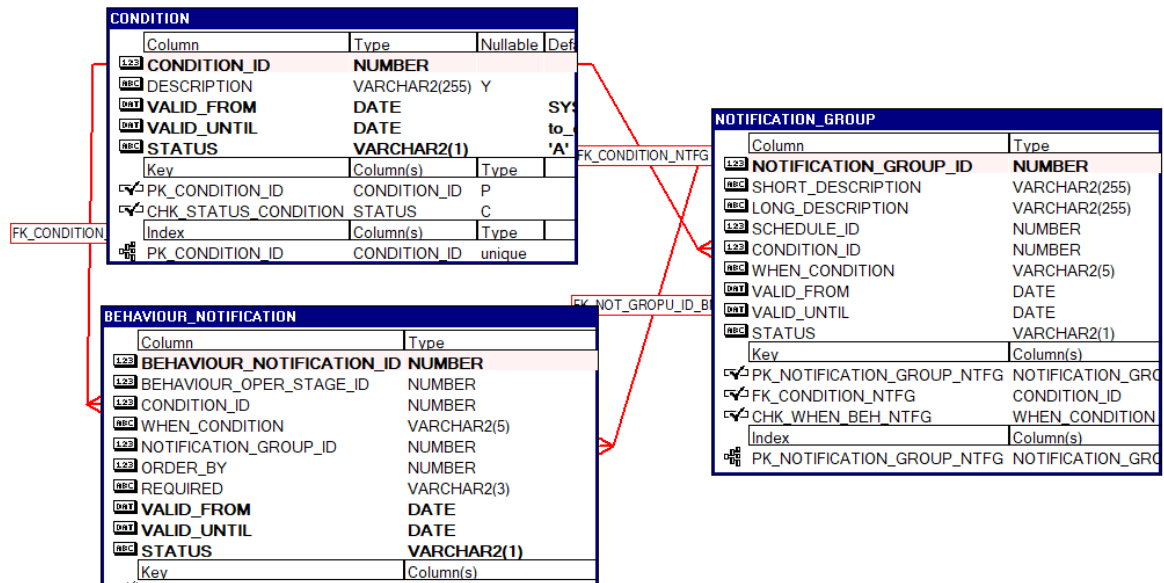
```

BEVAHIOUR_NOTIFICATION

El objetivo de usar esta configuración es poder notificar usando el componente de notificaciones.

El bloque de notificaciones permite llamar a una series de Grupos de notificaciones.

Los grupos de notificaciones incluyen tres tipos de llamados. WARNING, ERROR Y SUCCESS. De un grupo de Notificaciones, solo llamar a todos los que corresponden a WARNING o ERROR o SUCCESS.



QUICKWIN_MDS.NOTIFICATION_GROUP

NAME	TYPE	NULLABLE	DEFAULT
NOTIFICATION_GROUP_ID	NUMBER	N	
SHORT_DESCRIPTION	VARCHAR2(255)	Y	
LONG_DESCRIPTION	VARCHAR2(255)	Y	
SCHEDULE_ID	NUMBER	Y	
CONDITION_ID	NUMBER	Y	
WHEN_CONDITION	VARCHAR2(5)	Y	'TRUE'
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'

QUICKWIN_MDS.BEHAVIOUR_NOTIFICATION

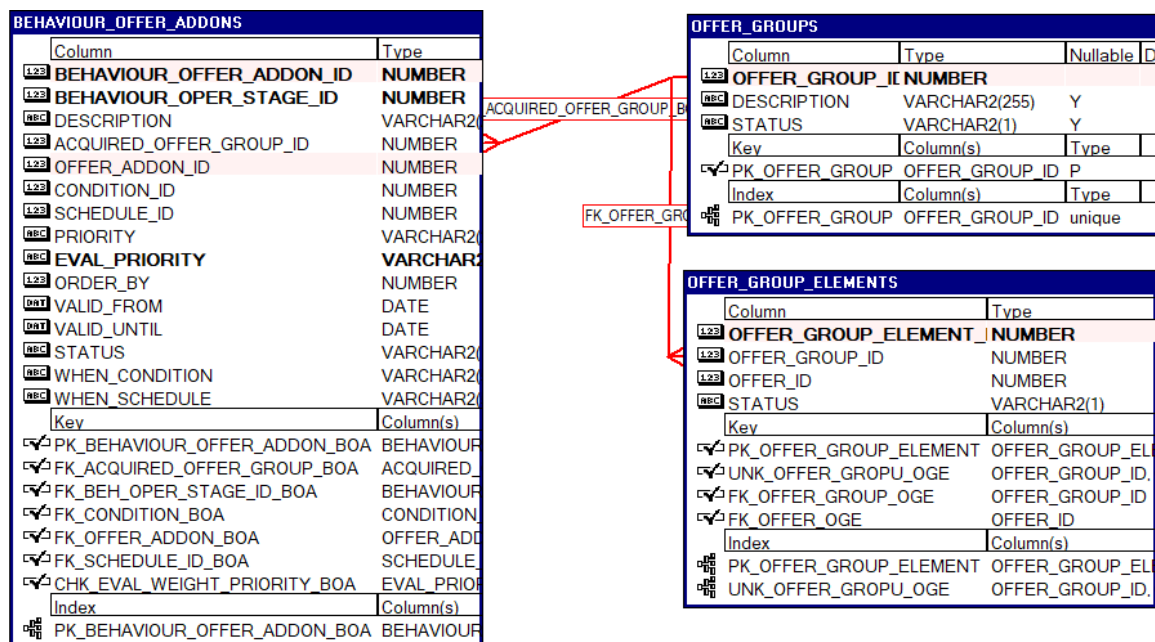
NAME	TYPE	NULLABLE	DEFAULT
BEHAVIOUR_NOTIFICATION_ID	NUMBER	N	
BEHAVIOUR_OPER_STAGE_ID	NUMBER	Y	
CONDITION_ID	NUMBER	Y	
WHEN_CONDITION	VARCHAR2(5)	Y	'TRUE'
NOTIFICATION_GROUP_ID	NUMBER	Y	
ORDER_BY	NUMBER	Y	10
REQUIRED	VARCHAR2(3)	Y	'YES'
VALID_FROM	DATE	N	SYSDATE
VALID_UNTIL	DATE	N	to_date('20/03/2987 23:59:59','DD/MM/YYYY HH24:MI:SS')
STATUS	VARCHAR2(1)	N	'A'

BEVAHIOUR_OFFER_ADDON

El objetivo de usar esta configuración es poder dar una oferta a un cliente siempre y cuando ese cliente tenga otra oferta existente. Las cuales se configuran en un grupo.

Un cliente puede aplicarmultiples ofertas dependiendo de una condicion, sin embargo cuando no se quiere dar todas las ofertas se puede especificar un prioridad en la configuracion. Todas las que deben evaluar prioridad deben estar marcadas como HIGH_PRIORITY en EVAL_PRIORITY y su peso de prioridad debe estar en PRIORITY donde se define un peso. Pero este tambien se puede enviar en un Subscriber Property dentro de un corchete. Si finalmente se quieren dar todas las ofertas sin evaluar prioridad en ese caso se debe configurar NONE en EVAL_PRIORITY y las ofertas se daran por condicion y schedule en en el orden especificado por ORDER_BY. Cuando se determina la oferta que se va a entregar, simplemente OFFER_ADDON invoca al feature de provisioning para aprovisione la oferta.

MODELO DE DATOS



Offer Addons espera que la oferta que se va a entregar **OFFER_ADDON_ID** este configurada en un grupo de ofertas para eso se define la siguiente tabla.

QUICKWIN_MDS.OFFER_GROUP_ELEMENTS

NAME	TYPE	NULLABLE	DEFAULT	COMMENTS
OFFER_GROUP_ELEMENT_ID	NUMBER	N		
OFFER_GROUP_ID	NUMBER	Y		Id de Grupo de oferta
OFFER_ID	NUMBER	Y		Id de Oferta
STATUS	VARCHAR2(1)	Y	'A'	Estado, Activo e Inactivo

La tabla de grupos es la siguiente.

QUICKWIN_MDS.OFFER_GROUPS

NAME	TYPE	NULLABLE	DEFAULT	COMMENTS
OFFER_GROUP_ID	NUMBER	N		
DESCRIPTION	VARCHAR2(255)	Y		Description
STATUS	VARCHAR2(1)	Y		Estado, Activo o Inactivo.

Las configuración **BEHAVIOUR_OFFER_ADDONS** especifican prioridad adicional a la evaluacion de Condition o Schedule. La columna EVAL_PRIORITY indica si tiene priorida NONE, ninguna o HIGH_PRIORITY. En el caso de aplicar dado las condiciones anteriores, se le otorga una nueva oferta al cliente **OFFER_ADDON_ID**.

NAME	TYPE	NULLABLE	DEFAULT	COMMENTS
BEHAVIOUR_OFFER_ADDON_ID	NUMBER	N		
BEHAVIOUR_OPER_STAGE_ID	NUMBER	N		Stage de Operacion
DESCRIPTION	VARCHAR2(255)	Y		Descripcion
ACQUIRED_OFFER_GROUP_ID	NUMBER	Y		Oferta adquirida
OFFER_ADDON_ID	NUMBER	Y		Oferta a entregar
CONDITION_ID	NUMBER	Y		Bajo que condicion se debe ejecutar
SCHEDULE_ID	NUMBER	Y		Bajo que schedule se debe ejecutar
PRIORITY	VARCHAR2(255)	Y	10	Peso debe ser numerico. Se puede configurar COMODIN subscriber_property entre []
EVAL_PRIORITY	VARCHAR2(255)	N	'NONE'	Que prioridad debe evaluar
ORDER_BY	NUMBER	Y	10	Aplica para cuando no hay evaluacion de pesos
VALID_FROM	DATE	Y	SYSDATE	Valido desde
VALID_UNTIL	DATE	Y	to_date('20/03/2987 23:59:59','DD/MM/YYYY HH24:MI:SS')	Valido hasta
STATUS	VARCHAR2(1)	Y	'A'	Estado Activo o Inactivo
WHEN_CONDITION	VARCHAR2(5)	Y		Si la condicion es TRUE, seguir, Si la condicion es FALSE seguir
WHEN_SCHEDULE	VARCHAR2(5)	Y		Si la condicion es TRUE, seguir, Si la condicion es FALSE seguir

EVAL_PRIORITY: HIGH_PRIORITY o NONE

Support

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

Stay in touch

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