SOLUCIONES

Best Technology for Best Business

# Engage® Business Solution 5

## Guía de trabajo con la base de datos

Engage® 5

Best in Business Technology

Tucumán 540, piso 9, oficina H, C1049AAL, C.A.B.A., Capital Federal, Argentina
www.soluciones-ar.com.ar / resur-info@soluciones-ar.com.ar
Tel. /Fax: (54-11) 4394-8575

SOLUCIONES
Best Technology for Best Business

# Contenido

SOLUCIONES
Best Technology for Best Business

## Acerca de Engage® Business Solution 5

**Engage® Business Solution 5** es una plataforma tecnológica orientada al diseño e implementación de procesos y aplicaciones de negocio de variada naturaleza.

Su organización modular permite instalar, configurar y distribuir sus componentes de muchas maneras posibles, posibilitando un escalamiento horizontal y vertical de la instalación.

El dimensionamiento y la configuración final de la plataforma dependerán del patrón de uso, de las aplicaciones que se ejecutarán sobre ella y del volumen de operación. Por lo tanto, la información contenida en el presente documento representa a una configuración normal, estándar y "de fábrica", la cual debe revisarse al momento de planear un entorno de producción específico.

## Alcance de este documento

Está comprendida la explicación de la creación y del despliegue básicos de la base de datos y/o el esquema de **Engage® Business Solution 5**, su usuario y los privilegios requeridos. No se incluye en este documento información técnica sobre el software o hardware de base (sistema operativo, bases de datos, equipamiento y/o infraestructura de comunicaciones).

La presente documentación puede aplicarse para realizar una instalación básica en ambientes de desarrollo, evaluación, capacitación o con fines de entrenamiento o demostración.

**Se recomienda aplicar todos los parches de Microsoft Update hasta la fecha de compilación de la versión de Engage que se instalará. Si posteriormente se va a actualizar Engage, actualice todos los parches de Microsoft Update hasta la nueva fecha de compilación de Engage.**

**En caso de que el cliente realice una actualización del sistema operativo por otros motivos, se requiere que el cliente realice pruebas funcionales y técnicas previas a la implementación de los cambios en producción.**

**Se requiere que las versiones de los módulos, de los servicios y de la base de datos de Engage instalados sean las mismas en cada ambiente.**

**Las instalaciones en producción se deben planificar en base a requerimientos dentro del marco y cronograma de un proyecto formal.**

Si tiene dudas o inquietudes acerca del presente documento, por favor, sírvase remitirlas a:
supportcrm@soluciones-ar.com.ar

## Caducidad de este documento

**Fecha de creación del documento: 20 de Octubre de 2014.**
**Fecha de caducidad del documento: 31 de Diciembre de 2014.**

Una vez que haya caducado el presente documento, remitirse al sitio o al correo de soporte oficiales para obtener una nueva versión del mismo.

https://soporte.engage5.com.ar/

SOLUCIONES
Best Technology for Best Business

# Introducción a la base de datos de Engage versión 4.12 o superior

A los efectos prácticos de este documento, asumiremos:

- Que el nombre de la base de datos de Engage es ENGAGEDEFAULT.
- Que el nombre del esquema que contiene el modelo de datos es ENGAGEDEFAULT.
- Que usted dispone de una base de datos de Engage creada en la versión 4.12 o superior.
- Que la existencia de los grupos de archivos mencionados en este documento sólo aplica para aquellos poseedores de una base de datos de Engage creada en la versión 4.12 o superior.

Engage requiere el uso de una cuenta de autenticación propia del motor de bases de datos, salvo algunas excepciones, por lo que, en Microsoft SQL Server se deberá habilitar el modo de autenticación mixto.

**El modelo de datos de Engage, contenido en su totalidad en el esquema ENGAGEDEFAULT de la base de datos de Engage, está formado por un conjunto de objetos estándares que no deben ser alterados o eliminados, dado que puede perjudicarse gravemente la calidad de los datos almacenados e impedirse el correcto funcionamiento del software.**

Cada modelo de negocios diseñado y desarrollado mediante Engage Desktop 5 Designer y/o otras herramientas CASE o de gestión de base de datos requerirá de un trabajo de relevamiento y análisis de las necesidades particulares del negocio, detectando ajustes que fueran precisos realizar como parte del proyecto de implementación de la solución.

Fundamentalmente, se deberán analizar a nivel técnico la creación y modificación de tablas y procesos ETL para integraciones con sistemas externos y de restricciones y validaciones de datos en todos los objetos creados como parte del modelo, para garantizar el correcto y esperado funcionamiento de la aplicación.

La metodología de trabajo adecuada y recomendada para el despliegue de un sistema requiere de 4 ambientes:

- Desarrollo.
- Evaluación.
- Preproducción.
- Producción.

El cliente deberá proveer y desplegar estos ambientes en tiempo y forma para la correcta operatoria durante el transcurso del proyecto y el posterior mantenimiento de la solución implementada.

Toda actividad de desarrollo y mantenimiento del modelo de datos de Engage debe llevarse a cabo en el ambiente de Desarrollo y sus resultados deberán trasladarse mediante scripts de SQL, DML y DDL al ambiente de Evaluación, los cuales serán probados para, luego, en caso de funcionar correctamente, ser ejecutados en los ambientes de Producción y Preproducción.

SOLUCIONES
Best Technology for Best Business

## Recomendaciones para almacenar bases de datos de producción

Para soportar bases de datos de producción de una solución informática OLTP se recomienda:

- Equipo SAN o similar con controladora RAID de, al menos, 512 MB de caché.
- Battery-Backed Write Cache.
- Asignar 50 % del caché para lectura y 50 % para escritura.
- Búfer de escritura de los discos habilitado.
- *Stripe size* de 8 KB para cada arreglo de los volúmenes lógicos.
- Crear una partición para cada volumen lógico. [(1)]
- Formatear los volúmenes lógicos usados para RDBMS con bloques de 8 KB. [(2)]
- En Windows, desactivar el servicio Index Server sobre volúmenes lógicos para RDBMS.
- Instalar solamente el software de RDBMS en el volumen lógico del SO.
- Utilizar reglas mnemotécnicas, tales como Oracle Optimal Flexibile Architecture (OFA), para nombrar los directorios y archivos de bases de datos de forma que sean fácilmente localizables.
- En Microsoft SQL Server, la base de datos TEMPDB debe tener tantos archivos de datos como núcleos de CPU haya para evitar la contención.
- Con Oracle Database Server 10g R2 o posterior, puede optar por utilizar *Automatic Storage Management (ASM)*.
- Ver las recomendaciones técnicas del fabricante para optimizar el almacenamiento de las bases de datos.

[(1)] En Windows, crear una partición para cada volumen lógico con DISKPART.EXE. Ejemplo:

```
LIST DISK
SELECT DISK 1
CREATE PARTITION PRIMARY ALIGN = 64
ASSIGN LETTER D
```

[(2)] En Windows, formatear la partición del volumen lógico con NTFS con bloques de 8 KB. Ejemplo:

```
FORMAT D: /FS:NTFS /A:8192 /V:D02P1_08K_ENGDF
```

SOLUCIONES
Best Technology for Best Business

# Creación de la base de datos de Desarrollo de Engage versión 4.12 o superior

Antes de crear la base de datos de Engage para Desarrollo, deberá asegurarse de que haya, al menos, 2 GB de espacio disponible en el volumen lógico donde se alojarán los grupos de archivos.

Los archivos de secuencias de comandos, scripts, de Engage para Microsoft SQL Server son:

- 01_MSSQL_ENGAGE_DATABASE_SCRIPT.SQL
- 02_MSSQL_ENGAGE_SYSTEM_VIEWS_SCRIPT.SQL
- 03_MSSQL_ENGAGE_TABLES_SCRIPT.SQL
- 04_MSSQL_ENGAGE_SYSTEM_BLOB_VIEWS_SCRIPT.SQL
- 05_MSSQL_ENGAGE_CONSTRAINTS_SCRIPT.SQL
- 06_MSSQL_ENGAGE_SEQUENCES_SCRIPT.SQL
- 07_MSSQL_ENGAGE_SOURCES_SCRIPT.SQL
- 08_MSSQL_ENGAGE_INITIAL_METADATA_SCRIPT.SQL
- 09_MSSQL_ENGAGE_COMPILE_INVALID_OBJECTS.SQL
- 10_MSSQL_ENGAGE_UPDATE_TABLE_STATISTICS_SCRIPT.SQL

El primer script se deberá correr desde SQL Server Management Studio, habiendo iniciado sesión en Microsoft SQL Server con la cuenta **sa**. Luego de correr el script, se deberá desconectar la sesión.

Dicho script permite:

- Crear la base de datos ENGAGEDEFAULT con sus grupos de archivos, filegroups.
- Crear la cuenta de inicio de sesión ENGAGEDEFAULT, cuyo password es @SAENG@2008.
- Establecer a ENGAGEDEFAULT como la base de datos predeterminada de la cuenta.
- Establecer al español como idioma predeterminado de la cuenta.
- Crear el esquema ENGAGEDEFAULT dentro de la base de datos.
- Crear el usuario ENGAGEDEFAULT dentro de la base de datos con permisos de propietario.
- Asociar la cuenta de inicio de sesión con el correspondiente usuario de la base de datos.
- Establecer el nivel de compatibilidad de la base de datos en 90 o 100, ofreciendo soporte a las características de base de datos de Microsoft SQL Server 2005, 2008 y 2008 R2, lo que no admite el uso de la sintaxis SQL no ANSI dentro de la misma, obligando al uso de LEFT OUTER JOIN y RIGHT OUTER JOIN en vez de *= y =*, respectivamente.
- Crear el usuario ENGAGEDEFAULT dentro de la base de datos MSDB con el privilegio de DatabaseMailUserRole asignado y acceso público sobre el esquema DBO.

El resto de los scripts se deberán correr en forma ordenada desde SQL Server Management Studio, habiendo iniciado sesión en Microsoft SQL Server con la cuenta ENGAGEDEFAULT y su password @SAENG@2008. Luego de correr los scripts, se deberá desconectar la sesión.

Dichos scripts crearán en el esquema ENGAGEDEFAULT los objetos de sistema de Engage:

- El segundo, las vistas de sistema.
- El tercero, las tablas de sistema.
- El cuarto, las vistas de sistema de objetos con columnas BLOB.
- El quinto, las restricciones e índices.
- El sexto, no deberá ejecutarse porque no contiene sentencias.
- El séptimo, los objetos de código fuente como funciones y procedimientos almacenados.
- El octavo, insertará los datos básicos del sistema.
- El noveno, no deberá ejecutarse porque no tiene sentencias.
- El décimo, actualizará las estadísticas de todas las tablas creadas por el segundo script.

La intercalación de lenguaje, **COLLATION**, predeterminada de la base de datos de Engage es **Modern_Spanish_CI_AS**. Si el servidor de bases de datos no tuviera la misma, se deberá modificar

el primer script y poner la que corresponda.

En el caso de Oracle, Engage requiere que previamente se cree una instancia de base de datos de tipo transaccional, de nombre ENGAGEDEFAULT, donde posteriormente se creará el esquema ENGAGEDEFAULT. Dicha instancia deberá brindar soporte al idioma español. En la mayoría de las instalaciones se requieren estos valores para compatibilidad con aplicaciones y servicios de Windows:

```
NLS_LANGUAGE = SPANISH
NLS_TERRITORY = SPAIN
CHARACTER_SET_NAME = WE8MSWIN1252
NLS_LANG = SPANISH_SPAIN.WE8MSWIN1252
```

Pero, pueden usar cualquier configuración de la base de datos de Oracle que soporte lenguaje español, la cual se define al momento de la creación, respetando el juego de caracteres.

Los archivos de secuencias de comandos, scripts, de Engage para Oracle son:

- 01_ORACLE_ENGAGE_SCHEMA_SCRIPT.SQL
- 02_ORACLE_ENGAGE_SYSTEM_VIEWS_SCRIPT.SQL
- 03_ORACLE_ENGAGE_TABLES_SCRIPT.SQL
- 04_ORACLE_ENGAGE_SYSTEM_BLOB_VIEWS_SCRIPT.SQL
- 05_ORACLE_ENGAGE_CONSTRAINTS_SCRIPT.SQL
- 06_ORACLE_ENGAGE_SEQUENCES_SCRIPT.SQL
- 07_ORACLE_ENGAGE_SOURCES_SCRIPT.SQL
- 08_ORACLE_ENGAGE_INITIAL_METADATA_SCRIPT.SQL
- 09_ORACLE_ENGAGE_COMPILE_INVALID_OBJECTS.SQL
- 10_ORACLE_ENGAGE_UPDATE_TABLE_STATISTICS_SCRIPT.SQL

El primer script se deberá correr desde TOAD, habiendo iniciado sesión en la instancia de base de datos de Oracle ENGAGEDEFAULT con la cuenta **sys**. Luego de correr el script, se deberá desconectar la sesión.

Dicho script permite:

- Crear el esquema ENGAGEDEFAULT con sus grupos de archivos, tablespaces.
- Crear el perfil de conexión ENGAGEDEFAULT_PROFILE con IDLE_TIME de 15 minutos.
- Crear el rol predeterminado ENGAGEDEFAULT_ROLE para el usuario ENGAGEDEFAULT.
- Asignar privilegios del sistema y del esquema para el rol predeterminado.
- Configurar el parámetro de UTL_FILE_DIR de la base de datos.
- Crear la cuenta de inicio de sesión ENGAGEDEFAULT, cuyo password es @SAENG@2008.
- Asociar a la cuenta de inicio de sesión con el correspondiente perfil de conexión.
- Establecer a ENGAGEDEFAULT_ROLE como el rol predeterminado de la cuenta.
- Asignar privilegios de sistema y cuotas en los tablespaces a la cuenta ENGAGEDEFAULT.

El resto de los scripts se deberán correr en forma ordenada desde TOAD, habiendo iniciado sesión en la instancia de base de datos de Oracle ENGAGEDEFAULT con la cuenta ENGAGEDEFAULT y su password @SAENG@2008. Luego de correr los scripts, se deberá desconectar la sesión.

Dichos scripts crearán en el esquema ENGAGEDEFAULT los objetos de sistema de Engage:

- El segundo, las vistas de sistema.
- El tercero, las tablas de sistema.
- El cuarto, las vistas de sistema de objetos con columnas BLOB.
- El quinto, las restricciones e índices.
- El sexto, las secuencias numéricas asignadas a algunas tablas de sistema.
- El séptimo, los objetos de código fuente como funciones y procedimientos almacenados.
- El octavo, insertará los datos básicos del sistema.
- El noveno, proveerá las sentencias necesarias para compilar los objetos inválidos.
- El décimo, actualizará las estadísticas de todas las tablas creadas por el segundo script.

Para cualquiera de los dos casos, los scripts están pensados para ejecutarse repetidas veces, por lo que, la primera vez que se corran van a dar errores las sentencias de DROP.

Una vez que se haya creado la base de datos de Engage, su esquema de trabajo y la cuenta de inicio de sesión, se deberán utilizar estos datos para editar los archivos de configuración de las aplicaciones y los servicios de Engage, permitiendo que se puedan conectar a la base de datos.

## Creación de la base de datos de Producción de Engage versión 4.12 o superior

El proceso de creación y despliegue de la base de datos de Producción de Engage forma parte del trabajo de planeamiento de la infraestructura, donde la adecuada instalación y configuración del servidor de bases de datos y de los dispositivos de almacenamiento son pilares fundamentales para el éxito de la solución implementada.

Dicho proceso se basa en la información actualizada de dimensionamiento, volumetría y proyección de crecimiento de la solución por el período mínimo de un año.

Las secuencias de comandos para la creación de la base de datos de Engage para Producción se modifican para cumplir con los requerimientos de almacenamiento, demanda de concurrencia y otros factores resultantes del trabajo de dimensionamiento de la infraestructura. Es decir que, la base de datos de Producción de Engage se crea a medida de las necesidades y recursos estimados, con suficiente espacio reservado para el período proyectado.

**Esta tarea debe ser solicitada al área de soporte de Engage Business Solution, de la misma forma que cuando se hace con el trabajo de dimensionamiento de la infraestructura.**

El conjunto de estructuras y datos probados y corregidos en la base de datos de Evaluación de Engage, diseñados en la base de datos de Desarrollo durante el transcurso del proyecto, deberá ser incorporado posteriormente a la base de datos de Producción mediante secuencias de comandos y procesos de migración de datos, que forman parte de este proceso de creación y despliegue.

Una vez concluido dicho proceso, la base de datos de Producción de Engage quedará lista para su evaluación y posterior puesta en funcionamiento.

Posteriormente, se tomará un respaldo de la base de datos de Producción de Engage y se reemplazará la base de datos de Preproducción con el mismo, de manera que las futuras pruebas de implementaciones o correcciones se realicen sobre una base de datos de iguales características.

SOLUCIONES
Best Technology for Best Business

## Hacer copias del login y de la base de datos de Engage

Para crear un nuevo inicio de sesión, login o cuenta de usuario o, hacer copias de la base de datos de Engage deberán iniciar sesión con **sa** en Microsoft SQL Server o con **sys** en Oracle, **no** con ENGAGEDEFAULT.

Cada vez que creen o restauren una copia de la base de datos ENGAGEDEFAULT, como por ejemplo, ENGAGEDEFAULTTEST, deberán crear, si no existe, un login del mismo nombre y enlazarlo al usuario ENGAGEDEFAULT de esta copia.

En Microsoft SQL Server, además, deberán realizar el siguiente procedimiento:

1. Iniciar sesión en SQL Server Management Studio con **sa**.
2. Abrir una consulta nueva sobre la base master y correr los siguientes comandos:

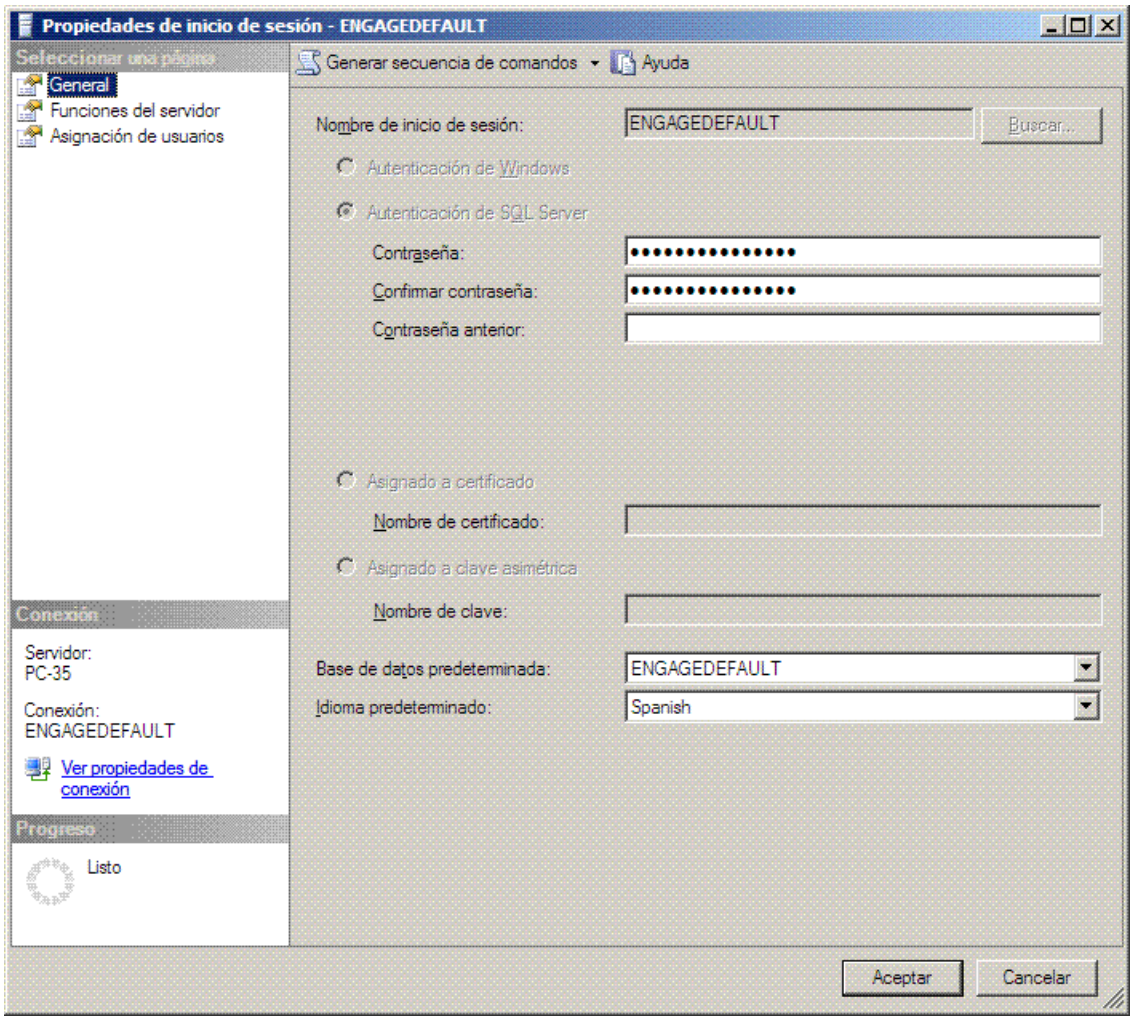
```
USE ENGAGEDEFAULTTEST;
GO
EXECUTE SP_CHANGE_USERS_LOGIN 'UPDATE_ONE', 'ENGAGEDEFAULT', 'ENGAGEDEFAULTTEST';
GO
```

3. Cerrar SQL Server Management Studio.
4. Abrir SQL Server Management Studio y conectarse con el nuevo login, ENGAGEDEFAULTTEST y su correspondiente password.

SOLUCIONES
Best Technology for Best Business

## Idioma predeterminado para el inicio de sesión de base de datos

Si utiliza Microsoft SQL Server como motor de base de datos, tenga en cuenta que deberá establecer como español el idioma predeterminado de la cuenta de inicio de sesión a la base de datos de Engage, como se ve en la figura de abajo.




En el caso de utilizar Oracle, se deberá agregar el soporte para lenguaje Español en la instancia de base de datos donde resida el esquema de Engage según se detalló anteriormente.

SOLUCIONES
Best Technology for Best Business

# Pasos para la actualización de versión de la base de datos de Engage

1. Hacer un respaldo completo de la base de datos de Engage.

2. Adaptar los scripts de actualización de Engage para incluir las modificaciones de los nombres de los grupos de archivos y de los procedimientos que referencian a los de sistema de Engage, según las instrucciones provistas dentro de los mismos.

2.1. Por ejemplo, con un editor de texto avanzado, abrir todos los scripts de actualización de versión y reemplazar “ENGAGEDEFAULT” por el nombre de su base de datos o esquema.

2.2. Las restricciones no son opcionales, sino obligatorias porque garantizan que los procedimientos funcionen correctamente, manteniendo la calidad de los datos y meta datos.

2.3. Engage 4.12.3 y versiones superiores requieren que se cumplan todas las pautas sobre el modelo de datos contenidas en este documento.

3. Ejecutar los scripts en el orden ascendente determinado por el nombre de los archivos, sea con SQL Server Management Studio o con TOAD for Oracle.

3.1. Los scripts no están probados con Oracle SQL*Plus y la presencia de barras al final de las sentencias podría duplicar su ejecución. También, podrían ocurrir problemas con el código fuente de los procedimientos almacenados debido a la presencia de caracteres especiales. En caso de no contar con TOAD for Oracle, quitar las barras donde no correspondan y utilizar Oracle SQL*Plus con las siguientes sentencias:

```
SET TIMING ON;
SET SERVEROUTPUT ON;
SET LINESIZE 32767;
SET SCAN OFF;
SET DEFINE OFF;
BEGIN
      SYS.DBMS_OUTPUT.DISABLE;
      SYS.DBMS_OUTPUT.ENABLE(1000000);
END;
/
```

3.2. Si hubiera algún error en la corrida, deberá editar los scripts con las sentencias necesarias para solucionar el inconveniente que pudiera producirse con los datos espurios.

3.3. Volver a ejecutar los scripts desde el primer punto en que habían cancelado.

4. Una vez que haya actualizado el esquema, deberá compilar todos los objetos inválidos, desfragmentarlo, compactarlo y actualizar las estadísticas completamente (COMPUTE).

SOLUCIONES
Best Technology for Best Business

## Pasos para la migración de meta datos de la base de datos de Engage

Los desarrollos de tipos de clientes, procesos, entidades, categorías de datos, transacciones y demás objetos diseñados en Engage Desktop 5 Designer persisten mediante meta datos en tablas de la base de datos de Engage.

El proceso de traslado desde el ambiente de desarrollo (origen) hacia el ambiente de evaluación (destino) debe realizarse mediante scripts de DML y DDL, los cuales serán probados para, luego, en caso de funcionar correctamente, ser ejecutados en otros ambientes.

Engage Desktop 5 Meta Data Manager permite generar scripts de inserción y de borrado de meta datos de un tipo de cliente, de un proceso o de una categoría de datos seleccionada. El procedimiento de generación de scripts con este módulo es el siguiente:

1. Seleccionar un archivo XML de estructura de tipo de cliente, proceso o categoría de datos que anteriormente haya exportado desde Engage Desktop 5 Designer.
2. Conectarse a la base de datos de destino y generar los scripts DML de borrado de cada tipo de cliente, proceso o categoría de datos, si es que va a actualizar meta datos ya existentes.
3. Conectarse a la base de datos de origen y generar los scripts DML de inserción de cada tipo de cliente, proceso o categoría de datos que desee migrar, verificando que tengan atomicidad.

Luego, se deben identificar y catalogar todas las tablas, secuencias, vistas, funciones, triggers, paquetes y procedimientos almacenados que se hayan creado para soportar al tipo de cliente o proceso que se desea migrar. Para ello, se debe utilizar SQL Server Management Studio o TOAD for Oracle, según sea el motor de base de datos utilizado. Las siguientes consultas sirven para determinar qué objetos son nuevos y cuáles han sido modificados en su esquema de Engage de desarrollo. Funcionan en ORACLE 9, 10 y 11, así como también en MSSQL 2005 y 2008.

```
-- IDENTIFICAR LOS OBJETOS NUEVOS POR LA FECHA DE CREACIÓN MÁS RECIENTE.
SELECT * FROM EVI_OBJECTS WHERE (CREATION_DATE = LAST_DDL_TIME) ORDER BY LAST_DDL_TIME DESC;

-- IDENTIFICAR LOS OBJETOS MODIFICADOS POR LA FECHA DE MODIFICACIÓN MÁS RECIENTE.
SELECT * FROM EVI_OBJECTS WHERE (CREATION_DATE <> LAST_DDL_TIME) ORDER BY LAST_DDL_TIME DESC;
```

Luego de catalogar los objetos, se deben generar con SQL Server Management Studio o TOAD for Oracle los siguientes scripts DDL en el esquema de origen:

1. Si son objetos nuevos, scripts con las sentencias de creación de cada uno de ellos ordenados por dependencia.
2. Si son objetos modificados, scripts con las sentencias de modificación y otros con sentencias que deshagan esas modificaciones, si correspondiera.
3. Dichos scripts se deben nombrar en forma fechada y por orden de dependencia dentro de una carpeta que indique su versión y fecha.

**Es importante utilizar una metodología mnemotécnica con carpetas y fechas para guardar los scripts de las distintas versiones de un tipo de cliente, proceso o categoría de datos, de forma que se pueda realizar un seguimiento de los cambios y permitir una vuelta atrás en caso de que fuera necesario.**

Luego, con los scripts DML y DDL terminados, se debe conectar al ambiente de destino con SQL Server Management Studio o TOAD for Oracle y ejecutarlos en forma ordenada.

Una vez que haya actualizado el esquema de destino, deberá compilar todos los objetos inválidos, desfragmentarlo, compactarlo y actualizar las estadísticas completamente (COMPUTE).

SOLUCIONES
Best Technology for Best Business

# Requisitos de diseño de paquetes y procedimientos almacenados

**Todos los paquetes y procedimientos almacenados deben realizar una transacción atómica y poseer control de errores estándar para salir por COMMIT o ROLLBACK según corresponda.**

Todos deben poseer un encabezado donde consten autor, editor y fechas de creación y de edición. Además, cuando dichos procedimientos se utilicen para transacciones de procesos de Engage, deben programarse para que su ejecución individual no exceda los 4 segundos.

Ejemplo de control de errores y atomicidad para un procedimiento de MSSQL:

```
CREATE PROCEDURE PA_SYS_DEPURAR_TABLAS_LOG (@PDFECHA_DESDE DATETIME = NULL, @PDFECHA_HASTA DATETIME = NULL, @PS_RET_MESSAGE
VARCHAR(440) = NULL OUTPUT) AS

-- ============================================================================================
-- AUTOR: TITO - ENGAGE SOFTWARE COMPANY
-- FECHA DE CREACIÓN: 25/06/2008
-- EDITOR: TITO - ENGAGE SOFTWARE COMPANY
-- FECHA DE MODIFICACIÓN: 19/03/2014
-- TABLAS LEÍDAS: DSS_TBPHYSICAL_CALL_CONV_STEPS, PHYSICAL_CALL_CONV_STEPS, PHYSICAL_CALL_TRANSACTIONS, ETC.
-- TABLAS MODIFICADAS: DSS_TBPHYSICAL_CALL_CONV_STEPS, PHYSICAL_CALL_CONV_STEPS, PHYSICAL_CALL_TRANSACTIONS, ETC.
-- DESCRIPCIÓN: PROCEDIMIENTO QUE DEPURA LOS REGISTROS SEGÚN EL RANGO DE FECHAS ENVIADO POR PARÁMETROS.
-- CONSIDERACIONES:
-- 1. EL PARÁMETRO @PDFECHA_DESDE DETERMINA DESDE QUÉ FECHA TOMAR EL RANGO.
-- 2. EL PARÁMETRO @PDFECHA_HASTA DETERMINA HASTA QUÉ FECHA TOMAR EL RANGO.
-- 3. EL PARÁMETRO DE SALIDA @PS_RET_MESSAGE DEVUELVE EL MENSAJE DE ERROR. NULL SI NO HUBO.
-- ============================================================================================

SET NOCOUNT ON;
SET XACT_ABORT ON;
SET DATEFORMAT YMD;

-- DECLARACIÓN E INICIALIZACIÓN DE VARIABLES DE ERROR Y AUXILIARES.
DECLARE @LNERRORNUM AS INTEGER,
        @LNSEVERITY AS SMALLINT,
        @LNFILAS_AFEC AS INTEGER,
        @LDINICIO_OPE AS DATETIME,
        @LNDURACION AS INTEGER,
        @LSOPERA AS VARCHAR(30),
        @LSPROC AS VARCHAR(30),
        @LSTABLA AS VARCHAR(30),
        @LSERRORLOG AS VARCHAR(440),
        @LSERRORDESC AS VARCHAR(440),
        @LSSQL_CMD AS VARCHAR(4000)
;
SELECT  @LNERRORNUM = 0,
        @LNSEVERITY = 16,
        @LNFILAS_AFEC = 0,
        @LDINICIO_OPE = GETDATE(),
        @LNDURACION = 0,
        @LSOPERA = 'EXECUTE',
        @LSPROC = OBJECT_NAME(@@PROCID),
        @LSTABLA = 'NINGUNA',
        @LSERRORLOG = @@SERVERNAME + '.' + DB_NAME() + '.' + USER_NAME() + '.' + OBJECT_NAME(@@PROCID) + ' - ',
        @LSERRORDESC = NULL,
        @LSSQL_CMD = NULL
;

-- DECLARA LAS VARIABLES.
DECLARE @LDFECHA_DESDE DATETIME,
        @LDFECHA_HASTA DATETIME
;
-- DETERMINA LAS FECHAS DE PROCESO. SÓLO FECHA SIN HORA.
SELECT @LDFECHA_DESDE = CASE WHEN @PDFECHA_DESDE IS NULL THEN CONVERT(DATETIME, '19000101', 112) ELSE CONVERT(VARCHAR,
@PDFECHA_DESDE, 112) END;
SELECT @LDFECHA_HASTA = CASE WHEN @PDFECHA_HASTA IS NULL THEN CONVERT(VARCHAR, GETDATE(), 112) ELSE CONVERT(VARCHAR,
@PDFECHA_HASTA, 112) END;

-- DECLARO LA TRANSACCIÓN.
BEGIN TRANSACTION

-- DEPURA LA TABLA DSS_TBPHYSICAL_CALL_CONV_STEPS POR EL CAMPO TS_BEGIN.
DELETE
FROM    DSS_TBPHYSICAL_CALL_CONV_STEPS
WHERE   TS_BEGIN BETWEEN @LDFECHA_DESDE AND @LDFECHA_HASTA;
SELECT @LSOPERA = 'DELETE', @LSTABLA = 'DSS_TBPHYSICAL_CALL_CONV_STEPS', @LNFILAS_AFEC = @@ROWCOUNT, @LNERRORNUM = @@ERROR;
IF (@LNERRORNUM <> 0) GOTO FALLO;

-- RESTO DE LAS SENTENCIAS.

FALLO:
-- REGISTRO DE SUCESOS Y CONTROL DE FIN DE PROCEDIMIENTO. CONCLUYE LA TRANSACCIÓN SI TODO FUE EXITOSO.
SELECT @LNDURACION = DATEDIFF(SECOND, @LDINICIO_OPE, GETDATE()), @LNSEVERITY = ISNULL(EM.SEVERITY, 16), @LSERRORDESC = 'ERROR
' + CONVERT(VARCHAR, ISNULL(EM.ERROR, 0)) + ' - ' + ISNULL(EM.DESCRIPTION, ' ') FROM MASTER.DBO.SYSMESSAGES EM WHERE
(EM.ERROR = @LNERRORNUM);
IF (@LNERRORNUM <> 0) BEGIN
        -- NOTIFICACIÓN EN EL REGISTRO DE SQL SERVER.
        SET @LSERRORLOG = @LSERRORLOG + CASE WHEN (ISNULL(CHARINDEX('%', @LSERRORDESC), 0) > 0) THEN RTRIM(LEFT(@LSERRORDESC,
11)) ELSE @LSERRORDESC END;
        -- DEVUELVE EL ERROR EN EL PARÁMETRO DE SALIDA.
        SET @PS_RET_MESSAGE = SUBSTRING(@LSERRORDESC, 1, 440);
```

```
        -- DESHACE LA TRANSACCIÓN COMPLETA SI LA OPERACIÓN FALLÓ.
        IF (@@TRANCOUNT > 0) ROLLBACK TRANSACTION;
        EXECUTE  @LNERRORNUM  =  PA_SYS_REGISTRO_SUCESOS_INSERT  @PSPROCEDIMIENTO_1  =  @LSPROC,  @PSTABLA_2  =  @LSTABLA,
@PNFILAS_AFECTADAS_3 = @LNFILAS_AFEC, @PSOPERACION_4 = @LSOPERA, @PNDURACION_5 = @LNDURACION, @PSRESULTADO_6 = @LSERRORDESC,
@PSSQL_CMD_7 = @LSERRORDESC;
END
ELSE BEGIN
        EXECUTE  @LNERRORNUM  =  PA_SYS_REGISTRO_SUCESOS_INSERT  @PSPROCEDIMIENTO_1  =  @LSPROC,  @PSTABLA_2  =  @LSTABLA,
@PNFILAS_AFECTADAS_3 = @LNFILAS_AFEC, @PSOPERACION_4 = @LSOPERA, @PNDURACION_5 = @LNDURACION, @PSRESULTADO_6 = @LSERRORDESC,
@PSSQL_CMD_7 = @LSERRORDESC;
        IF (@@TRANCOUNT > 0) COMMIT TRANSACTION;
END;
RETURN;
GO
```

Ejemplo de control de errores y atomicidad para un procedimiento de Oracle:

```
CREATE PROCEDURE PA_SYS_DEPURAR_TABLAS_LOG (PDFECHA_DESDE IN DATE := NULL, PDFECHA_HASTA IN DATE := NULL, PS_RET_MESSAGE OUT
VARCHAR2) AS

-- ================================================================================================================================
-- AUTOR: TITO - ENGAGE SOFTWARE COMPANY
-- FECHA DE CREACIÓN: 25/06/2008
-- EDITOR: TITO - ENGAGE SOFTWARE COMPANY
-- FECHA DE MODIFICACIÓN: 19/03/2014
-- TABLAS LEÍDAS: DSS_TBPHYSICAL_CALL_CONV_STEPS, PHYSICAL_CALL_CONV_STEPS, PHYSICAL_CALL_TRANSACTIONS, ETC.
-- TABLAS MODIFICADAS: DSS_TBPHYSICAL_CALL_CONV_STEPS, PHYSICAL_CALL_CONV_STEPS, PHYSICAL_CALL_TRANSACTIONS, ETC.
-- DESCRIPCIÓN: PROCEDIMIENTO QUE DEPURA LOS REGISTROS SEGÚN EL RANGO DE FECHAS ENVIADO POR PARÁMETROS.
-- CONSIDERACIONES:
-- 1. EL PARÁMETRO PDFECHA_DESDE DETERMINA DESDE QUÉ FECHA TOMAR EL RANGO.
-- 2. EL PARÁMETRO PDFECHA_HASTA DETERMINA HASTA QUÉ FECHA TOMAR EL RANGO.
-- 3. EL PARÁMETRO DE SALIDA PS_RET_MESSAGE DEVUELVE EL MENSAJE DE ERROR. NULL SI NO HUBO.
-- ================================================================================================================================

-- DECLARACIÓN E INICIALIZACIÓN DE VARIABLES DE ERROR Y AUXILIARES.
LNERRORNUM NUMBER := 0;
LNSEVERITY NUMBER(9) := 16;
LNFILAS_AFEC NUMBER(9) := 0;
LDINICIO_OPE DATE := SYSDATE;
LNDURACION NUMBER(9) := 0;
LSOPERA VARCHAR2(30) := 'EXECUTE';
LSPROC VARCHAR2(30) := 'PA_SYS_DEPURAR_TABLAS_LOG';
LSTABLA VARCHAR2(30) := 'NINGUNA';
LSERRORLOG VARCHAR2(440) := SYS_CONTEXT('USERENV', 'DB_NAME') || '.' || SYS_CONTEXT('USERENV', 'SESSION_USER') || '.' ||
'PA_SYS_DEPURAR_TABLAS_LOG' || ' - ';
LSERRORDESC VARCHAR2(440) := NULL;
LSSQL_CMD VARCHAR2(4000) := NULL;

-- DECLARA LAS VARIABLES.
LDFECHA_DESDE DATE;
LDFECHA_HASTA DATE;

-- DECLARO LA TRANSACCIÓN.
BEGIN

-- DETERMINA LAS FECHAS DE PROCESO. SÓLO FECHA SIN HORA.
IF (PDFECHA_DESDE IS NULL) THEN
        LDFECHA_DESDE := TO_DATE('19000101', 'YYYYMMDD');
ELSE
        LDFECHA_DESDE := TO_DATE(PDFECHA_DESDE);
END IF;
IF (PDFECHA_HASTA IS NULL) THEN
        LDFECHA_HASTA := TO_DATE(SYSDATE);
ELSE
        LDFECHA_HASTA := TO_DATE(PDFECHA_HASTA);
END IF;

-- DEPURA LA TABLA DSS_TBPHYSICAL_CALL_CONV_STEPS POR EL CAMPO TS_BEGIN.
SELECT 'DELETE', 'DSS_TBPHYSICAL_CALL_CONV_STEPS' INTO LSOPERA, LSTABLA FROM DUAL;
DELETE
FROM    DSS_TBPHYSICAL_CALL_CONV_STEPS
WHERE   TS_BEGIN BETWEEN LDFECHA_DESDE AND LDFECHA_HASTA;
LNFILAS_AFEC := LNFILAS_AFEC + SQL%ROWCOUNT;
LNDURACION := (SYSDATE - LDINICIO_OPE) * 86400;

-- RESTO DE LAS SENTENCIAS.

-- REGISTRO DE SUCESOS Y CONTROL DE FIN DE PROCEDIMIENTO. CONCLUYE LA TRANSACCIÓN SI TODO FUE EXITOSO.
PA_SYS_REGISTRO_SUCESOS_INSERT  (PSPROCEDIMIENTO_1  =>  LSPROC,  PSTABLA_2  =>  LSTABLA,  PNFILAS_AFECTADAS_3  =>  LNFILAS_AFEC,
PSOPERACION_4 => LSOPERA, PNDURACION_5 => LNDURACION, PSRESULTADO_6 => LSERRORDESC, PSSQL_CMD_7 => LSERRORDESC);
COMMIT;

EXCEPTION
WHEN OTHERS THEN
        SELECT 0, (SYSDATE - LDINICIO_OPE) * 86400, 16 INTO LNFILAS_AFEC, LNDURACION, LNSEVERITY FROM DUAL;
        LNERRORNUM := SQLCODE;
        LSERRORDESC := 'ERROR ' || TO_CHAR(LNERRORNUM) || ' - ' || SQLERRM(LNERRORNUM);
        -- NOTIFICACIÓN DE ERROR.
        LSERRORLOG := LSERRORLOG || LSERRORDESC;
        -- DEVUELVE EL ERROR EN EL PARÁMETRO DE SALIDA.
        PS_RET_MESSAGE := SUBSTR(LSERRORDESC, 1, 440);
        -- DESHACE LA TRANSACCIÓN COMPLETA SI LA OPERACIÓN FALLÓ.
        ROLLBACK;
        PA_SYS_REGISTRO_SUCESOS_INSERT  (PSPROCEDIMIENTO_1  =>  LSPROC,  PSTABLA_2  =>  LSTABLA,  PNFILAS_AFECTADAS_3  =>
LNFILAS_AFEC,  PSOPERACION_4  =>  LSOPERA,  PNDURACION_5  =>  LNDURACION,  PSRESULTADO_6  =>  LSERRORDESC,  PSSQL_CMD_7  =>
LSERRORDESC);
        COMMIT;

END PA_SYS_DEPURAR_TABLAS_LOG;
/
```

SOLUCIONES
Best Technology for Best Business

# Programación de tareas de depuración y mantenimiento de tablas de Engage

1. Incluir en un JOB o tarea programada del motor de base de datos, que corra antes del ETL nocturno de Engage Analytical Solution, los siguientes procesos de mantenimiento:

1.1. La depuración diaria de las tablas de LOG de Engage para los datos de más de 31 días de antigüedad. El siguiente código da soporte para realizar dicha tarea:

Para SQL Server:

```
DECLARE
@LS_RET_MESSAGE VARCHAR(440),
@LD_FECHA_HASTA DATETIME;
SELECT @LS_RET_MESSAGE = NULL, @LD_FECHA_HASTA = CONVERT(VARCHAR, GETDATE() - 31, 112);
EXECUTE PA_SYS_DEPURAR_TABLAS_LOG @PDFECHA_DESDE = NULL, @PDFECHA_HASTA = @LD_FECHA_HASTA,
@PS_RET_MESSAGE = @LS_RET_MESSAGE OUTPUT;
PRINT @LS_RET_MESSAGE;
GO
```

Para Oracle:

```
DECLARE
LS_RET_MESSAGE VARCHAR2(440) := NULL;
LD_FECHA_HASTA DATE := (TRUNC(SYSDATE) - 31);
BEGIN
PA_SYS_DEPURAR_TABLAS_LOG (PDFECHA_DESDE => NULL, PDFECHA_HASTA => LD_FECHA_HASTA,
PS_RET_MESSAGE => LS_RET_MESSAGE);
PRINT (LS_RET_MESSAGE);
COMMIT;
END;
/
```

1.2. La depuración diaria de las tablas de gestión de Engage para los datos de más de 731 días de antigüedad, cuya información ya ha sido consolidada en el modelo analítico. El siguiente código da soporte para realizar dicha tarea:

Para SQL Server:

```
DECLARE
@LS_RET_MESSAGE VARCHAR(440),
@LD_FECHA_HASTA DATETIME;
SELECT @LS_RET_MESSAGE = NULL, @LD_FECHA_HASTA = CONVERT(VARCHAR, GETDATE() - 731, 112);
EXECUTE PA_SYS_DEPURAR_TABLAS_WKFD @PDFECHA_DESDE = NULL, @PDFECHA_HASTA = @LD_FECHA_HASTA,
@PS_RET_MESSAGE = @LS_RET_MESSAGE OUTPUT;
PRINT @LS_RET_MESSAGE;
GO
```

Para Oracle:

```
DECLARE
LS_RET_MESSAGE VARCHAR2(440) := NULL;
LD_FECHA_HASTA DATE := (TRUNC(SYSDATE) - 731);
BEGIN
PA_SYS_DEPURAR_TABLAS_WKFD (PDFECHA_DESDE => NULL, PDFECHA_HASTA => LD_FECHA_HASTA,
PS_RET_MESSAGE => LS_RET_MESSAGE);
PRINT (LS_RET_MESSAGE);
COMMIT;
END;
/
```

1.3. El procedimiento de bloqueo y/o baja lógica de las cuentas de usuarios de Engage según parámetros de la tabla PRM cargados desde Engage Desktop 5 Security Administrator:

Para SQL Server:

```
DECLARE
@LS_RET_MESSAGE VARCHAR(440);
SELECT @LS_RET_MESSAGE = NULL;
EXECUTE PA_SYS_UPDATE_USER_STATUS @PS_RET_MESSAGE = @LS_RET_MESSAGE OUTPUT;
PRINT @LS_RET_MESSAGE;
GO
```

Para Oracle:

```
DECLARE
LS_RET_MESSAGE VARCHAR2(440) := NULL;
BEGIN
PA_SYS_UPDATE_USER_STATUS (PS_RET_MESSAGE => LS_RET_MESSAGE);
PRINT (LS_RET_MESSAGE);
COMMIT;
END;
/
```

2. Cada proceso ETL debería incluir sentencias de actualización de estadísticas para aquellas tablas que afecta.

## Plan recomendado de mantenimiento diario de la base de datos

Programar un plan de mantenimiento para que, luego de que hayan finalizado todos los procesos ETL nocturnos, realice las siguientes tareas sobre la base de datos de Engage:

1. Depurar archivos de corridas anteriores del plan de mantenimiento.
2. Depurar respaldos obsoletos de la base de datos.
3. Crear un respaldo completo de la base de datos.
4. Reconstruir los índices de la base de datos.
5. Verificar la integridad de la base de datos.
6. Actualizar las estadísticas de la base de datos (COMPUTE).

**Nota: el plan podría incluir como primeros pasos la ejecución de los procesos ETL nocturnos.**

SOLUCIONES
Best Technology for Best Business

# Integraciones de tipo BATCH de sistemas externos con Engage

La mejor forma de hacer las integraciones de datos imprescindibles para Engage, como los clientes y sus datos relacionados, entre un sistema externo que funcione en un gestor de bases de datos y la base de datos de Engage suele ser mediante el uso de un servidor vinculado en el servidor de Engage y la creación de un conjunto de tablas de intercambio de datos en el grupo de archivos ENGAGEDEFAULT_ETL, que cumplan el rol de **Staging Area** y reciban las novedades provenientes del sistema externo, el cual también debe contar con su propia **Staging Area** con tablas y vistas que reúnan la información requerida ya validada, de forma que los procesos ETL solamente deban hacer un traslado de los registros de novedades sin especificar condiciones de filtro y volcarlos en las tablas de la **Staging Area** de Engage.

De esta forma, en un corto plazo de tiempo se pueden realizar procesos ETL basados en procedimientos almacenados que pueden leer directamente sobre las tablas y vistas de la **Staging Area** remota e insertar novedades en las tablas temporales, con prefijo TMP_, en la **Staging Area** de la base de datos de Engage.

La lógica de obtención y validación de los datos en el repositorio de origen y los permisos necesarios para accederlos constituyen una tarea imprescindible que el cliente debe realizar o, en caso contrario, no se podrán desarrollar los procesos ETL para las integraciones con Engage.

Todas las configuraciones de red, de servicios, de servidores vinculados, de DSN de ODBC, de controladores ODBC de terceros, de cuentas de acceso y de cualquier otro recurso necesario para que puedan realizarse integraciones con sistemas externos deben ser provistos por el cliente.

Un consultor de Engage es quien debe modelar y crear las tablas temporales que servirán de **Staging Area** en la propia base de datos para que se pueblen con los procesos ETL.

Personal del cliente debe modelar y crear las tablas temporales y vistas que servirán de **Staging Area** en las bases de datos externas desde donde se tomarán los datos con los procesos ETL.

Una vez que las novedades hayan sido validadas por unicidad, existencia, obligatoriedad, integridad referencial y de dominio durante el proceso ETL que carga las tablas temporales, las mismas podrán ser leídas desde las tablas temporales y volcadas en las tablas del modelo de negocios de Engage mediante procedimientos almacenados ETL más simples que, también, cuenten con validaciones de unicidad, existencia, obligatoriedad, integridad referencial y de dominio.

Para poder realizar correctamente las validaciones en los procesos ETL, las tablas temporales, con prefijo TMP_ y las tablas del modelo de negocios de Engage, con prefijos CUSTOMER_ y TMT_, deberán contar con las restricciones que sean necesarias descriptas en la sección de **Restricciones a crear y respetar en el modelo de Engage**, así como, también, contar con los valores obligatorios definidos en aquellas columnas que lo requieran.

Es importante realizar esta tarea en conjunto con la creación y actualización de la documentación del modelo de datos del negocio, para poder comenzar, luego, con el desarrollo de los procesos ETL y que los mismos puedan funcionar adecuadamente.

SOLUCIONES
Best Technology for Best Business

## Detalle de los grupos de archivos de la base de datos de Engage

Los grupos de archivos pueden identificarse como **filegroups** o **tablespaces** según se trate de una base de datos de Microsoft SQL Server u Oracle, respectivamente.

Para **trabajar** en la base de datos de Engage, ENGAGEDEFAULT, deberá usarse **siempre** el login ENGAGEDEFAULT, cuyo password original es @SAENG@2008, para asegurarse que todos los objetos pertenezcan al esquema adecuado, que posee el mismo nombre que la base de datos. La siguiente línea de ejemplo identifica una tabla perteneciente al esquema de trabajo de la base de datos de Engage:

```
SERVIDOR.ENGAGEDEFAULT.ENGAGEDEFAULT.CUSTOMER
```

La base de datos de Engage tiene muchos **grupos de archivos** pero, **solamente** deberán crearse objetos en algunos de ellos, a saber:

- **ENGAGEDEFAULT_DATA**: Es el predeterminado. Aquí, Engage Desktop 5 Designer creará las tablas que se modelen y es donde el consultor deberá crear, mediante scripts DDL, las restricciones PRIMARY KEY o UNIQUE que sean CLUSTERED, esto último sólo para Microsoft SQL Server, ya que en el caso de Oracle no se deben crear restricciones o índices aquí.

- **ENGAGEDEFAULT_INDX**: Aquí es donde el consultor deberá crear, mediante scripts DDL, todas las restricciones e índices NONCLUSTERED de las tablas creadas por Engage Desktop 5 Designer. Además, en Oracle, deberá crear las restricciones PRIMARY KEY o UNIQUE.

- **ENGAGEDEFAULT_ETL**: Aquí es donde el consultor deberá crear, mediante scripts DDL, todas las tablas temporales, de prueba o de log así como, también, las restricciones PRIMARY KEY o UNIQUE o índices CLUSTERED de las mismas, esto último sólo para Microsoft SQL Server, ya que en el caso de Oracle no se deben crear restricciones o índices aquí.

- **ENGAGEDEFAULT_ETL_INDX**: Aquí es donde el consultor deberá crear, mediante scripts DDL, todas las restricciones e índices NONCLUSTERED de las tablas creadas a mano, sean temporales, de prueba o de log. En el caso de Oracle, además deberá crear las restricciones PRIMARY KEY o UNIQUE.

- **RESTO DE LOS GRUPOS DE ARCHIVOS**: **no se deben crear o modificar objetos allí** bajo ninguna circunstancia. En caso de haber creado por error algún objeto, el mismo deberá eliminarse y crearse donde corresponda.

SOLUCIONES
Best Technology for Best Business

## Restricciones a crear y respetar en el modelo de Engage

En el ambiente de desarrollo, durante el proceso de modelado y antes de hacer un pase al ambiente de evaluación, para garantizar la calidad e integridad de los datos de las tablas modeladas con Engage Desktop 5 Designer y de las creadas manualmente, el consultor deberá encarar un proceso de relevamiento de la claves del negocio de dichas tablas para desarrollar un script DDL que las aplique y tenga en cuenta las siguientes reglas:

1. Todos los objetos que se vayan a crear deberán tener un **nombre en letras MAYÚSCULAS que no supere los 30 caracteres de longitud**. Utilizar la siguiente consulta para revisar los objetos:

```
-- OBJETOS CUYO NOMBRE TIENE MÁS DE 30 CARACTERES.
SELECT      OBJECT_TYPE,
      OBJECT_NAME,
      NAME_LENGTH
FROM  EVI_OBJECTS
WHERE (NAME_LENGTH > 30)
ORDER BY OBJECT_TYPE ASC,
      OBJECT_NAME ASC;
```

2. Todos los objetos que se vayan a crear deberán tener un **nombre único e irrepetible**. Utilizar la siguiente consulta para revisar los objetos:

```
-- OBJETOS CON NOMBRE DUPLICADO.
SELECT      OBJECT_NAME
FROM  EVI_OBJECTS
WHERE (OBJECT_TYPE <> 'PACKAGE BODY')
GROUP BY OBJECT_NAME
HAVING      (COUNT(1) > 1)
ORDER BY OBJECT_NAME ASC;
```

3. Cuando se creen o modifiquen tablas, se deberá evitar que la suma de los bytes de sus columnas exceda los 8000 bytes. Utilizar la siguiente consulta para revisar las tablas:

```
-- TABLAS CUYA DEFINICIÓN TIENE UNA CANTIDAD DE BYTES MAYOR A 8000.
SELECT      TABLE_NAME,
      SUM(BYTE_LENGTH) AS BYTE_LENGTH
FROM  EVI_TABLE_COLUMNS
WHERE DATA_TYPE_NAME NOT IN ('BLOB', 'IMAGE')
      AND TABLE_NAME NOT IN ('PHYSICAL_CALL_TRANSACTIONS')
GROUP BY TABLE_NAME
HAVING      (SUM(BYTE_LENGTH) > 8000)
ORDER BY TABLE_NAME ASC;
```

3.1. Con esta consulta podrá revisar aquellas entidades o tablas lógicas donde la suma de los bytes de sus columnas excede los 8000 bytes y, luego, corregirlas desde Engage Desktop 5 Designer.

```
-- TABLAS DENTRO DE ENGAGE CUYA DEFINICIÓN TIENE UNA CANTIDAD DE BYTES MAYOR A 8000.
SELECT      EB.ENTITY_TABLE_REL,
      SUM(EBD.FIELD_LENGTH) AS ROW_LENGTH
FROM  ENTITY_BROWSER EB
      INNER JOIN ENTITY_BROWSER_DETAIL EBD ON (EB.PKEY = EBD.PAR_KEY)
GROUP BY EB.ENTITY_TABLE_REL
HAVING      (SUM(EBD.FIELD_LENGTH) > 8000)
ORDER BY 1 ASC;
```

4. Cuando se cree o modifique una columna de tabla, se deberá evitar que su tamaño en bytes exceda los 4000 bytes, teniendo en cuenta siempre el cumplimiento del punto 3. Utilizar la siguiente consulta para revisar las columnas de las tablas:

```
-- COLUMNAS DE TABLAS CUYA CANTIDAD DE BYTES ES MAYOR A 4000.
SELECT      TABLE_NAME,
      COLUMN_NAME,
```

```
      BYTE_LENGTH
FROM  EVI_TABLE_COLUMNS
WHERE (BYTE_LENGTH > 4000)
ORDER BY TABLE_NAME ASC,
      COLUMN_NAME ASC;
```

4.1. Con esta consulta podrá revisar aquellas columnas de entidades o tablas lógicas que tengan más de 4000 bytes y, luego, corregirlas desde Engage Desktop 5 Designer.

```
-- COLUMNAS DE TABLAS DENTRO DE ENGAGE CUYA CANTIDAD DE BYTES ES MAYOR A 4000.
SELECT      EB.ENTITY_TABLE_REL,
      EBD.FIELD_NAME,
      EBD.FIELD_LENGTH
FROM  ENTITY_BROWSER EB
      INNER JOIN ENTITY_BROWSER_DETAIL EBD ON (EB.PKEY = EBD.PAR_KEY)
WHERE (EBD.FIELD_LENGTH > 4000)
ORDER BY 1 ASC, 2 ASC;
```

5. Cuando se creen o modifiquen restricciones o índices, se deberá evitar que la suma de los bytes de sus columnas exceda los 900 bytes. Utilizar la siguiente consulta para revisar los índices:

```
-- ÍNDICES CUYA CANTIDAD DE BYTES ES MAYOR A 900.
SELECT      TABLE_NAME,
      INDEX_NAME,
      SUM(BYTE_LENGTH) AS BYTE_LENGTH
FROM  EVI_TABLE_COLUMN_INDEXES
GROUP BY TABLE_NAME,
      INDEX_NAME
HAVING      (SUM(BYTE_LENGTH) > 900)
ORDER BY TABLE_NAME ASC,
      INDEX_NAME ASC;
```

6. Cuando se cree una tabla con una columna auto numerada o asociada a una secuencia numérica, la misma deberá tener el tipo de datos numérico de 15 posiciones sin decimales. La secuencia numérica asociada deberá tener la misma precisión que la columna. Ejemplos:

```
-- MSSQL
CREATE TABLE TABLE1 (PKEY NUMERIC(15, 0) IDENTITY (1, 1) NOT FOR REPLICATION NOT NULL);
GO

-- ORACLE
CREATE TABLE TABLE1 (PKEY NUMBER(15, 0) NOT NULL);
CREATE SEQUENCE SEQ_TABLE1 INCREMENT BY 1 START WITH 1 MINVALUE 1 MAXVALUE 999999999999999
NOCYCLE NOCACHE ORDER;
```

Para obtener las columnas auto numeradas o con secuencias numéricas que no tienen el tipo de datos y la precisión requerida, ejecute la siguiente consulta:

```
-- COLUMNAS DE TABLAS CON SECUENCIAS NUMÉRICAS QUE NO SON DE 15 DÍGITOS.
SELECT      TABLE_NAME,
      COLUMN_NAME,
      SEQUENCE_NAME,
      MAX(MAX_VALUE) AS MAX_VALUE
FROM  EVI_SEQUENCES
GROUP BY TABLE_NAME,
      COLUMN_NAME,
      SEQUENCE_NAME
HAVING      (MAX(MAX_VALUE) > 999999999999999)
      OR (MIN(MAX_VALUE) < 999999999999999)
ORDER BY 1 ASC, 2 ASC, 3 ASC;
```

7. Cuando se desee utilizar una columna con clave auto generada, por ejemplo, al crear una nueva tabla en forma manual, optar por el uso de valores GUID y no de secuencias numéricas. Ejemplos:

```
-- MSSQL
CREATE TABLE TABLE1 (PKEY VARCHAR(36) NOT NULL);
GO
ALTER TABLE TABLE1 ADD CONSTRAINT DF_TABLE1_PKEY DEFAULT NEWID() FOR PKEY;
GO
```

```
-- ORACLE
CREATE TABLE TABLE1 (PKEY VARCHAR2(36) NOT NULL);
ALTER TABLE TABLE1 MODIFY PKEY VARCHAR2(36) DEFAULT (SYS_GUID());
```

8. Las claves primarias (PRIMARY KEY) deberán tener un prefijo **PK_** seguido por el nombre de tabla, apocopado si es necesario y, luego, separado por un carácter _ del nombre, convenientemente apocopado, de las columnas que la componen, siempre respetando las reglas 1 y 2. Los caracteres correspondientes al nombre de la tabla no pueden ser más de 21.

9. Las claves alternativas (UNIQUE) deberán tener un prefijo **UQ_** seguido por el nombre de tabla, apocopado si es necesario y, luego, separado por un carácter _ del nombre, convenientemente apocopado, de las columnas que la componen, siempre respetando las reglas 1 y 2. Los caracteres correspondientes al nombre de la tabla no pueden ser más de 21.

10. Los índices deberán tener un prefijo **IX_** seguido por el nombre de tabla, apocopado si es necesario y, luego, separado por un carácter _ del nombre, convenientemente apocopado, de las columnas que la componen, siempre respetando las reglas 1 y 2. Los caracteres correspondientes al nombre de la tabla no pueden ser más de 21.

11. Todas las tablas creadas manualmente, temporales o de interfaces, deben tener, al menos, una clave primaria (PRIMARY KEY) del negocio constituida por una o más columnas que determinen la unicidad de sus registros. Dicha clave primaria deberá constituirse como CLUSTERED. Ejemplo:

```
FROM  INTERFACES_TARJETAS_AUR ITA,
      TMT_A14
WHERE (ITA.NROTJTITULAR = TMT_A14.TJ_NRO_PROD_COMPLETO)
```

En el ejemplo, ITA.NROTJTITULAR debería ser PRIMARY KEY (CLUSTERED para SQL Server).
TMT_A14.TJ_NRO_PROD_COMPLETO debería ser UNIQUE (CLUSTERED para SQL Server) junto con TMT_A14.PAR_KEY, suponiendo que la tabla TMT_A14 tiene una cardinalidad de 1 a N con PHYSICAL_JOB.
O, TMT_A14.TJ_NRO_PROD_COMPLETO debería ser UNIQUE (NONCLUSTERED para SQL Server), suponiendo que la tabla TMT_A14 tiene una cardinalidad de 1 a 1 con PHYSICAL_JOB.

```
-- EJEMPLO DE CREACIÓN Y ELIMINACIÓN DE RESTRICCIONES DE UNICIDAD E ÍNDICES EN ORACLE.

ALTER TABLE INTERFACES_TARJETAS_AUR ADD CONSTRAINT PK_INTERFACES_TARJETAS_AUR_NTT PRIMARY KEY
(NROTJTITULAR) USING INDEX TABLESPACE ENGAGEDEFAULT_ETL_INDX;

ALTER TABLE INTERFACES_TARJETAS_AUR DROP CONSTRAINT PK_INTERFACES_TARJETAS_AUR_NTT;

ALTER TABLE TMT_A14 ADD CONSTRAINT PK_TMT_A14_PKEY PRIMARY KEY (PKEY) USING INDEX TABLESPACE
ENGAGEDEFAULT_INDX;

ALTER TABLE TMT_A14 DROP CONSTRAINT PK_TMT_A14_PKEY;

ALTER TABLE TMT_A14 ADD CONSTRAINT UQ_TMT_A14_PAR_KEY_TNPC UNIQUE (PAR_KEY,
TJ_NRO_PROD_COMPLETO) USING INDEX TABLESPACE ENGAGEDEFAULT_INDX;

ALTER TABLE TMT_A14 DROP CONSTRAINT UQ_TMT_A14_PAR_KEY_TNPC;

CREATE INDEX IX_TMT_A14_TS_BEGIN ON TMT_A14 (TS_BEGIN) TABLESPACE ENGAGEDEFAULT_INDX;

DROP INDEX IX_TMT_A14_TS_BEGIN;


-- EJEMPLO DE CREACIÓN DE RESTRICCIONES DE UNICIDAD E ÍNDICES EN SQL SERVER.

ALTER TABLE INTERFACES_TARJETAS_AUR ADD CONSTRAINT PK_INTERFACES_TARJETAS_AUR_NTT PRIMARY KEY
CLUSTERED (NROTJTITULAR) ON ENGAGEDEFAULT_ETL;
GO

ALTER TABLE INTERFACES_TARJETAS_AUR DROP CONSTRAINT PK_INTERFACES_TARJETAS_AUR_NTT;
GO
```

```
ALTER TABLE TMT_A14 ADD CONSTRAINT PK_TMT_A14_PKEY PRIMARY KEY NONCLUSTERED (PKEY) ON
ENGAGEDEFAULT_INDX;
GO

ALTER TABLE TMT_A14 DROP CONSTRAINT PK_TMT_A14_PKEY;
GO

ALTER TABLE TMT_A14 ADD CONSTRAINT UQ_TMT_A14_PAR_KEY_TNPC UNIQUE CLUSTERED (PAR_KEY,
TJ_NRO_PROD_COMPLETO) ON ENGAGEDEFAULT_DATA;
GO

ALTER TABLE TMT_A14 DROP CONSTRAINT UQ_TMT_A14_PAR_KEY_TNPC;
GO

CREATE NONCLUSTERED INDEX IX_TMT_A14_TS_BEGIN ON TMT_A14 (TS_BEGIN) ON ENGAGEDEFAULT_INDX;
GO

DROP INDEX TMT_A14.IX_TMT_A14_TS_BEGIN;
GO
```

12. Además, dichas tablas podrían tener otras claves, en este caso UNIQUE (NONCLUSTERED para SQL Server) pero, también, deberán tener índices (NONCLUSTERED para SQL Server) para todos aquellos grupos de columnas que sean consultadas en los **WHERE** de los procedimientos almacenados. Es decir, no un índice por cada columna, sino, índices por varias columnas que formen parte de los filtros usados en el **WHERE**. Un caso concreto donde se requiere un índice es para toda aquella columna que actué como clave foránea y no forme parte de una restricción u otro índice.

13. Todas las **entidades** diseñadas en Engage Desktop 5 Designer que tengan una **cardinalidad de 1 a N** deben tener, al menos, **2 claves y 1 o más índices**.
13.1. PRIMARY KEY (NONCLUSTERED para SQL Server) por columna PKEY.
13.2. UNIQUE (CLUSTERED para SQL Server) por columnas PAR_KEY y columna(s) clave del negocio.
13.3. (NONCLUSTERED para SQL Server) INDEX por grupos de columnas o columnas individuales que sean consultadas en los **WHERE** de procedimientos o vistas desarrollados.
13.4. Las columnas **PKEY**, **PAR_KEY**, **TS_BEGIN**, **TS_END, TS_CREATED_USER_ID** y **TS_USER_ID** junto con la(s) columna(s) que forman la clave del negocio deben tener la restricción **NOT NULL**, sin excepción.

14. Todas las **entidades** diseñadas en Engage Desktop 5 Designer que tengan una **cardinalidad de 1 a 1** deben tener, al menos, **3 claves y 1 o más índices**.
14.1. PRIMARY KEY (NONCLUSTERED para SQL Server) por columna PKEY.
14.2. UNIQUE (CLUSTERED para SQL Server) por columna PAR_KEY.
14.3. UNIQUE (NONCLUSTERED para SQL Server) por columna(s) clave(s) del negocio.
14.4. (NONCLUSTERED para SQL Server) INDEX por grupos de columnas o columnas individuales que sean consultadas en los **WHERE** de procedimientos o vistas desarrollados.
14.5. Las columnas **PKEY**, **PAR_KEY**, **TS_BEGIN**, **TS_END, TS_CREATED_USER_ID** y **TS_USER_ID** junto con la(s) columna(s) que forman la clave del negocio deben tener la restricción **NOT NULL**, sin excepción.

15. Las columnas **PKEY**, **TS_BEGIN**, **TS_END, TS_CREATED_USER_ID** y **TS_USER_ID** deben tener una restricción DEFAULT con el prefijo **DF_** seguido por el nombre de tabla, apocopado si es necesario y, luego, separado por un carácter **_** del nombre, convenientemente apocopado, de la columna, siempre respetando las reglas 1 y 2. Los caracteres correspondientes al nombre de la tabla no pueden ser más de 21. Los valores predeterminados para cada columna deben ser, NEWID() o (SYS_GUID()) para **PKEY**, GETDATE() o (SYSDATE) para **TS_BEGIN** y **TS_END** y, 'SYSTEM' para **TS_CREATED_USER_ID** y **TS_USER_ID**, para SQL Server y Oracle, respectivamente.

16. Los procedimientos de tipo **BATCH**, no deben tener sentencias de tipo PRINT o, SELECT con el mismo fin, dentro de ciclos del código, ya que generan uso innecesario del búfer.

17. Todos los procedimientos desarrollados deberán insertar obligatoriamente los valores de las columnas **PKEY**, **PAR_KEY**, **TS_BEGIN**, **TS_END**, **TS_CREATED_USER_ID** y **TS_USER_ID**, así como, también, en la(s) columna(s) que forman la clave del negocio.

18. Las referencias a datos de clientes desde la tabla PHYSICAL_JOB deben realizarse haciendo junta con la tabla CUSTOMER por PHYSICAL_JOB.PKEY_CUSTOMER y CUSTOMER.PKEY. Luego, desde allí, se podrá hacer junta con otras tablas mediante CUSTOMER.CUST_TYPE_CODE y CUSTOMER.CUST_PRIMARY_ID o CUSTOMER.CUST_SECONDARY_ID. Ejemplo:

```
-- FILTRO INCORRECTO.

FROM  PHYSICAL_JOB PJ
WHERE PJ.CUST_PRIMARY_ID IN (SELECT CUST_PRIMARY_ID FROM TMP_PHYSICAL_JOB);

-- FILTRO CORRECTO. SERÍA IDEAL QUE LA TABLA TMP PRESENTARA INFORMACIÓN POR TIPO DE CLIENTE
PARA EVITAR EL USO DE UN VALOR LITERAL.

FROM  PHYSICAL_JOB PJ
      INNER JOIN CUSTOMER T1 ON (PJ.PKEY_CUSTOMER = T1.PKEY)
WHERE EXISTS (
       SELECT    1
       FROM TMP_PHYSICAL_JOB T2
       WHERE     ('TIPO_CLIENTE' = T1.CUST_TYPE_CODE)
              AND (T2.CUST_PRIMARY_ID = T2.CUST_PRIMARY_ID);
```

SOLUCIONES
Best Technology for Best Business

# Consultas útiles para analizar el modelo de datos

```
-- CADA REGISTRO QUE DEVUELVA CADA UNA DE LAS SIGUIENTES CONSULTAS REPRESENTA UNA TAREA DE CORRECCIÓN DEL MODELO DE DATOS QUE HAY QUE REALIZAR, EXCEPTO QUE SE ACLARE LO CONTRARIO.
-- ANTES DE EJECUTAR LAS CONSULTAS HAY QUE REEMPLAZAR EL NOMBRE DE LOS TABLESPACES ENGAGEDEFAULT_XXXX POR LOS CORRESPONDIENTES A SU BASE DE DATOS DE ENGAGE.

-- OBJETOS CUYO NOMBRE TIENE MÁS DE 30 CARACTERES.
SELECT      OBJECT_TYPE,
      OBJECT_NAME,
      NAME_LENGTH
FROM  EVI_OBJECTS
WHERE (NAME_LENGTH > 30)
ORDER BY OBJECT_TYPE ASC,
      OBJECT_NAME ASC;

-- OBJETOS CON NOMBRE DUPLICADO.
SELECT      OBJECT_NAME
FROM  EVI_OBJECTS
WHERE (OBJECT_TYPE <> 'PACKAGE BODY')
GROUP BY OBJECT_NAME
HAVING      (COUNT(1) > 1)
ORDER BY OBJECT_NAME ASC;

-- TABLAS CUYA DEFINICIÓN TIENE UNA CANTIDAD DE BYTES MAYOR A 8000.
SELECT      TABLE_NAME,
      SUM(BYTE_LENGTH) AS BYTE_LENGTH
FROM  EVI_TABLE_COLUMNS
WHERE DATA_TYPE_NAME NOT IN ('BLOB', 'IMAGE')
      AND TABLE_NAME NOT IN ('PHYSICAL_CALL_TRANSACTIONS')
GROUP BY TABLE_NAME
HAVING      (SUM(BYTE_LENGTH) > 8000)
ORDER BY TABLE_NAME ASC;

-- TABLAS DENTRO DE ENGAGE CUYA DEFINICIÓN TIENE UNA CANTIDAD DE BYTES MAYOR A 8000.
SELECT      EB.ENTITY_TABLE_REL,
      SUM(EBD.FIELD_LENGTH) AS ROW_LENGTH
FROM  ENTITY_BROWSER EB
      INNER JOIN ENTITY_BROWSER_DETAIL EBD ON (EB.PKEY = EBD.PAR_KEY)
GROUP BY EB.ENTITY_TABLE_REL
HAVING      (SUM(EBD.FIELD_LENGTH) > 8000)
ORDER BY 1 ASC;

-- CAMPOS DE TABLAS CUYA CANTIDAD DE BYTES ES MAYOR A 4000.
SELECT      TABLE_NAME,
      COLUMN_NAME,
      BYTE_LENGTH
FROM  EVI_TABLE_COLUMNS
WHERE (BYTE_LENGTH > 4000)
ORDER BY TABLE_NAME ASC,
      COLUMN_NAME ASC;

-- CAMPOS DE TABLAS DENTRO DE ENGAGE CUYA CANTIDAD DE BYTES ES MAYOR A 4000.
SELECT      EB.ENTITY_TABLE_REL,
      EBD.FIELD_NAME,
      EBD.FIELD_LENGTH
FROM  ENTITY_BROWSER EB
      INNER JOIN ENTITY_BROWSER_DETAIL EBD ON (EB.PKEY = EBD.PAR_KEY)
WHERE (EBD.FIELD_LENGTH > 4000)
ORDER BY 1 ASC, 2 ASC;

-- ÍNDICES CUYA CANTIDAD DE BYTES ES MAYOR A 900.
SELECT      TABLE_NAME,
      INDEX_NAME,
      SUM(BYTE_LENGTH) AS BYTE_LENGTH
FROM  EVI_TABLE_COLUMN_INDEXES
GROUP BY TABLE_NAME,
      INDEX_NAME
HAVING      (SUM(BYTE_LENGTH) > 900)
ORDER BY TABLE_NAME ASC,
      INDEX_NAME ASC;

-- COLUMNAS DE TABLAS CON SECUENCIAS NUMÉRICAS QUE NO SON DE 15 DÍGITOS.
SELECT      TABLE_NAME,
      COLUMN_NAME,
```

```
      SEQUENCE_NAME,
      MAX(MAX_VALUE) AS MAX_VALUE
FROM  EVI_SEQUENCES
GROUP BY TABLE_NAME,
      COLUMN_NAME,
      SEQUENCE_NAME
HAVING      (MAX(MAX_VALUE) > 999999999999999)
      OR (MIN(MAX_VALUE) < 999999999999999)
ORDER BY 1 ASC, 2 ASC, 3 ASC;

-- TABLAS QUE EXISTEN EN ENGAGE DESIGNER Y QUE RESIDEN EN EL TABLESPACE EQUIVOCADO.
-- DEBEN MOVERSE A ENGAGEDEFAULT_DATA.
SELECT      *
FROM  EVI_TABLES
WHERE TABLE_NAME IN (SELECT EB.ENTITY_TABLE_REL FROM ENTITY_BROWSER EB)
      AND (TABLESPACE_NAME <> 'ENGAGEDEFAULT_DATA')
      AND TABLE_NAME NOT IN ('EIS_RESPONSE')
ORDER BY 1 ASC, 2 ASC, 3 ASC;

-- TABLAS QUE NO EXISTEN EN ENGAGE DESIGNER Y QUE RESIDEN EN EL TABLESPACE EQUIVOCADO.
-- DEBEN MOVERSE A ENGAGEDEFAULT_ETL.
SELECT      *
FROM  EVI_TABLES
WHERE TABLE_NAME NOT IN (SELECT EB.ENTITY_TABLE_REL FROM ENTITY_BROWSER EB)
      AND TABLE_NAME NOT IN (
      'CUSTOMER',
      'CUSTOMER_ACCOUNTS',
      'CUSTOMER_CONTACTS',
      'CUSTOMER_DATA',
      'CUSTOMER_ENVIOS',
      'CUSTOMER_OPPORTUNITIES',
      'CUSTOMER_PAYMENTS',
      'TMT_COLLECTION')
      AND TABLE_NAME NOT LIKE 'ELS_%'
      AND TABLESPACE_NAME IN ('ENGAGEDEFAULT_DATA', 'ENGAGEDEFAULT_INDX')
ORDER BY 1 ASC, 2 ASC, 3 ASC;

-- LOS TABLESPACES DE LAS TABLAS NO DEBEN TENER EL SUFIJO _INDX O SER PRIMARY.
SELECT      *
FROM  EVI_TABLES
WHERE TABLESPACE_NAME NOT IN (
      'ENGAGEDEFAULT_COLL',
      'ENGAGEDEFAULT_DATA',
      'ENGAGEDEFAULT_DSST',
      'ENGAGEDEFAULT_EB',
      'ENGAGEDEFAULT_ETL',
      'ENGAGEDEFAULT_MD',
      'ENGAGEDEFAULT_SD',
      'ENGAGEDEFAULT_WKFD')
ORDER BY 1 ASC, 2 ASC, 3 ASC;

-- LOS TABLESPACES DE LOS ÍNDICES CLUSTERED DE LAS TABLAS NO DEBEN TENER EL SUFIJO _INDX O SER
PRIMARY.
SELECT      *
FROM  EVI_TABLE_INDEXES
WHERE (INDEX_TYPE = 'CLUSTERED' OR INDEX_TYPE = 'IOT - TOP')
      AND TABLESPACE_NAME NOT IN (
      'ENGAGEDEFAULT_COLL',
      'ENGAGEDEFAULT_DATA',
      'ENGAGEDEFAULT_DSST',
      'ENGAGEDEFAULT_EB',
      'ENGAGEDEFAULT_ETL',
      'ENGAGEDEFAULT_MD',
      'ENGAGEDEFAULT_SD',
      'ENGAGEDEFAULT_WKFD')
ORDER BY 1 ASC, 2 ASC, 3 ASC, 4 ASC;

-- LOS TABLESPACES DE LOS ÍNDICES NONCLUSTERED DE LAS TABLAS DEBEN TENER EL SUFIJO _INDX.
SELECT      *
FROM  EVI_TABLE_INDEXES
WHERE (INDEX_TYPE = 'NONCLUSTERED' OR INDEX_TYPE = 'NORMAL')
      AND TABLESPACE_NAME NOT IN (
      'ENGAGEDEFAULT_COLL_INDX',
      'ENGAGEDEFAULT_DSST_INDX',
      'ENGAGEDEFAULT_EB_INDX',
      'ENGAGEDEFAULT_ETL_INDX',
      'ENGAGEDEFAULT_INDX',
      'ENGAGEDEFAULT_MD_INDX',
```

```
      'ENGAGEDEFAULT_SD_INDX',
      'ENGAGEDEFAULT_WKFD_INDX')
ORDER BY 1 ASC, 2 ASC, 3 ASC, 4 ASC;

-- TABLAS QUE NO SON DE SISTEMA Y ESTÁN EN EL TABLESPACE 'ENGAGEDEFAULT_DSST'.
SELECT      *
FROM  EVI_TABLES
WHERE (TABLESPACE_NAME = 'ENGAGEDEFAULT_DSST')
      AND TABLE_NAME NOT IN (
      'DSS_TBATTENDED_CUSTOMER',
      'DSS_TBPHYS_CALL_ANSWERED_QUES',
      'DSS_TBPHYSICAL_CALL',
      'DSS_TBPHYSICAL_CALL_CONV_STEPS',
      'DSS_TBPHYSICAL_JOB')
ORDER BY 1 ASC, 2 ASC, 3 ASC;

-- TABLAS QUE NO SON DE SISTEMA Y ESTÁN EN EL TABLESPACE 'ENGAGEDEFAULT_MD'.
SELECT      *
FROM  EVI_TABLES
WHERE (TABLESPACE_NAME = 'ENGAGEDEFAULT_MD')
      AND TABLE_NAME NOT IN (
      'ACTIVITY_TYPE',
      'ALARM',
      'ALARM_CHECKS',
      'ALARM_POPUP_DETAIL',
      'ALARM_TRANSACTIONS',
      'ANSWER',
      'CALL_CONV_STRUCTURES',
      'CALL_RESULT_TYPE',
      'CALL_TYPE',
      'CALL_TYPE_EMAIL',
      'CAT_DATA',
      'CAT_TYPE',
      'CAT_TYPE_DATA_REL',
      'CAT_TYPE_GROUP',
      'CHAT_MESSAGE',
      'CHAT_STRUCTURES',
      'COD_GEN',
      'CONV_STRUCTURE_STEP_NORMAL',
      'CONV_STRUCTURE_STEP_QUESTION',
      'CONV_STRUCTURE_STEP_TRAN',
      'CUSTOMER_DOCUMENTS',
      'CUSTOMER_DOCUMENTS_ATT',
      'CUSTOMER_TYPE',
      'CUSTOMER_TYPE_DATA_REL',
      'CUST_TYPE_ENT_KEY_DET',
      'DATA_TYPE',
      'DETAIL_FIELD',
      'DETAIL_FILE',
      'DETAIL_LABEL',
      'DETAIL_SPREAD',
      'DNIS',
      'DNIS_CALL_TYPE',
      'EMAILIN_MESSAGE',
      'EMAILIN_STRUCTURES',
      'ENTITY_BROWSER',
      'ENTITY_BROWSER_DETAIL',
      'ENTITY_BROWSER_SUGER',
      'ENTITY_VIRTUAL_DETAIL',
      'JOB_ACTIONS',
      'JOB_ATTACHED_DATA',
      'JOB_DIAGRAM',
      'JOB_DIAGRAM_MULT_INFLOWS_RULES',
      'JOB_FLOW',
      'JOB_NEXTATT_RULES',
      'JOB_NOTIF_BY_EVENT',
      'JOB_NOTIF_BY_RULE',
      'JOB_RESULT_TYPE',
      'JOB_TYPE',
      'JOB_TYPE_DATA_REL',
      'LINK',
      'LINK_PRM',
      'NOT_CONTACT_REASONS',
      'POPUP_BROWSER',
      'POPUP_BROWSER_DETAIL',
      'PRM',
      'QUESTION',
      'SMS_MESSAGE',
```

```
      'SP_PRM_IN',
      'SP_PRM_OUT',
      'SP_PRM_OUT_VALUE',
      'SQL_SELECTS',
      'STEP_CHECKS',
      'STEP_CHECKS_SP',
      'STEP_NORMAL_FLOW',
      'STEP_QUESTION_FLOW',
      'STEP_RESULTS',
      'STEP_TRANSACTION_FLOW',
      'STORED_PROCEDURES',
      'TEXT_MESSAGE',
      'TRAN_PRM_IN',
      'TRAN_PRM_OUT',
      'TRAN_PRM_VALUE',
      'TRANSACTIONS')
ORDER BY 1 ASC, 2 ASC, 3 ASC;

-- TABLAS QUE NO SON DE SISTEMA Y ESTÁN EN EL TABLESPACE 'ENGAGEDEFAULT_SD'.
SELECT      *
FROM  EVI_TABLES
WHERE (TABLESPACE_NAME = 'ENGAGEDEFAULT_SD')
      AND TABLE_NAME NOT IN (
      'AGENTE',
      'AUDITOR',
      'CAMPAIGN',
      'CAMPAIGN_GROUP',
      'CAMPAIGN_PROGRAM',
      'COLL_ACCION',
      'COLL_ACE',
      'COLL_ACE_PLAZOS',
      'COLL_ACE_RP',
      'COLL_ACE_SP',
      'COLL_ACUERDOS',
      'COLL_ACUERDOS_DET',
      'COLL_AUTORIZACIONES',
      'COLL_CAMBIOS_ACTIVIDAD',
      'COLL_CONTACTO',
      'COLL_CPAGOS',
      'COLL_CPAGOS_REL',
      'COLL_EFECTO',
      'COLL_ESTACTIVIDAD',
      'COLL_ESTDEUDA',
      'COLL_ESTFUNCIONES',
      'COLL_ESTOBLIGATORIOS',
      'COLL_ESTPRIORIDAD',
      'COLL_ESTRATEGIA',
      'COLL_ESTRATEGIA_PRM',
      'COLL_ESTRATEGIA_VAL',
      'COLL_ESTSUCESIONES',
      'COLL_ESTUBICABILIDAD',
      'COLL_ESTVALIDOS',
      'COLL_HISTORY',
      'COLL_INTERES_CUST',
      'COLL_MONEDAS',
      'COLL_PROMESA',
      'COLL_PROMESA_DET',
      'COLL_REFINANCIA',
      'COLL_TMOSAIX',
      'COLL_VMOSAIX',
      'CONNECTION_TYPE',
      'CUSTOMER_TYPE_PRM',
      'CUSTOMER_TYPE_VAL',
      'EIS_RESPONSE',
      'EMD_LISTA',
      'EMD_REPORTE',
      'EMD_TMP_CLAVES',
      'EN_FERIADOS',
      'EXAM_PORT_LIMITS',
      'JOB_HEADER',
      'JOB_STATS',
      'LIST_TYPE',
      'LIST_TYPE_INDI',
      'MAIL_FORMATOS',
      'MAIL_FORMATOS_DET',
      'MAIL_LISTA',
      'MAIL_LISTA_DET',
      'ORGANIZATION',
```

```
      'PHYSICAL_LIST',
      'PORTFOLIO',
      'PROGRAM_TYPE',
      'PSESSIONS',
      'PSESSIONS_FAILURES',
      'SECURITY_ACCOUNT',
      'SECURITY_ACCOUNT_HISTORY',
      'SHARED_CUSTOMER',
      'SUPERVISOR',
      'SUPERVISOR_PERMISSIONS',
      'UNIT_CALL_TYPE',
      'UNIT_CUST_TYPE',
      'UNIT_CUSTOMER',
      'UNIT_ENTITY_BROWSER',
      'UNIT_ENTITY_DETAIL',
      'UNIT_JOB_TYPE',
      'UNIT_PORTFOLIO_LIMITS',
      'USER_STATS')
ORDER BY 1 ASC, 2 ASC, 3 ASC;

-- TABLAS QUE NO SON DE SISTEMA Y ESTÁN EN EL TABLESPACE 'ENGAGEDEFAULT_WKFD'.
SELECT      *
FROM  EVI_TABLES
WHERE (TABLESPACE_NAME = 'ENGAGEDEFAULT_WKFD')
      AND TABLE_NAME NOT IN (
      'ATTENDED_CUSTOMER',
      'PHYS_CALL_ANSWERED_QUESTIONS',
      'PHYSICAL_ACTIVITIES',
      'PHYSICAL_ATTACHED_DOCUMENT',
      'PHYSICAL_CALL',
      'PHYSICAL_CALL_CONV_STEPS',
      'PHYSICAL_CALL_NUMBERS',
      'PHYSICAL_JOB',
      'PHYSICAL_JOB_ACTIONS',
      'PHYSICAL_JOB_ALARMS',
      'PHYSICAL_JOB_ATTACHED_DATA',
      'PHYSICAL_JOB_NOTIFICATIONS',
      'PHYSICAL_NOTES')
ORDER BY 1 ASC, 2 ASC, 3 ASC;

-- TABLAS QUE NO SON DE SISTEMA Y ESTÁN EN EL TABLESPACE 'ENGAGEDEFAULT_DATA'.
-- NO SE DEBE HACER NADA EN ESTE CASO. SON TABLAS AGREGADAS POR EL USUARIO.
SELECT      *
FROM  EVI_TABLES
WHERE (TABLESPACE_NAME = 'ENGAGEDEFAULT_DATA')
      AND TABLE_NAME NOT IN (
      'CUSTOMER',
      'CUSTOMER_ACCOUNTS',
      'CUSTOMER_CONTACTS',
      'CUSTOMER_DATA',
      'CUSTOMER_ENVIOS',
      'CUSTOMER_OPPORTUNITIES',
      'CUSTOMER_PAYMENTS',
      'TMT_COLLECTION')
ORDER BY 1 ASC, 2 ASC, 3 ASC;

-- TABLAS QUE NO SON DE SISTEMA Y ESTÁN EN EL TABLESPACE 'ENGAGEDEFAULT_ETL'.
-- NO SE DEBE HACER NADA EN ESTE CASO. SON TABLAS AGREGADAS POR EL USUARIO.
SELECT      *
FROM  EVI_TABLES
WHERE (TABLESPACE_NAME = 'ENGAGEDEFAULT_ETL')
      AND TABLE_NAME NOT IN (
      'EDA_REGISTRO_SUCESOS',
      'ENGAGELOG',
      'PHYSICAL_CALL_TRANSACTIONS')
ORDER BY 1 ASC, 2 ASC, 3 ASC;

-- TABLAS QUE NO TIENEN RESTRICCIÓN DE CLAVE PRIMARIA. ORDENADAS POR TABLESPACE Y NOMBRE.
SELECT      T1.*
FROM  EVI_TABLES T1
WHERE NOT EXISTS (
      SELECT      1
      FROM  EVI_TABLE_CONSTRAINTS T2
      WHERE (T2.TABLE_NAME = T1.TABLE_NAME)
            AND (T2.CONSTRAINT_TYPE = 'PRIMARY KEY')
      )
ORDER BY 1 ASC, 2 ASC, 4 ASC, 3 ASC;
```

SOLUCIONES
Best Technology for Best Business

```
-- TABLAS QUE NO TIENEN RESTRICCIÓN UNIQUE. ORDENADAS POR TABLESPACE Y NOMBRE.
SELECT      T1.*
FROM  EVI_TABLES T1
WHERE TABLESPACE_NAME NOT IN (
      'ENGAGEDEFAULT_DSST',
      'ENGAGEDEFAULT_MD',
      'ENGAGEDEFAULT_SD',
      'ENGAGEDEFAULT_WKFD')
      AND TABLE_NAME NOT IN (
      'EDA_REGISTRO_SUCESOS',
      'ENGAGELOG',
      'PHYSICAL_CALL_TRANSACTIONS')
      AND NOT EXISTS (
      SELECT      1
      FROM  EVI_TABLE_CONSTRAINTS T2
      WHERE (T2.TABLE_NAME = T1.TABLE_NAME)
            AND (T2.CONSTRAINT_TYPE = 'UNIQUE')
      )
ORDER BY 1 ASC, 2 ASC, 4 ASC, 3 ASC;

-- TABLAS QUE NO TIENEN ÍNDICES. ORDENADAS POR TABLESPACE Y NOMBRE.
SELECT      T1.*
FROM  EVI_TABLES T1
WHERE TABLESPACE_NAME NOT IN (
      'ENGAGEDEFAULT_DSST',
      'ENGAGEDEFAULT_MD',
      'ENGAGEDEFAULT_SD',
      'ENGAGEDEFAULT_WKFD')
      AND TABLE_NAME NOT IN (
      'EDA_REGISTRO_SUCESOS',
      'ENGAGELOG',
      'PHYSICAL_CALL_TRANSACTIONS')
      AND NOT EXISTS (
      SELECT      1
      FROM  EVI_TABLE_INDEXES T2
      WHERE (T2.TABLE_NAME = T1.TABLE_NAME)
            AND NOT EXISTS (
            SELECT      1
            FROM  EVI_TABLE_CONSTRAINTS T3
            WHERE (T3.CONSTRAINT_NAME = T2.INDEX_NAME)
                  AND (T3.CONSTRAINT_TYPE IN ('PRIMARY KEY', 'UNIQUE'))
            )
      )
ORDER BY 1 ASC, 2 ASC, 4 ASC, 3 ASC;

-- COLUMNAS DE TABLAS DE LA BASE DE DATOS QUE NO EXISTEN EN LA TABLA DEFINIDA EN LA METADATA
DE ENGAGE.
SELECT      T2.TABLE_NAME,
      T2.ORDINAL_POSITION,
      T2.COLUMN_NAME,
      T2.IS_NULLABLE,
      T2.DATA_TYPE_NAME,
      T2.BYTE_LENGTH,
      T2.CHAR_COL_DECL_LENGTH,
      T2.NUMERIC_PRECISION,
      T2.NUMERIC_SCALE
FROM  EVI_TABLES T1
      INNER JOIN EVI_TABLE_COLUMNS T2 ON (T1.TABLE_NAME = T2.TABLE_NAME)
WHERE (T1.TABLESPACE_NAME = 'ENGAGEDEFAULT_DATA')
      AND T1.TABLE_NAME NOT IN (
      'CUSTOMER',
      'CUSTOMER_ACCOUNTS',
      'CUSTOMER_CONTACTS',
      'CUSTOMER_DATA',
      'CUSTOMER_ENVIOS',
      'CUSTOMER_OPPORTUNITIES',
      'CUSTOMER_PAYMENTS',
      'TMT_COLLECTION')
      AND T1.TABLE_NAME NOT LIKE 'ELS_%'
      AND  T2.COLUMN_NAME  NOT  IN  ('PKEY',  'PAR_KEY',  'TS_BEGIN',  'TS_END',  'TS_USER_ID',
'ATT_PKEY')
      AND NOT EXISTS (
      SELECT      1
      FROM  ENTITY_BROWSER EB
            INNER JOIN ENTITY_BROWSER_DETAIL EBD ON (EB.PKEY = EBD.PAR_KEY)
      WHERE (EB.ENTITY_TABLE_REL = T2.TABLE_NAME)
            AND (EBD.FIELD_NAME = T2.COLUMN_NAME)
      )
```

SOLUCIONES
Best Technology for Best Business

```
ORDER BY 1 ASC, 2 ASC;

-- COLUMNAS DE TABLAS DE LA BASE DE DATOS CON TIPO DE DATO DIFERENTE QUE EN LA MISMA TABLA
DEFINIDA EN LA METADATA DE ENGAGE.
-- LUEGO DE ELIMINAR LAS COLUMNAS DE LAS TABLAS, SI CORRESPONDE, HABRÁ QUE PONER VIGENTES A
LAS ENTIDADES DESDE ENGAGE DESIGNER.
SELECT      T2.TABLE_NAME,
      T2.ORDINAL_POSITION,
      T2.COLUMN_NAME,
      T2.IS_NULLABLE,
      T2.DATA_TYPE_NAME,
      EBD.FIELD_TYPE,
      T2.BYTE_LENGTH,
      T2.CHAR_COL_DECL_LENGTH,
      T2.NUMERIC_PRECISION,
      T2.NUMERIC_SCALE
FROM  EVI_TABLES T1
      INNER JOIN EVI_TABLE_COLUMNS T2 ON (T1.TABLE_NAME = T2.TABLE_NAME)
      INNER JOIN
      (ENTITY_BROWSER EB
      INNER JOIN ENTITY_BROWSER_DETAIL EBD ON (EB.PKEY = EBD.PAR_KEY)
      ) ON (EB.ENTITY_TABLE_REL = T2.TABLE_NAME) AND (EBD.FIELD_NAME = T2.COLUMN_NAME)
WHERE (T1.TABLESPACE_NAME = 'ENGAGEDEFAULT_DATA')
      AND T1.TABLE_NAME NOT IN (
      'CUSTOMER',
      'CUSTOMER_ACCOUNTS',
      'CUSTOMER_CONTACTS',
      'CUSTOMER_DATA',
      'CUSTOMER_ENVIOS',
      'CUSTOMER_OPPORTUNITIES',
      'CUSTOMER_PAYMENTS',
      'TMT_COLLECTION')
      AND T1.TABLE_NAME NOT LIKE 'ELS_%'
      AND  T2.COLUMN_NAME  NOT  IN  ('PKEY',  'PAR_KEY',  'TS_BEGIN',  'TS_END',  'TS_USER_ID',
'ATT_PKEY')
      AND (EBD.FIELD_TYPE <> CASE T2.DATA_TYPE_NAME
                        WHEN 'NUMBER' THEN 'entero'
                        WHEN 'NUMERIC' THEN 'entero'
                        WHEN 'DATE' THEN 'fecha'
                        WHEN 'DATETIME' THEN 'fecha'
--                      WHEN  'VARCHAR2'  THEN  (CASE  T2.CHAR_COL_DECL_LENGTH  WHEN  1  THEN
'logico' ELSE 'texto' END)
--                      WHEN 'VARCHAR' THEN (CASE T2.CHAR_COL_DECL_LENGTH WHEN 1 THEN 'logico'
ELSE 'texto' END)
                        WHEN 'VARCHAR2' THEN 'texto'
                        WHEN 'VARCHAR' THEN 'texto'
                        ELSE T2.DATA_TYPE_NAME END)
ORDER BY 1 ASC, 2 ASC;

-- COLUMNAS DE TABLAS DE LA BASE DE DATOS CON TIPO DE DATO 'texto' PERO DIFERENTE LONGITUD QUE
EN LA MISMA TABLA DEFINIDA EN LA METADATA DE ENGAGE.
-- LUEGO DE ELIMINAR LAS COLUMNAS DE LAS TABLAS, SI CORRESPONDE, HABRÁ QUE PONER VIGENTES A
LAS ENTIDADES DESDE ENGAGE DESIGNER.
SELECT      T2.TABLE_NAME,
      T2.ORDINAL_POSITION,
      T2.COLUMN_NAME,
      T2.IS_NULLABLE,
      T2.DATA_TYPE_NAME,
      T2.BYTE_LENGTH,
      T2.CHAR_COL_DECL_LENGTH,
      EBD.FIELD_TYPE,
      EBD.FIELD_LENGTH
FROM  EVI_TABLES T1
      INNER JOIN EVI_TABLE_COLUMNS T2 ON (T1.TABLE_NAME = T2.TABLE_NAME)
      INNER JOIN
      (ENTITY_BROWSER EB
      INNER JOIN ENTITY_BROWSER_DETAIL EBD ON (EB.PKEY = EBD.PAR_KEY)
      ) ON (EB.ENTITY_TABLE_REL = T2.TABLE_NAME) AND (EBD.FIELD_NAME = T2.COLUMN_NAME)
WHERE (T1.TABLESPACE_NAME = 'ENGAGEDEFAULT_DATA')
      AND T1.TABLE_NAME NOT IN (
      'CUSTOMER',
      'CUSTOMER_ACCOUNTS',
      'CUSTOMER_CONTACTS',
      'CUSTOMER_DATA',
      'CUSTOMER_ENVIOS',
      'CUSTOMER_OPPORTUNITIES',
      'CUSTOMER_PAYMENTS',
      'TMT_COLLECTION')
```

```
      AND T1.TABLE_NAME NOT LIKE 'ELS_%'
      AND T2.COLUMN_NAME NOT IN ('PKEY', 'PAR_KEY', 'TS_BEGIN', 'TS_END', 'TS_USER_ID',
'ATT_PKEY')
      AND (CASE T2.DATA_TYPE_NAME
            WHEN 'NUMBER' THEN 'entero'
            WHEN 'NUMERIC' THEN 'entero'
            WHEN 'DATE' THEN 'fecha'
            WHEN 'DATETIME' THEN 'fecha'
--          WHEN 'VARCHAR2' THEN (CASE T2.CHAR_COL_DECL_LENGTH WHEN 1 THEN 'logico' ELSE
'texto' END)
--          WHEN 'VARCHAR' THEN (CASE T2.CHAR_COL_DECL_LENGTH WHEN 1 THEN 'logico' ELSE
'texto' END)
            WHEN 'VARCHAR2' THEN 'texto'
            WHEN 'VARCHAR' THEN 'texto'
            ELSE T2.DATA_TYPE_NAME END = 'texto')
      AND (EBD.FIELD_LENGTH <> T2.CHAR_COL_DECL_LENGTH)
ORDER BY 1 ASC, 2 ASC;

-- COLUMNAS DE TABLAS DE LA BASE DE DATOS CON TIPO DE DATO 'entero' PERO DIFERENTE LONGITUD Y
PRECISIÓN QUE EN LA MISMA TABLA DEFINIDA EN LA METADATA DE ENGAGE.
-- LUEGO DE ELIMINAR LAS COLUMNAS DE LAS TABLAS, SI CORRESPONDE, HABRÁ QUE PONER VIGENTES A
LAS ENTIDADES DESDE ENGAGE DESIGNER.
SELECT      T2.TABLE_NAME,
      T2.ORDINAL_POSITION,
      T2.COLUMN_NAME,
      T2.IS_NULLABLE,
      T2.DATA_TYPE_NAME,
      T2.NUMERIC_PRECISION,
      T2.NUMERIC_SCALE,
      EBD.FIELD_TYPE,
      EBD.FIELD_LENGTH,
      EBD.FIELD_DECIMALS
FROM  EVI_TABLES T1
      INNER JOIN EVI_TABLE_COLUMNS T2 ON (T1.TABLE_NAME = T2.TABLE_NAME)
      INNER JOIN
      (ENTITY_BROWSER EB
      INNER JOIN ENTITY_BROWSER_DETAIL EBD ON (EB.PKEY = EBD.PAR_KEY)
      ) ON (EB.ENTITY_TABLE_REL = T2.TABLE_NAME) AND (EBD.FIELD_NAME = T2.COLUMN_NAME)
WHERE (T1.TABLESPACE_NAME = 'ENGAGEDEFAULT_DATA')
      AND T1.TABLE_NAME NOT IN (
      'CUSTOMER',
      'CUSTOMER_ACCOUNTS',
      'CUSTOMER_CONTACTS',
      'CUSTOMER_DATA',
      'CUSTOMER_ENVIOS',
      'CUSTOMER_OPPORTUNITIES',
      'CUSTOMER_PAYMENTS',
      'TMT_COLLECTION')
      AND T1.TABLE_NAME NOT LIKE 'ELS_%'
      AND T2.COLUMN_NAME NOT IN ('PKEY', 'PAR_KEY', 'TS_BEGIN', 'TS_END', 'TS_USER_ID',
'ATT_PKEY')
      AND (CASE T2.DATA_TYPE_NAME
            WHEN 'NUMBER' THEN 'entero'
            WHEN 'NUMERIC' THEN 'entero'
            WHEN 'DATE' THEN 'fecha'
            WHEN 'DATETIME' THEN 'fecha'
--          WHEN 'VARCHAR2' THEN (CASE T2.CHAR_COL_DECL_LENGTH WHEN 1 THEN 'logico' ELSE
'texto' END)
--          WHEN 'VARCHAR' THEN (CASE T2.CHAR_COL_DECL_LENGTH WHEN 1 THEN 'logico' ELSE
'texto' END)
            WHEN 'VARCHAR2' THEN 'texto'
            WHEN 'VARCHAR' THEN 'texto'
            ELSE T2.DATA_TYPE_NAME END = 'entero')
      AND (EBD.FIELD_LENGTH <> (T2.NUMERIC_PRECISION - T2.NUMERIC_SCALE)
      OR EBD.FIELD_DECIMALS <> T2.NUMERIC_SCALE)
ORDER BY 1 ASC, 2 ASC;
```