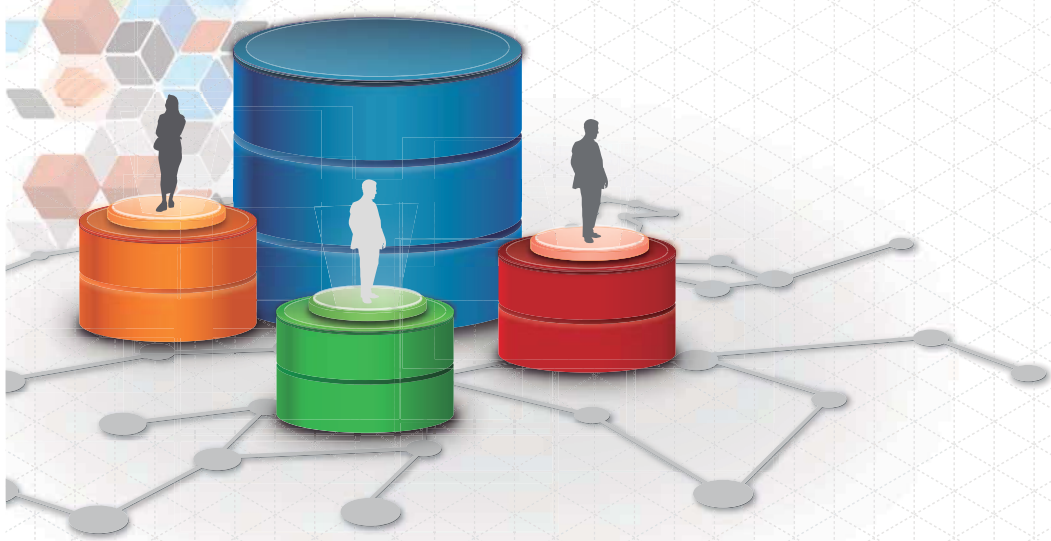
Programación TRANSACT SQL Server 2012



PROGRAMACIÓN TRANSACT CON SQL SERVER 2012
Autor: Manuel Angel Torres Remon

Edición a cargo de:
Empresa Editora Macro E.I.R.L.
Av. Paseo de la República 5613 - Miraflores
Lima - Perú
(511) 719-9700
ventas@editorialmacro.com
http://www.editorialmacro.com

Primera edición: Octubre 2012 - 1000 ejemplares

Impreso en los Talleres Gráficos de
Empresa Editora Macro E.I.R.L.
Lima - Perú

ISBN Nº 978-612-304-084-0
Hecho el Depósito Legal en la Biblioteca Nacional del Perú Nº 2012-12379

MANUEL ANGEL TORRES REMON

Manuel Torres es un consultor dedicado a la docencia de cursos de tecnología como Visual NET y Microsoft SQL Server; ha brindado capacitación en las instituciones más importantes de Lima-Perú.

Recibió su formación tecnológica en el Instituto Superior Tecnológico Público Manuel Arévalo Cáceres y también en la Universidad Alas Peruanas de la República del Perú. En ambas instituciones recibió una buena formación profesional, demostrada en las diferentes instituciones en que laboró.

Actualmente se desempeña como consultor tecnológico de grandes empresas como TopyTop, Nestle y como docente en instituciones educativas como el instituto Manuel Arévalo Cáceres, Cibertec y Unimaster de la Universidad Nacional de Ingeniería en todas ellas impartiendo cursos de tecnología especialmente en Análisis, Programación y Base de Datos.

Ha publicado el libro Programación VBA con Excel donde expone que los usuarios de oficina pueden programar de manera profesional sin necesidad de ser expertos en programación y también publicó la Guía Práctica de Programación VBA con Excel.

Se le puede localizar al email manuel.torresr@hotmail.com o a los teléfonos (511)-982360540 / (511)-996396023

Agradecimientos

Cada libro desarrollado es una labor en equipo, donde el autor sacrifica muchas veces tiempo con la familia, horas de descanso y otras actividades que tienen su recompensa cuando el material es una realidad. No cabe duda que me complace compartir con ustedes los casos expuestos ya que me facilitará en el desempeño de mis labores como docente, por eso este agradecimiento en primer lugar a la familia MACRO por confiar nuevamente en mi persona para el desarrollo de este material.

En segundo lugar agradecer enormemente a las personas que me apoyaron como lo son mis hijas Ángela Victoria y Fernanda Ximena Torres Lázaro y mi esposa Luz que con mucha paciencia me han dado el tiempo necesario para la elaboración de este material, eso para mí es un gran sacrificio que lo agradeceré por siempre. Gracias de nuevamente.

Introducción

Server 2012 es un sistema integrado de gestión de base de datos; hoy en día las organizaciones necesitan tener un control automatizado de sus archivos, es decir, una simple factura no necesariamente tendrá que ser física en la actualidad se la puede enviar por email; por ejemplo, en Perú mediante las leyes 26612 y 681, se aprueba el uso de las imágenes como medio de sustento legal, considerándose las mismas con el mismo valor legal que el documento original, con esto se pueden dejar de lado la papelería para pasar a los archivos digitales; SQL puede tener el control de sus archivos administrándolos de manera eficaz, rápida y segura.

Programación Transact SQL propone realizar procesos de manera profesional por medio de script que se ejecutará tanto en el cliente como en el servidor y que será de gran utilidad dominarlo. Los lenguajes de Programación siembran la cultura de la programación nativa mientras que SQL Server propone instrucciones o sentencias para la obtención de resultados, Transact SQL rompe ese esquema y le quita un poco de protagonismo a los lenguajes de programación usando sus estructuras que son entendidas en el motor de base de datos de SQL Server 2012.

Transact SQL Server 2012 tiene como fundamento primordial gestionar la información almacenada en una base de datos sin dejar toda la responsabilidad de la gestión a los lenguajes de programación, más bien usa de ellos sus estructuras como el If o While para procesar reglas de negocio.

La versión de SQL Server 2012 no presenta grandes cambios en el trabajo de programación con Transact-SQL así es que si no tiene esta versión podrá ejecutar los casos desarrollados con SQL Server 2008.

A quién está dirigido este libro

Los administradores de base de datos ya tienen definida su labor frente a una base de datos, pero este material propone ver más allá de la administración y preparalos al desarrollo, no se implementará una aplicación como lo hace Visual Net o Java pero se podrá programar consultas e implementar de manera profesional los procedimientos almacenados, las funciones, los cursores y muchos temas que verá en este material.

Un pre-requisito para este material es tener un conocimiento básico de los comandos e instrucciones de SQL Server, además de conocer el objetivo de las estructuras selectivas o repetivas eso será suficiente para que este material le sea útil, mejor dicho es el siguiente nivel de un usuario que administra una base de datos.

Este material no pretende ser una introducción a la programación Transact más bien propone a programación avanzada de SQL Server exigiendo al motor de base de datos la responsabilidad de compilar una porción de código Transact.

Índice

Capítulo 1

Capítulo 2

CAP.

1

Microsoft SQL Server 2012

CAPACIDAD:

El lector podrá reconocer las diferencias entre las versiones de SQL Server, además de aplicar con casos desarrollados las sentencias pertenecientes al lenguaje de definición de datos DDL.

Primero se reconocerá las versiones de SQL Server y luego se procederá a mostrar paso a paso la instalación de SQL Server 2012. Al final del capítulo se presentarán casos desarrollados por cada sentencia del lenguaje de definición de datos.

CONTENIDO:

- Definición de Microsoft SQL Server 2012
- Versiones SQL Server 2012
- Características de SQL Server 2012
- Preparando la instalación de SQL Server 2012
- Pre-requisitos para la instalación de SQL Server 2012
- Pantalla inicial de autorun del CD de instalación
- Configuración de fuente para el entorno de trabajo
- Lenguaje de Definición de Datos (LDD)
- Sentencia Create
- Sentencia Alter
- Sentencia Drop

1.1. DEFINICIÓN DE MICROSOFT SQL SERVER 2012

Microsoft SQL Server es un sistema para la gestión de bases de datos producidos por Microsoft basado en el modelo relacional. Sus lenguajes para consultas son Transact-SQL y ANSI SQL. Microsoft SQL Server constituye la alternativa de Microsoft a otros potentes sistemas gestores de bases de datos como son Oracle, PostgreSQL o MySQL.

Las principales características de Microsoft SQL Server 2012 son:

- Ofrece a los desarrolladores de base de datos un soporte potente de transacciones.
- Soporte de procedimientos almacenados.
- Todas las versiones de SQL Server presentan un entorno gráfico de administración de los objetos del motor de base de datos, que permite el uso de comandos DDL y DML gráficamente.
- Permite trabajar en modo cliente-servidor, donde la información y datos se alojan en el servidor y los terminales o clientes de la red sólo acceden a la información.
- Permite la administración de información de otros servidores de datos y no necesariamente el mismo sistema operativo.

A partir de la versión 2005 se incluyó dentro del sistema la versión reducida que la llamaron MSDE con el mismo motor de base de datos, pero orientado a proyectos más pequeños, reduciendo el espacio en disco y lo engorroso que podría ser instalar la versión completa del SQL Server, a esta versión se le llamo SQL Express Edition, que se distribuye en forma gratuita desde la página oficial de microsoft http://msdn.microsoft.com. En la versión 2008 sale una nueva utilidad dentro de la administración de base de datos que conllevaba a gestionar base de datos distribuidas.

1.2. VERSIONES SQL SERVER 2012

Veamos una tabla de comparación Versión, Año y Nombre clave en donde veremos la evolución de SQL Server hasta la última versión 2012.

VERSIÓN	AÑO DE LANZAMIENTO	NOMBRE DEL PROYECTO
1.0	1989	SQL
4.21	1993	SEQUEL
6.0	1995	SQL95
6.5	1996	Hydra
7.0	1998	Sphinx
8.0	2000 2003	Shiloh (SQL Server 2000) Liberty (SQL Server 2000 64 bit)
9.0	2005	Yukon (SQL Server 2005)
10.0	2008	Katmai (SQL Server 2008)
10.5	2010	Kilimanjaro (SQL Server 2008 R2)
11.0	2012	Denali (SQL Server 2012)

1.3. Características de SQL Server 2012

Las principales características y los puntos más destacables de SQL Server 2012 son:

- Mayor disponibilidad. Alcance de los 9s exigidos de disponibilidad y el nivel de protección de datos que requiere su organización con AlwaysOn, que ahora ofrece más funcionalidades que la versión CTP1 y permite a los clientes disfrutar un nivel aún mayor de flexibilidad y valor de negocio.
- Los análisis más avanzados. Descubra la potencia escondida en sus datos con análisis mucho más potentes y una exploración de datos más rápida a través de toda su organización Power View, disponible para los clientes de SQL Server por primera vez.
- Datos creíbles y consistentes. Ofrezca a sus usuarios una visión consistente de la información entre orígenes de datos muy diversos con el Modelo de Semántica de BI (BI Semantic Model, BISM) un modelo unificado y común para las aplicaciones de Business Intelligence. La calidad de los datos dejará de ser una tarea habitual gracias al complemento Master Data Services para Excel y los nuevos Data Quality Services, que se integran con proveedores de datos externos disponibles desde el Datamarket de Windows Azure Marketplace. Los clientes pueden probar ya esta funcionalidad.
- Una experiencia de desarrollo productiva. Aumente la productividad de sus departamentos de TI y desarrollo tanto en sus propias instalaciones de servidor como en la nube, con el Componente de Aplicación de Capa de Datos (DAC, Data-tier Application Component) que establece una relación de paridad con SQL Azure y SQL Server Data Tools para lograr una experiencia de desarrollo unificada y moderna en funciones de bases de datos, Business Intelligence y en la nube. Además, los clientes de SQL Server Express Edition pueden probar una nueva versión LocalDB para instalaciones rápidas sin configuración.

1.4. Preparando la instalación de SQL Server 2012

SQL Server presenta ediciones de 32 y 64 bits para lo cual se recomienda tener las siguientes consideraciones antes de empezar la instalación del producto:

- Se recomienda instalar el SQL Server 2012 en equipos con formatos NTFS por ser más seguro que los de formato FAT32; pero finalmente es sólo recomendación ya que también se puede instalar en este último.
- El programa de instalación de SQL Server 2012 bloqueará las instalaciones en unidades de disco de sólo lectura, asignadas o comprimidas.
- SQL Server 2012 requiere que se instale una actualización para asegurarse de que se puede instalar correctamente el componente de Visual Studio. El programa de instalación de SQL Server comprueba la presencia de esta actualización y, a continuación, le exige que descargue e instale la actualización antes de continuar con la instalación de SQL Server. Para evitar la interrupción durante la instalación de SQL Server, puede descargar e instalar la actualización antes de ejecutar el programa de instalación de SQL Server, según se describe a continuación (o instalar todas las actualizaciones de .NET 3.5 SP1 disponibles en Windows Update):

1.5. PRE-REQUISITOS PARA LA INSTALACIÓN DE SQL SERVER 2012

HARDWARE

Memoria: recomendado

- SQL Server Express 1GB
- Todas las demás versiones 4GB

Procesador: mínimo

- Procesador X86 : 1GHZ
- Procesador X64 : 2GHZ a más

Disco Duro: mínimo 6GB de espacio libre

A continuación listaremos la distribución de espacios requeridos por características de SQL Server 2012:

- Motor de Base de datos : 811MB
- Servicio de Análisis y archivo de datos : 345MB
- Servicio de Reportes y administración de Informes : 304MB
- Servicios de Integración : 591MB
- Servicios de Datos Maestros : 243MB
- Componentes de Cliente : 1.78GB
- Libros en pantalla de SQL Server : 375KB

FRAMEWORK

.NET 3.5 SP1 es un requisito de SQL Server 2012 al seleccionar el Motor de base de datos, Reporting Services, Replicación, Data Quality Services, Master Data Services o SQL Server Management Studio, y el programa de instalación de SQL Server ya no lo instala. Si el programa de instalación se ejecuta en un equipo con el sistema operativo Windows Server 2008 R2 SP1, debe habilitar .NET Framework 3.5 SP1 antes de instalar SQL Server 2012.

.NET 4.0 es un requisito para SQL Server 2012. SQL Server instala .NET 4.0 durante el paso de instalación de características. El programa de instalación de SQL Server instala los siguientes componentes de software requeridos por el producto:

- .NET Framework 4 1
- SQL Server Native Client
- Archivos auxiliares para la instalación de SQL Server

WINDOWS POWER SHELL

SQL Server 2012 no instala ni habilita Windows PowerShell 2.0; sin embargo, Windows PowerShell 2.0 es un requisito previo de instalación para los componentes del Motor de base de datos y SQL Server Management Studio. Si el programa de instalación notifica que Windows PowerShell 2.0 no está presente, puede instalarlo o habilitarlo siguiendo las instrucciones de la página Windows Management Framework.

- **Red**

  Los sistemas operativos admitidos para SQL Server 2012 tienen software de red integrado. Las instancias con nombre y predeterminadas de una instalación independiente admiten los siguientes protocolos de red:

  - Memoria compartida
  - Canalizaciones con nombre
  - TCP/IP
  - VIA

- **Virtualización**

  SQL Server 2012 se admite en entornos de máquina virtual que se ejecuten en el rol Hyper-V de las ediciones Standard, Enterprise y Datacenter de Windows Server 2008 SP2 y las ediciones Standard, Enterprise y Datacenter de Windows Server 2008 R2 SP1.

  Además de los recursos requeridos por la partición primaria, a cada máquina virtual (partición secundaria) se le deben proporcionar suficientes recursos de procesador, memoria y recursos de disco para su instancia de SQL Server 2012.

  En el rol Hyper-V de Windows Server 2008 SP2 y Windows Server 2008 R2 SP1, se puede asignar un máximo de cuatro procesadores virtuales a las máquinas virtuales que ejecuten las versiones de 32 o 64 bits de Windows Server 2008 SP2 o Windows Server 2008 R2 SP1.

- **Navegador**

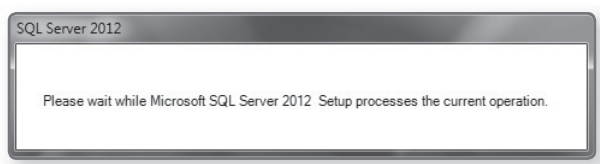
  Se requiere Internet Explorer 7 o una versión posterior para Microsoft Management Console (MMC), Herramientas de datos de SQL Server (SSDT), el componente Diseñador de informes de Reporting Services y la Ayuda HTML.

1.6. Pantalla inicial de autorun del CD de instalación

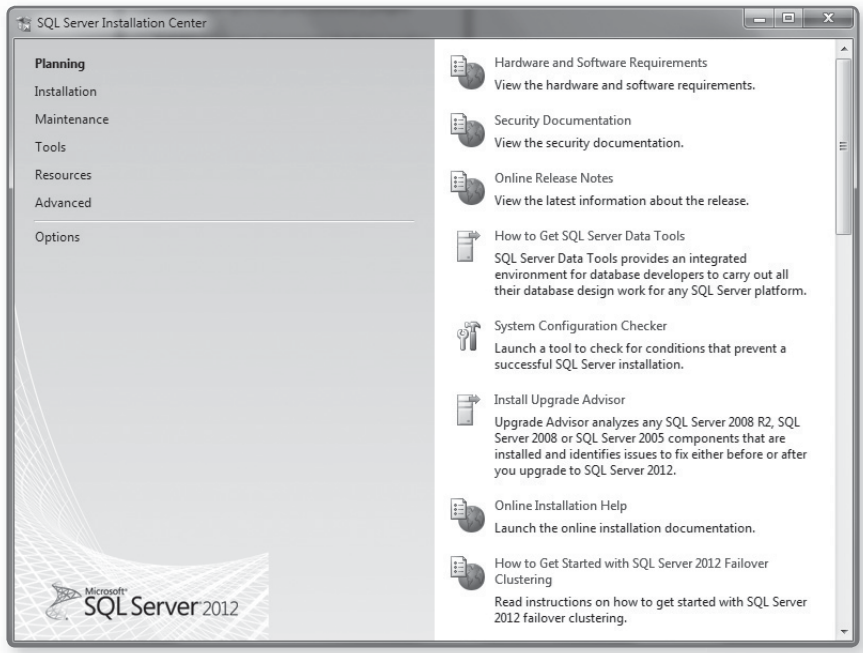
Para obtener el software puede descargar la versión de prueba desde la página oficial www.microsoft.com.

El Asistente para instalación se ejecutara después de seleccionar el archivo SETUP.EXE

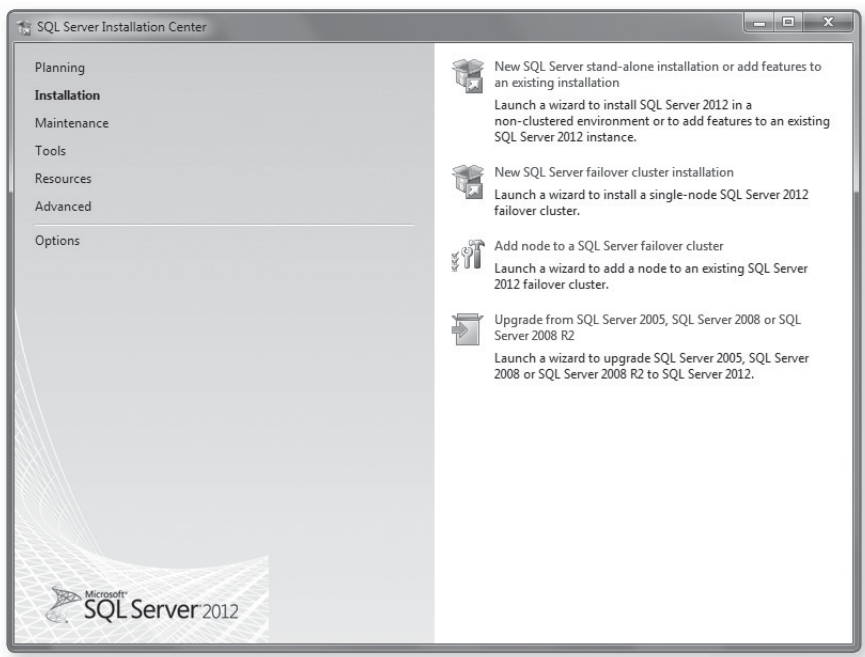
El asistente verificará los requisitos mínimos tanto del hardware como del software para continuar con la instalación, mostrando la siguiente imagen:



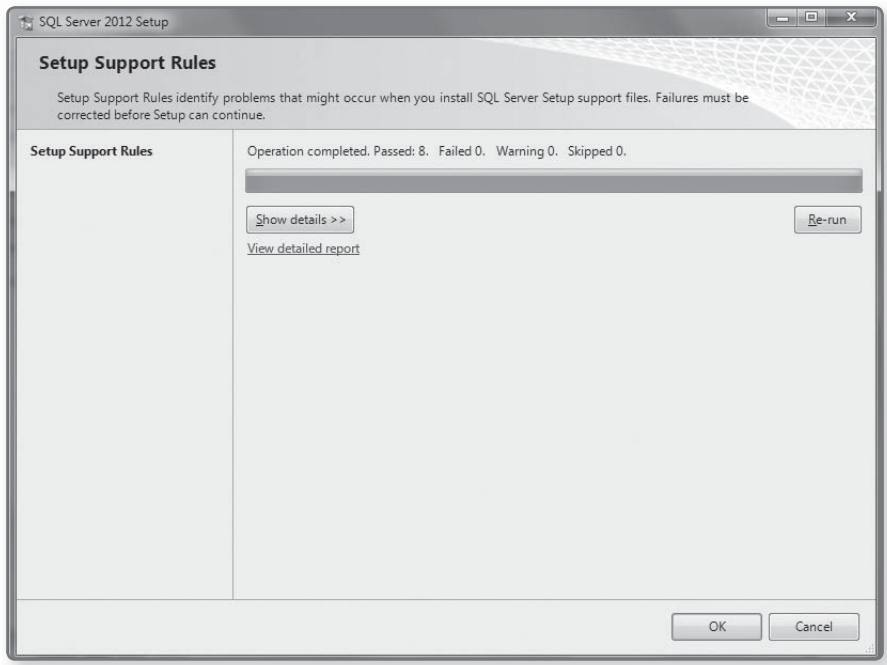
Luego se presenta el Centro de Instalación de SQL Server, a partir de aquí se tendrá que configurar características propias de SQL Server 2012. Selección **Installation** de la imagen siguiente:



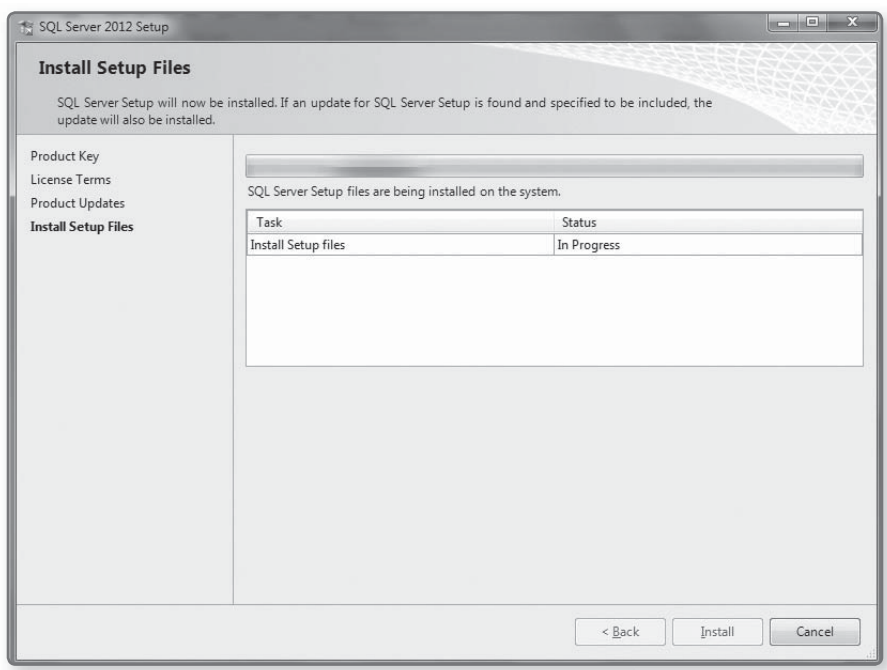
Seguidamente seleccione: **New SQL Server stand-alone installation or add features to an existing installation**, para poder seleccionar una nueva instancia de la instalación de SQL Server 2012.



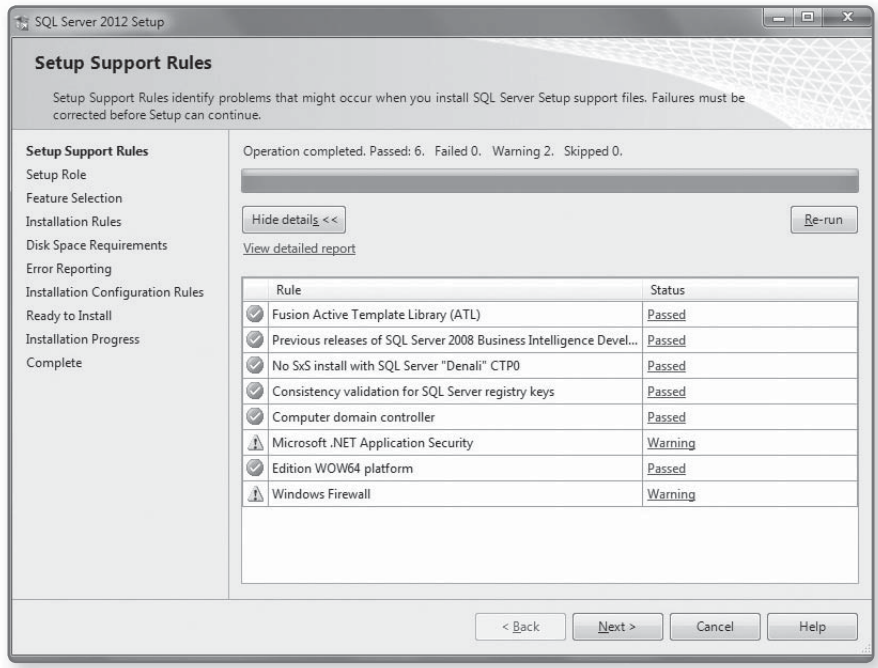
Como vemos en la siguiente imagen sólo será cuestión de esperar las comprobaciones propias de SQL Server 2012, si hubiera algún problema durante esta búsqueda tendrá que subsanarla para poder continuar con la misma, si desea ver cuáles son los errores presione **Show details>>**.



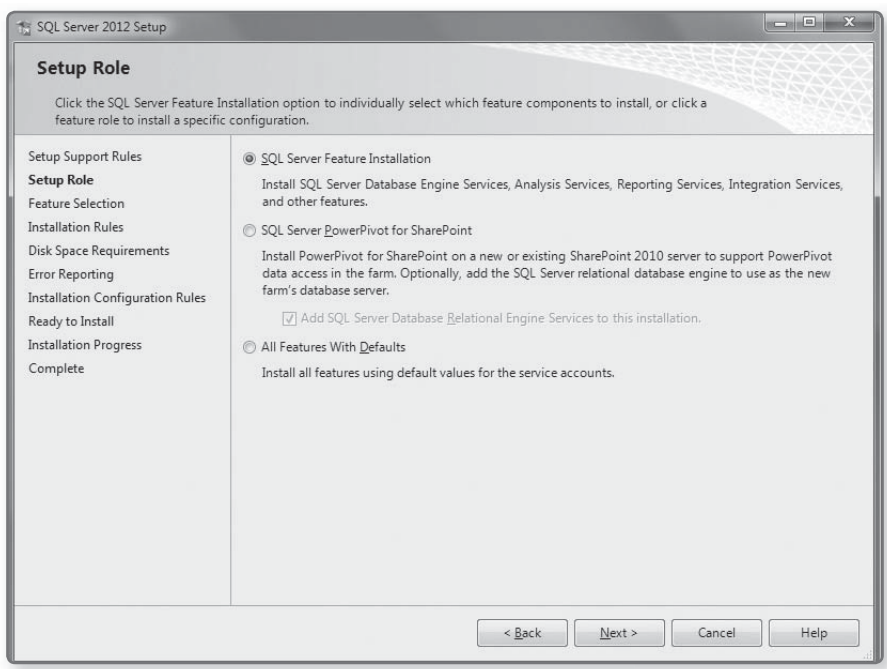
Toda vez verificando las reglas de SQL Server 2012, el asistente comprobará si todas las especificaciones están correctas.



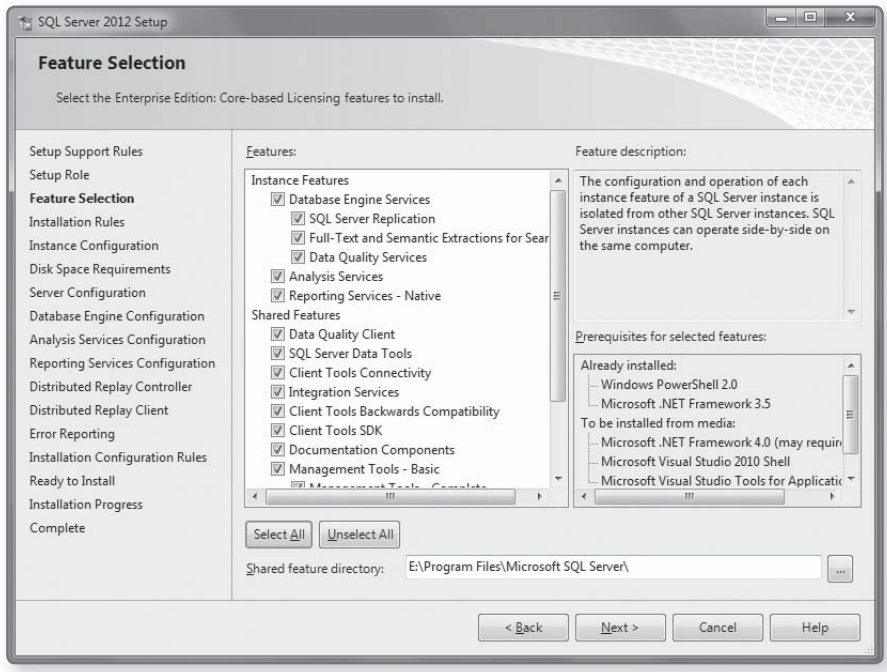
Consideremos el caso que no tenga instalado en su computador Visual NET ni un Firewall habilitado, entonces se mostrara el siguiente resumen:



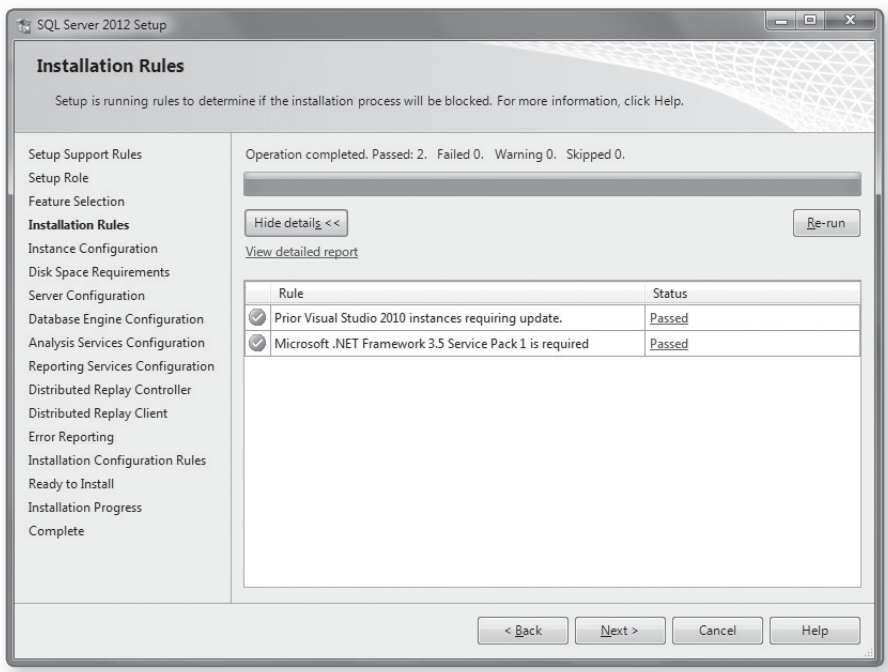
En este caso estos errores de cuidado no son relevantes en la instalación de SQL Server así que podemos continuar con la instalación presionando el botón **Next>**. Seguidamente en la ventana Setup Role se debe seleccionar **SQL Server Feature Installation**.



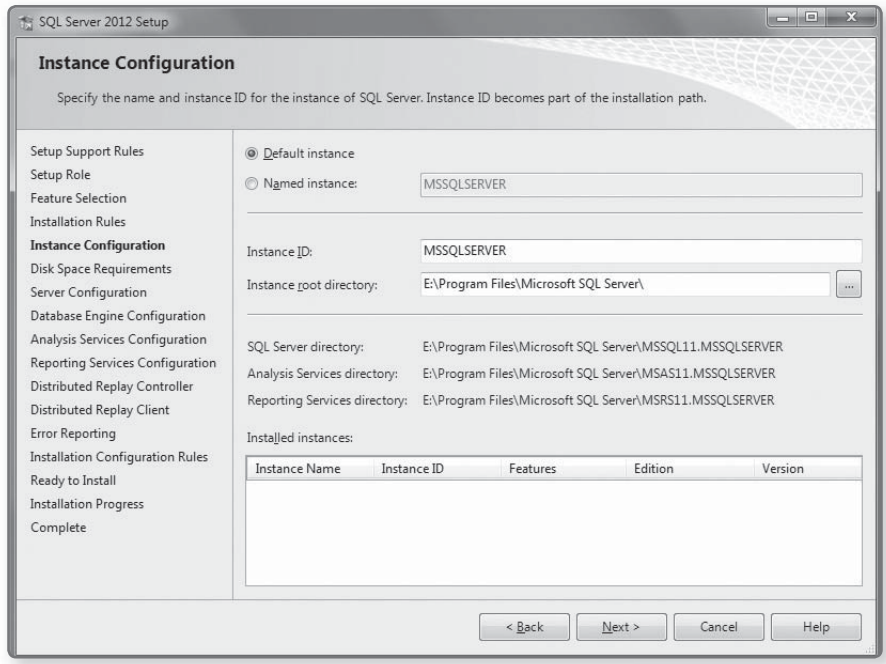
Seguidamente debemos seleccionar las características de la instalación, es recomendado seleccionar **Select All**:



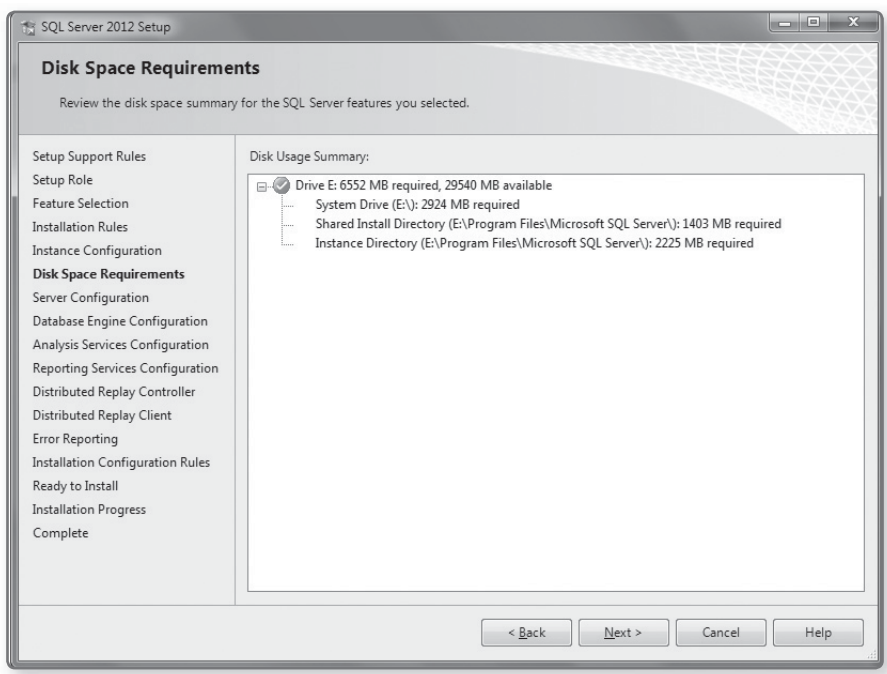
Luego se procede a instalar los requerimientos básicos de SQL Server 2012, en este caso no hubo errores ya que en esta instalación no hay Visual NET.



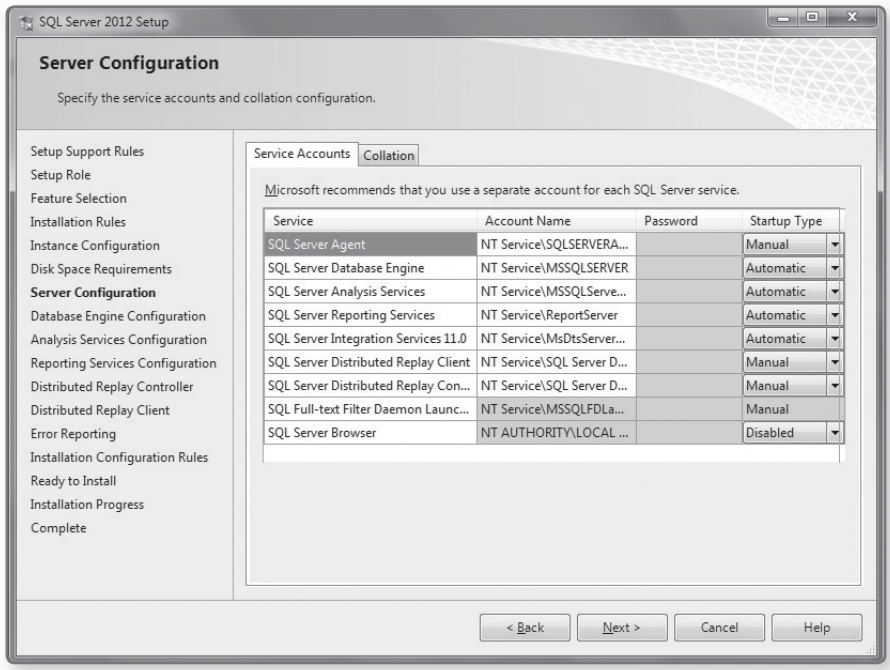
En la ventana **Instance Configuration** se debe configurar el tipo de instancia a instalar y definir el lugar donde se grabaran los archivos de SQL Server 2012, normalmente se direcciona en Archivos de Programa.



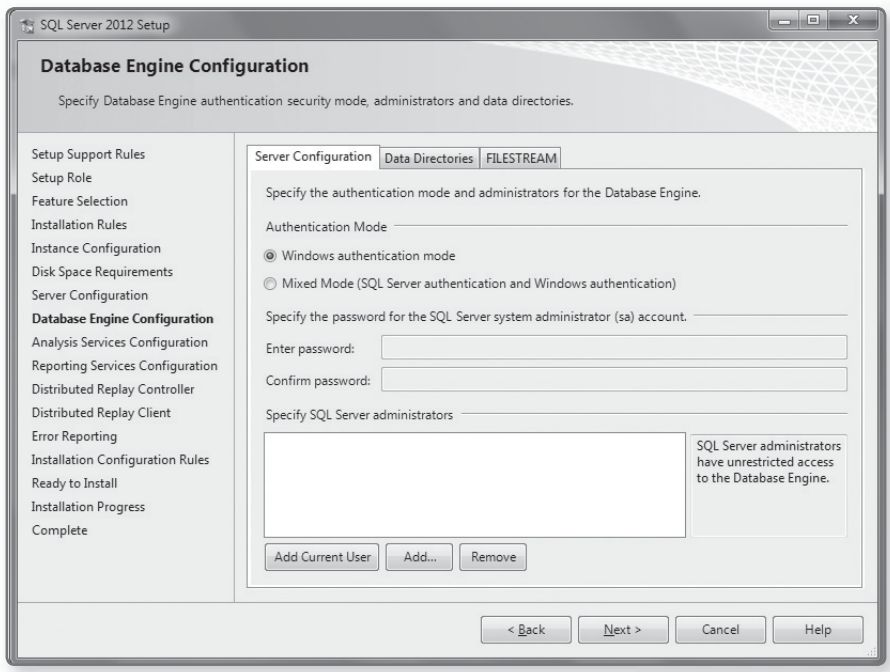
El asistente verificará los requerimientos de espacio en disco antes de proceder con la instalación de los archivos en la ubicación especificada en el paso anterior.



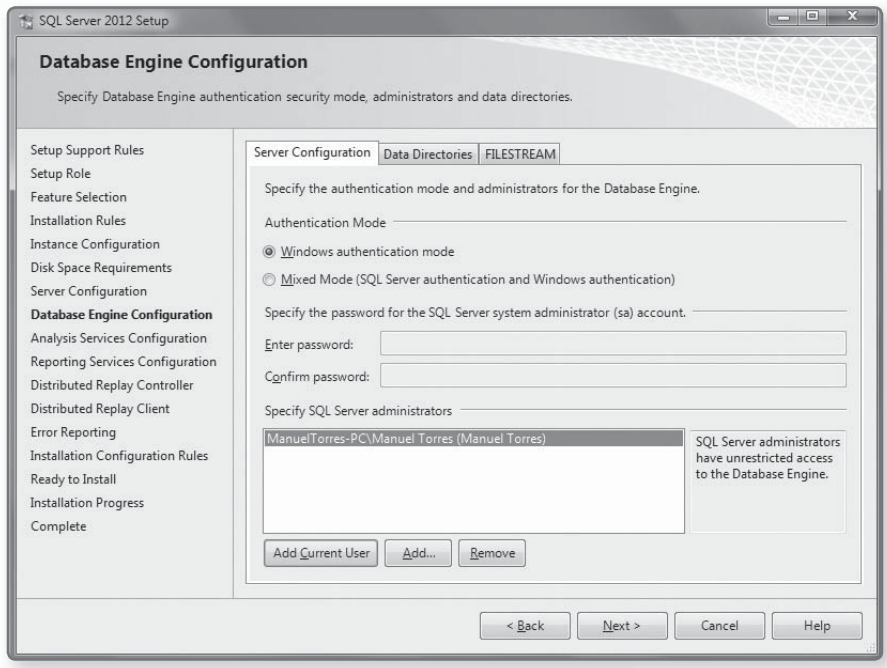
En la siguiente ventana sólo presionar **Next** puesto que el asistente de SQL Server 2012 determinó la forma de instalarlo.



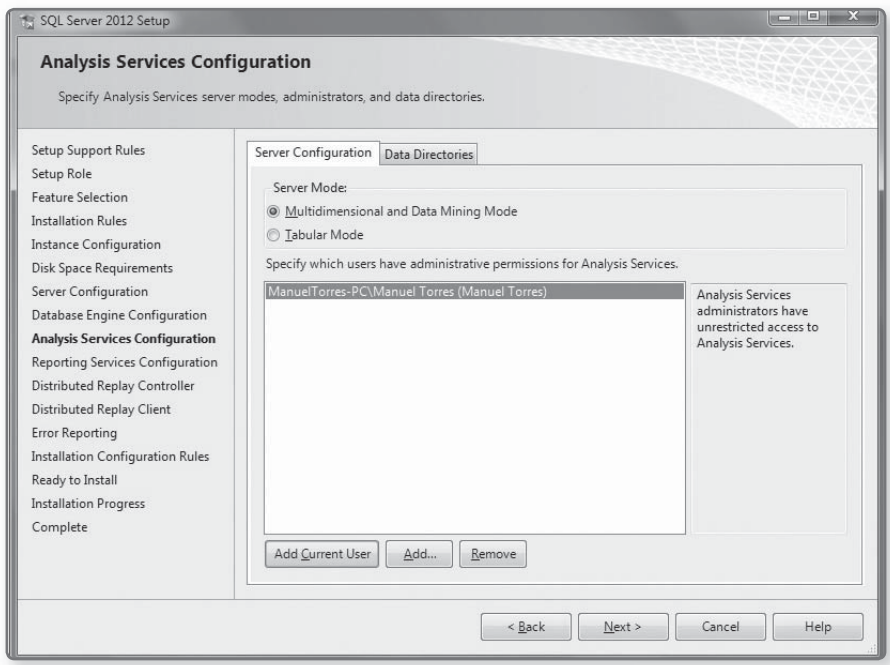
En la siguiente ventana se debe especificar el modo de autenticación que tendrá el acceso al Motor de base de datos SQL Server 2012, si está instalando en un computador personal o portátil se recomienda usar Windows Authentication Mode, si se encuentra en una organización y las bases pueden ser vulnerables entonces use Mixed Mode para poder definir una clave de acceso a los objetos del servidor.



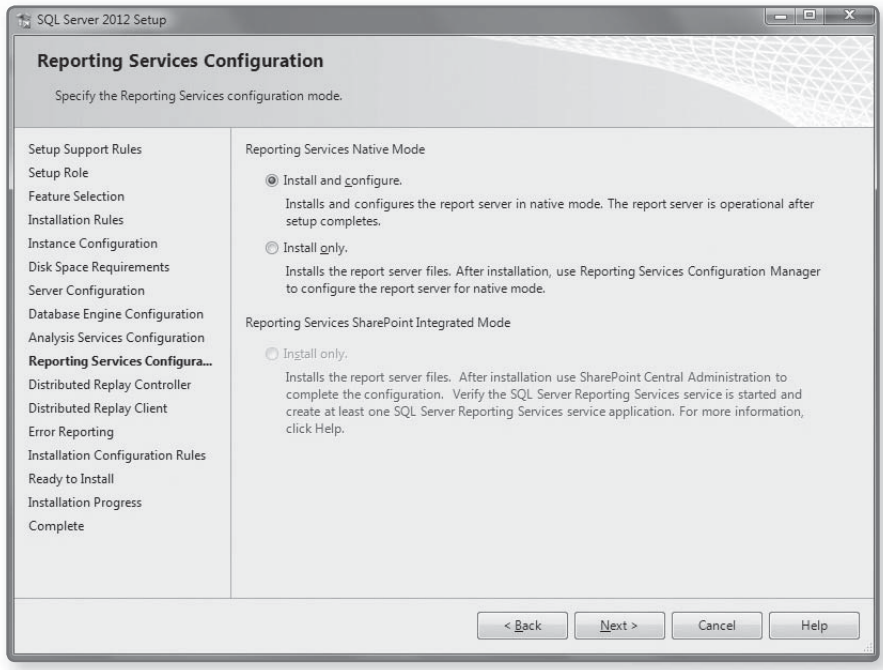
Antes de presionar **Next>** debe seleccionar **Add Current** User para determinar el usuario activo.



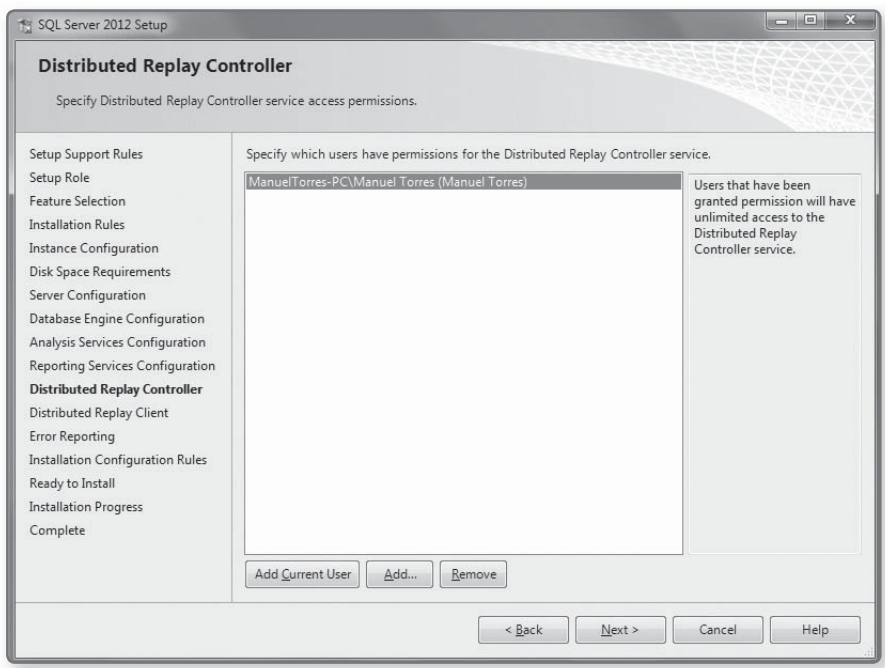
En la ventana **Analysis Services Configuration** también se debe agregar el usuario actual con **Add Current User**.



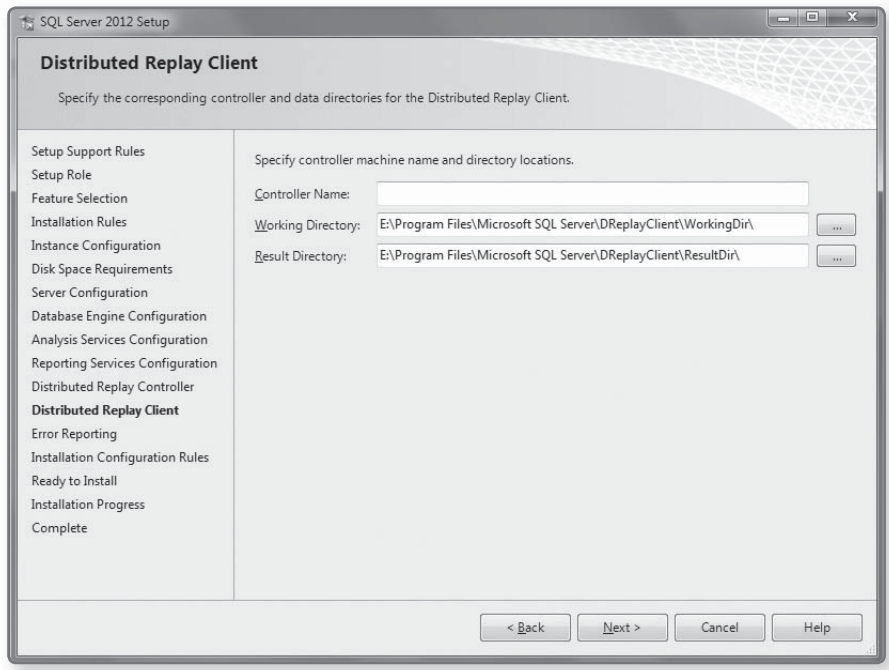
Luego en la ventana **Reporting Services Configuration** debe seleccionar **Install and Configure** para iniciar con la instalación del producto.



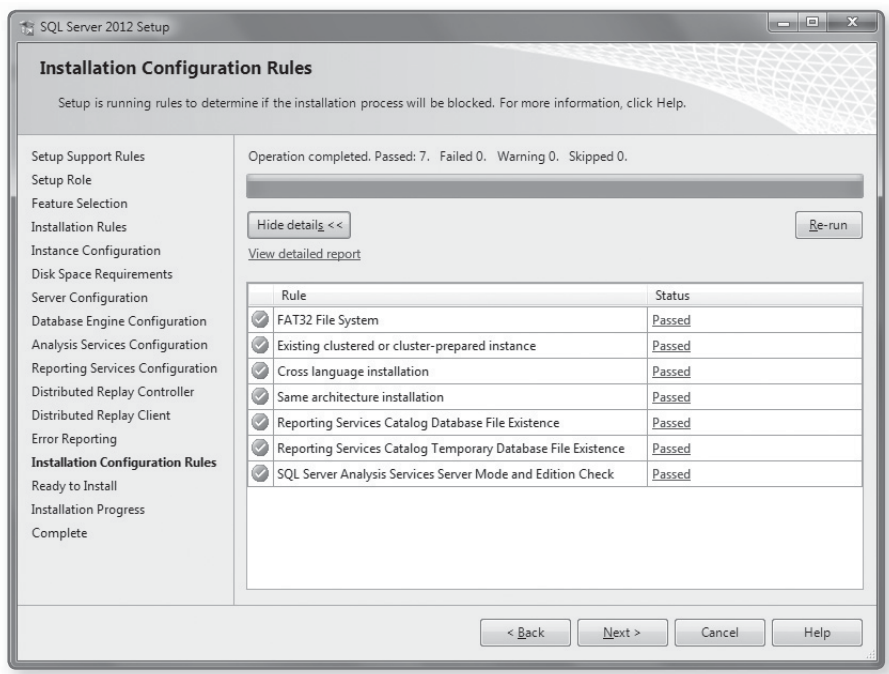
En la ventana **Distributed Replay Controller** debe seleccionar **Add Current** para seleccionar el usuario de distribución.



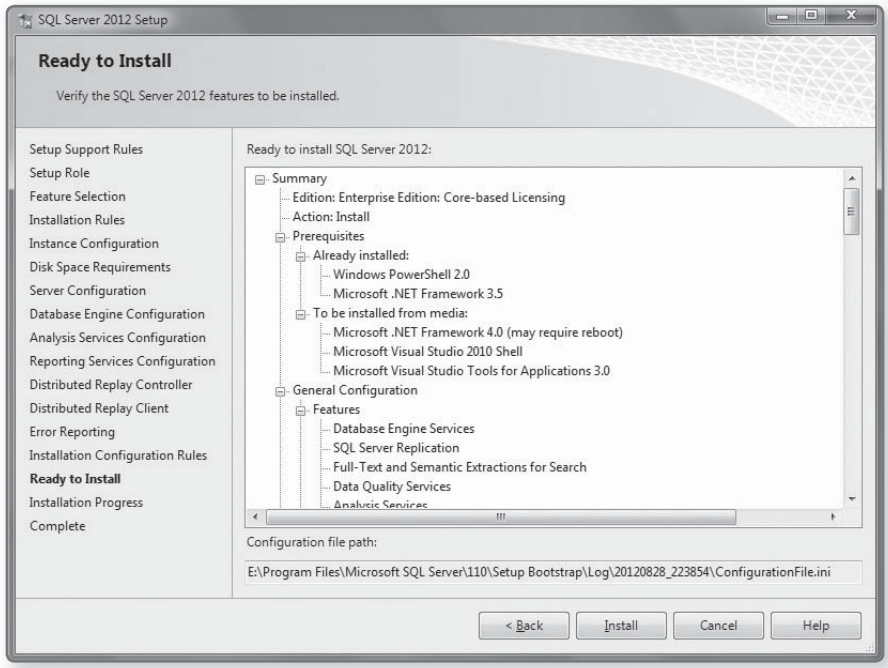
Seguidamente se debe especificar el directorios de los archivos de control, estos ya se encuentran predeterminados; por lo tanto, sólo presione **Next>**.



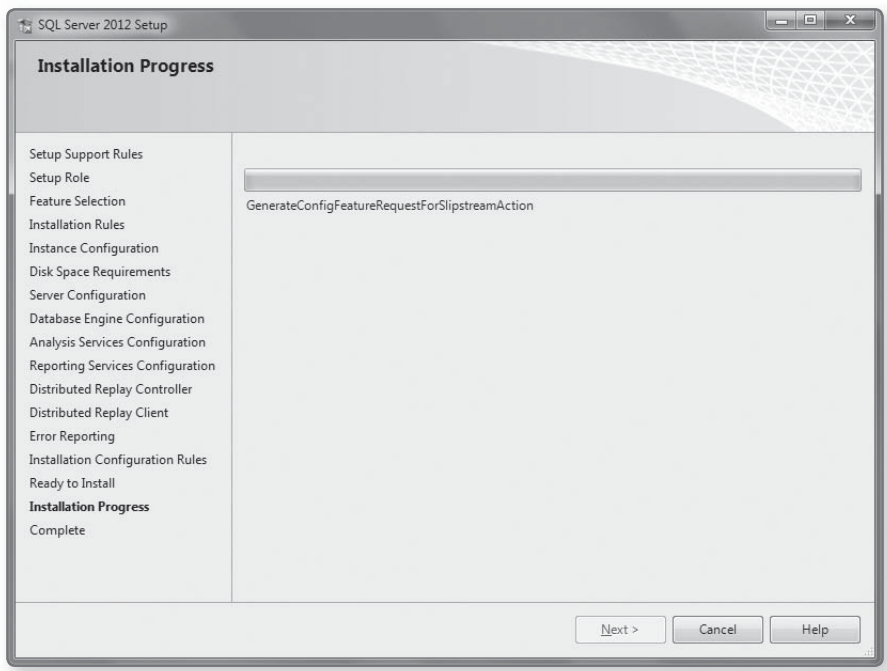
En la ventana Installation Configuration Rules, el asistente verificará si todo es correcto, usted podrá seleccionar **Next>** para comenzar con la instalación.



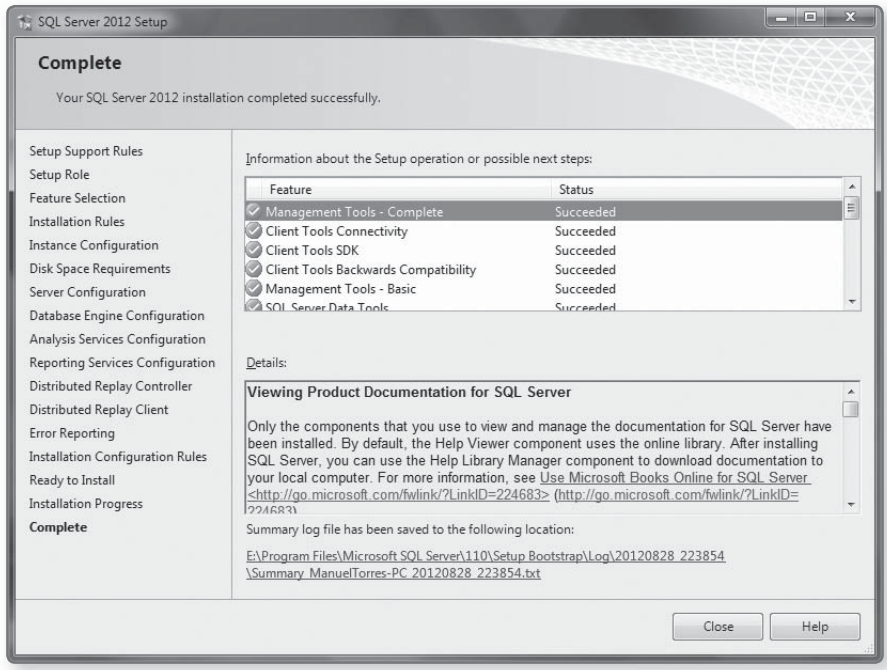
En la ventana **Ready to Install** estamos listos para iniciar la instalación, presionando el botón **Install**.



La instalación está en progreso.

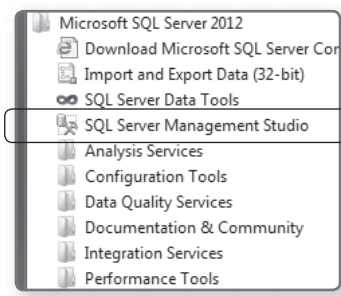


Finalmente, el asistente de instalación muestra la ventana de instalación completa, aquí sólo debe presionar **Close** para cerrar el asistente.

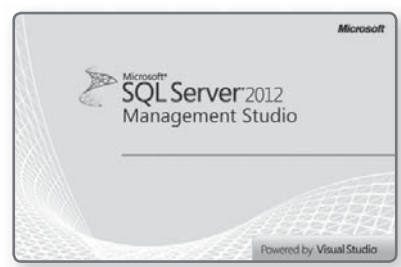


1.7. ACESSO AL SQL SERVER 2012

Desde el botón iniciar del Windows Seven debe seleccionar > Todos los programas > Microsoft SQL Server 2012 > SQL Server Management Studio.

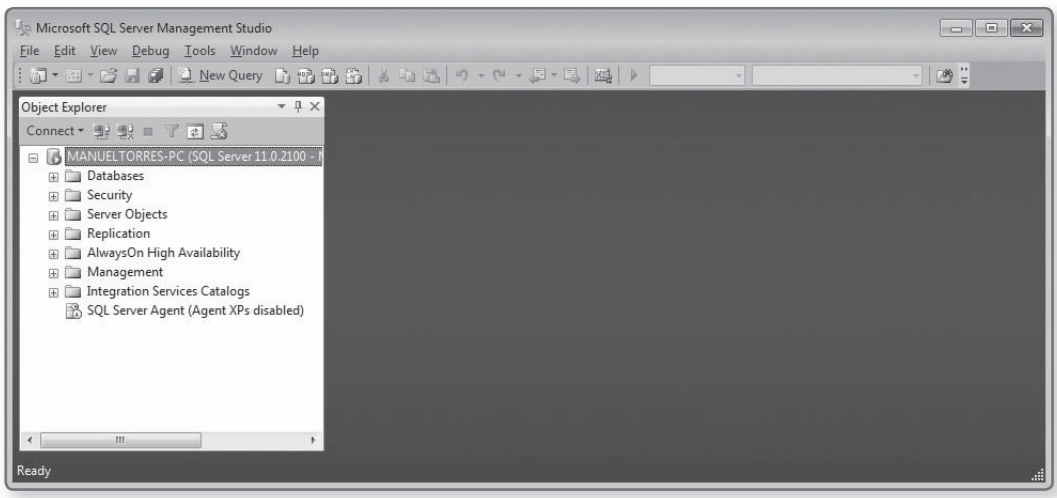


Al iniciar la aplicación se muestra la pantalla inicial de SQL Server 2012.

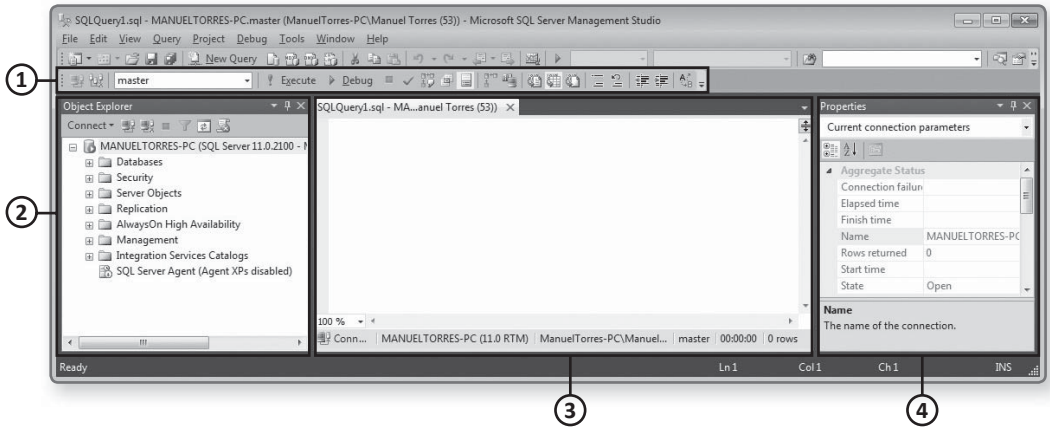


Seguidamente debemos seleccionar el nombre del servidor y el tipo de autenticación configurado en la ventana **Database Engine Configuration**.

A continuación se muestra el entorno de SQL Server 2012.



Después de seleccionar **New Query** se mostrará de la siguiente manera:



1. **Barra de Herramientas**: SQL Editor

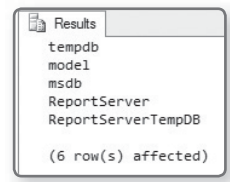
Desde aquí se muestran las bases de datos disponibles, también puede combinar las teclas CTRL+U.

Permite ejecutar un conjunto de instrucciones, también se puede presionar F5.

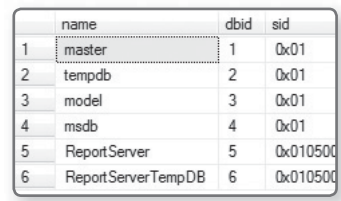
Permite verificar si el conjunto de instrucciones es correcto, pero no lo ejecuta.

El resultado de la ejecución de un conjunto de instrucciones puede tener tres entornos:

- **Texto(CTRL+T)**: se muestran los resultados parecidos a la salida por consola.



- **Grilla (CTRL+D)**: presenta los resultados en forma de cuadrículas, esta forma de mostrarse es la más común.



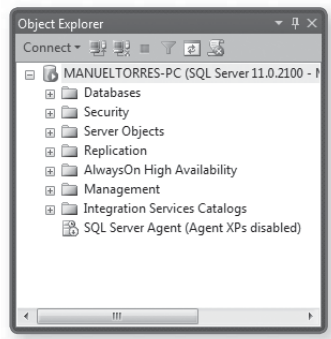
	name	dbid	sid
1	master	1	0x01
2	tempdb	2	0x01
3	model	3	0x01
4	msdb	4	0x01
5	ReportServer	5	0x010500
6	ReportServerTempDB	6	0x010500

- **Reporte**: permite grabar en forma de reporte los resultados obtenidos, la extensión de este archivo es RPT.

Permiten colocar y eliminar asignación de comentarios sobre instrucciones seleccionadas por el usuario.

2. **Panel Explorador de Objetos (F8)**

Desde aquí se podrá administrar los objetos del servidor como bases de datos, seguridad, objetos de servidor, etc. Hay que tener en cuenta que usted puede conectarse a varios servidores y administrarlos al mismo tiempo.



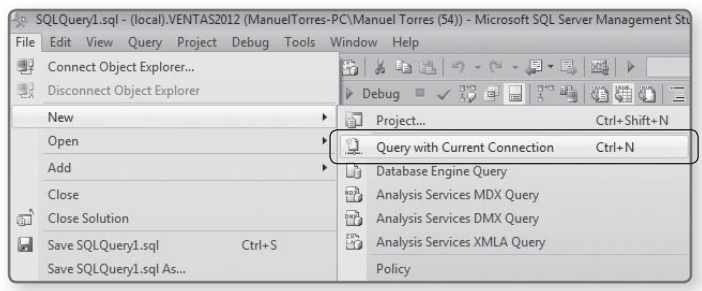
Veamos algunas opciones presentadas dentro del Explorador de Objetos:

- Si queremos conectarnos a un nuevo servidor debe presionar sobre el botón Connect > Database Engine...
- Si queremos visualizar las bases de datos del sistema debemos seleccionar Databases > System Databases…
- Si tuvieramos una base de datos llamada Ventas2012 como podriamos sus procedimientos almacenados: seleccionar Databases > Ventas2012 > Programmability > Stored Procedures.

3. **Entorno de desarrollo**

SQL Server se caracteriza por implementar script dentro del editor de codigo ya que desde aquí se podrá tener acceso a todos los objetos de una base. Todo esto gracias a los comandos que se pueda implementar desde versiones anteriores al SQL Server 2012 se viene utilizando la administración de ficheros eso quiere decir que se podrá implementar script desde dieferentes hojas del editor de consultas.

Para agregar una nueva hoja de edicion debe seleccionar el botón **New Query** desde la barra de herramientas o desde el menu File > New > Database Engine Query.



4. Panel de propiedades (F4)

Mientras desarrollemos script sobre las bases de datos no será necesario el manejo de las propiedades, así es que para nuestro caso cerraremos este panel.

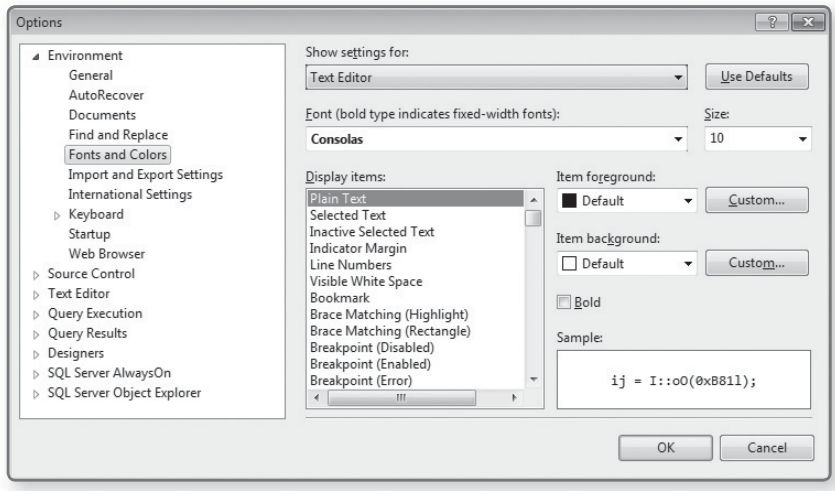
1.8. Configuración de fuente para el entorno de Trabajo

Transact-SQL se caracteriza por generar procesos en una base de datos por medio de script que se ejecutan dentro de un editor de consultas, en muchas ocasiones podemos confundir algunos operadores símbolos en los script; por ejemplo, la letra o con el número cero (O - 0) ambos nos pueden ocasionar errores lógicos cuando ejecutemos algún script; por lo tanto, se recomienda que use la fuente **Consolas** con un tamaño establecido por el usuario para el mejor desempeño de los script dentro de Transact-SQL, observemos el cambio entre la o y el cero (O - 0).

Acceso:

- Herramientas(Tools) > Opciones (Option)
- Seleccione Fonts and Colors
- Cambie el tipo de fuente en Font

- Presione OK



1.9. LENGUAJE DE DEFINICIÓN DE DATOS (LDD)

El lenguaje SQL está compuesto por comandos, cláusulas, operadores y funciones de agregado. Estos elementos se combinan en las instrucciones para crear, actualizar y manipular las bases de datos.

Existen dos (2) tipos de comandos SQL:

- Los comandos del Lenguaje de Definición de Datos (DDL) que permiten crear y definir nuevas bases de datos, campos e índices.
- Los comandos del Lenguaje de Manipulación de Datos (DML) que permiten modificar y generar consultas para insertar, modificar o eliminar, así como, ordenar, filtrar y extraer datos de la base de datos.

Para el desarrollo de los casos propuestos se propone el siguiente enunciado:

La ciudad de Lima está construyendo una línea metropolitana de transporte que une los principales distritos de la ciudad. Para lo cual cuenta con unas máquinas en cada estación que permiten comprar y recargar de dinero en unas tarjetas que permiten el acceso a estas líneas.

En cualquiera de los casos cuando el cliente introduce el dinero, el sistema debe, en primer lugar, identificar el tipo de billete, validar que es correcta la definición del mismo y procesar lo seleccionado por el cliente, este proceso acaba cuando la maquina emite una boleta y le entrega su cambio por dicho proceso.

1.10. SENTENCIA CREATE

COMANDO DE CREACION DE OBJETOS DE UNA BASE DE DATOS	CREATE

```
CREATE OBJETO NOMBREOBJETO (
   --Estructura del objeto
)
```

La sentencia **CREATE** permite crear base de datos, tablas, desencadenadores, procedimientos, funciones, vistas e índices de una base de datos.

CASO DESARROLLADO Nº 1.1

Implementar un script que permita crear la base de datos **METROPOLITANO** con valores estándar.

CREANDO UNA BASE DE DATOS CON VALORES ESTÁNDAR	COMANDO LDD CREATE DATABASE

```
CREATE DATABASE METROPOLITANO
```

En el script se implementa la creación de la base de datos Metropolitano en el servidor local. Para verificar los valores estándar se tiene que ejecutar el siguiente script:

```
SP_HELPDB METROPOLITANO
```

	name	db_size	owner	dbid	created	status	compatibility_level
1	METROPOLITANO	7.81 MB	ManuelTorres-PC\Manuel Torres	31	Aug 7 2012	Status=ONLINE, Updateability=READ_WRITE, UserAcc...	100

	name	fileid	filename	filegroup	size	maxsize	growth	usage
1	METROPOLITANO	1	E:\Program Files\Microsoft SQL Server\MSSQL10.MSS...	PRIMARY	2304 KB	Unlimited	1024 KB	data only
2	METROPOLITANO_log	2	E:\Program Files\Microsoft SQL Server\MSSQL10.MSS...	NULL	576 KB	2147483648 KB	10%	log only

CASO DESARROLLADO Nº 1.2

Implementar un script que permita crear la tabla **BOLETA** dentro de la base de datos **METROPOLITANO** con las siguientes características:

Boleta Nº 000000011515452

Fecha de Emisión: 06/08/2012
Monto: 10.00

LIMA-PERÚ

CREANDO UNA TABLA DE LA BASE DE DATOS METROPOLITANO	COMANDO LDD CREATE TABLE

```
CREATE TABLE BOLETA (
  NUMEROBOLETA    CHAR(15) NOT NULL,
  FECHAEMISION    DATE NOT NULL,
  MONTO           MONEY NOT NULL
)
GO
```

En el script se implementa la creación de la tabla **BOLETA** que permite definir la columnas Número de Boleta, Fecha de Emisión y el Monto asignado por el proceso. Como podrá ver el comando Create se comporta de acuerdo al objeto especificado en el script. Considere que para ejecutar el script de creación de tabla se deberá activar la base de datos para esto deberá ejecutar el siguiente script:

```
USE METROPOLITANO
GO
```

CASO DESARROLLADO Nº 1.3

Implementar un script que permita crear el **TRIGGER TX_MENSAJE** que muestre un mensaje al usuario cuando se realice una inserción o actualización a la tabla **BOLETA**.

CREANDO UNA TRIGGER A LA TABLA BOLETA	COMANDO LDD CREATE TRIGGER

```
CREATE TRIGGER TX_MENSAJE
ON BOLETA
FOR INSERT, UPDATE
AS
   PRINT 'LA BOLETA SE ACTUALIZO CORRECTAMENTE'
GO
```

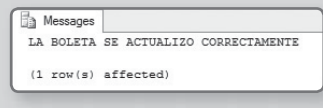
En el script se implementa la creación del desencadenador TX_MENSAJE que permite mostrar un mensaje cuando el usuario registra una Boleta nueva o cuando se actualiza algún campo de la tabla Boleta.

Para probar el Trigger debe agregar un Nuevo registro a la tabla Boleta con el siguiente script:

```
INSERT INTO BOLETA
   VALUES('00000013','07/08/2012',20.50)
```

o una actualización a la tabla Boleta con el siguiente script:

```
UPDATE BOLETA
   SET FECHAEMISION='07/08/2012',
       MONTO=20
   WHERE NUMEROBOLETA='00000013'
```



En ambos casos la imagen mostrada es la misma ya que al añadir o actualizar un registro el Trigger se activará.

CASO DESARROLLADO Nº 1.4

Implementar un script que permita crear el procedimiento almacenado **SP_TOTALBOLETAS** de la tabla **BOLETA**.

CREANDO UN PROCEDIMIENTO ALMACENADO A LA TABLA BOLETA	COMANDO LDD CREATE PROCEDURE

```
CREATE PROCEDURE SP_TOTALBOLETAS
AS
BEGIN
    SELECT  YEAR(FECHAEMISION) AS [AÑO],
                    COUNT(*) AS [TOTAL]
                    FROM BOLETA
                    GROUP BY YEAR(FECHAEMISION)
END
GO
```

En el script se implementa la creación del procedimiento almacenado **SP_TOTALBOLETAS** que permite contabilizar el total de boletas registradas por años. Para la ejecución del procedimiento almacenado deberá colocar el siguiente script **EXEC SP_TOTALBOLETAS** el resultado será:

	AÑO	TOTAL
1	2007	5
2	2008	7
3	2012	1

La imagen muestra el resumen de boletas por año, es decir, en el año 2007 se registró 5 boletas, en el 2008 7 boletas y en el 2012 1 boleta.

CASO DESARROLLADO Nº 1.5

Implementar un script que permita crear la función **FN_TOTALBOLETAS** de un determinado año ingresado por el usuario a la tabla **BOLETAS**.

CREANDO UNA FUNCIÓN ESCALAR A LA TABLA BOLETA	COMANDO LDD CREATE FUNCTION

```
CREATE FUNCTION FN_TOTALBOLETAS(@AÑO INT)
RETURNS INT
AS
BEGIN
     DECLARE @TOTAL INT
     SELECT  @TOTAL=COUNT(*)
                    FROM BOLETA
                    WHERE YEAR(FECHAEMISION)=@AÑO
                    GROUP BY YEAR(FECHAEMISION)
     RETURN @TOTAL
END
```

En el script se implementa la creación de la función **FN_TOTALBOLETAS** que de acuerdo a un año ingresado por el usuario deberá mostrar el total boletas registradas en dicho año. Para poder ejecutar la función puede tomar 2 formas mostradas a continuación:

Usando una consulta:

```
SELECT DBO.FN_TOTALBOLETAS(2007) AS [TOTAL DE BOLETAS]
```

	TOTAL DE BOLETAS
1	5

O usando el comando Print:

```
PRINT 'EL TOTAL DE BOLETAS ES: '+STR(DBO.FN_TOTALBOLETAS(2007))
```

```
EL TOTAL DE BOLETAS ES:          5
```

La diferencia entre ambos es sólo la forma en que muestran los resultados lo demás es idéntico como lo podrá visualizar en las imágenes de los comandos.

La función **STR** permite convertir a cadena un valor de otro tipo, en este caso la función devuelve un valor numérico y al tratar de imprimirlo con el comando **Print** emitirá un error si no lo convierte, a continuación se muestra el error ocasionado sino aplican la función **STR**.

```
Msg 245, Level 16, State 1, Line 1
Conversion failed when converting the varchar value 'EL TOTAL DE BOLETAS ES: ' to data type int.
```

CASO DESARROLLADO Nº 1.6

Implementar un script que permita crear la vista **VBOLETAS** donde muestre los registros contenidos en la tabla **BOLETAS**.

CREANDO UNA VISTA DE LA TABLA BOLETA	COMANDO LDD CREATE VIEW

```
CREATE VIEW VBOLETA
AS
     SELECT B.NUMEROBOLETA,B.FECHAEMISION,B.MONTO
            FROM BOLETA B
```

En el script se implementa la creación de la vista **VBOLETA** que permite listar los registros de la tabla Boleta, para poder ejecutar la vista debe colocar el siguiente script:

```
SELECT * FROM VBOLETA
```

	NUMEROBOLETA	FECHAEMISION	MONTO
1	00000001	2008-04-22	200.00
2	00000002	2008-01-02	350.00
3	00000003	2008-10-03	250.00
4	00000004	2007-05-01	150.00
5	00000005	2008-12-07	200.00
6	00000006	2008-08-16	120.00
7	00000007	2007-01-03	50.00

Como notará es muy parecido a realizar una consulta a la tabla **BOLETA**, en ocasiones los lenguajes de programación solicitan vistas para mostrar consultas.

1.11. SENTENCIA ALTER

COMANDO DE MODIFICACIÓN DE OBJETOS DE UNA BASE DE DATOS	ALTER

```
ALTER OBJETO NOMBREOBJETO (
   --Datos modificados
)
```

La sentencia **ALTER** permite la modificación de un objeto asociado a una base de datos, puede modificar archivos, grupo de archivos, cambiar atributos de un objeto.

CASO DESARROLLADO Nº 1.7

Implementar un script que permita agregar un archivo secundario a la base de datos **METROPOLITANO** llamado **METROPOLITANO_SEC2** de tamaño inicial 5MB y un tamaño máximo de 10MB con un factor de crecimiento de 2%.

AGREGANDO UN ARCHIVO SECUNDARIO A LA BASE DE DATOS	COMANDO LDD ALTER DATABASE

```
ALTER DATABASE METROPOLITANO
ADD FILE (
  NAME='METROPOLITANO_SEC2',
  FILENAME='E:\Program Files\Microsoft SQL Server\MSSQL10.MSSQLSERVER\
                                        MSSQL\DATA\METROPOLITANO_SEC2.NDF',
  SIZE=5MB,
  MAXSIZE=10MB,
  FILEGROWTH=2%
)
```

En el script se agrega un archivo secundario a la base de datos METROPOLITANO con las características solicitadas en el caso. Hay que comprobar los archivos que contaba dicha base de datos, con el siguiente comando:

```
SP_HELPDB METROPOLITANO
```

Note: en la imagen siguiente la base de datos estándar tiene solo 2 archivos:

	name	db_size	owner	dbid	created	status	compatibility_level
1	METROPOLITANO	2.81 MB	ManuelTorres-PC\Manuel Torres	31	Aug 7 2012	Status=ONLINE, Updateability=READ_WRITE, UserAcc...	100

	name	fileid	filename	filegroup	size	maxsize	growth	usage
1	METROPOLITANO	1	E:\Program Files\Microsoft SQL Server\MSSQL10.MSSQLSERV...	PRIMARY	2304 KB	Unlimited	1024 KB	data only
2	METROPOLITANO_log	2	E:\Program Files\Microsoft SQL Server\MSSQL10.MSSQLSERV...	NULL	576 KB	2147483648 KB	10%	log only

Y al agregar un archivo secundario se vería de la siguiente forma:

	name	db_size	owner	dbid	created	status	compatibility_level
1	METROPOLITANO	7.81 MB	ManuelTorres-PC\Manuel Torres	31	Aug 7 2012	Status=ONLINE, Updateability=READ_WRITE, UserAcc...	100

	name	fileid	filename	filegroup	size	maxsize	growth	usage
1	METROPOLITANO	1	E:\Program Files\Microsoft SQL Server\MSSQL10.MSS...	PRIMARY	2304 KB	Unlimited	1024 KB	data only
2	METROPOLITANO_log	2	E:\Program Files\Microsoft SQL Server\MSSQL10.MSS...	NULL	576 KB	2147483648 KB	10%	log only
3	METROPOLITANO_SEC2	3	E:\Program Files\Microsoft SQL Server\MSSQL10.MSS...	PRIMARY	5120 KB	10240 KB	2%	data only

CASO DESARROLLADO Nº 1.8

Implementar un script que permita modificar la precisión de la columna Número de Boleta asignada inicialmente con el valor 15, modificarlo por 20 de la tabla Boleta:

MODIFICAR EL VALOR DE LA COLUMNA EN UNA TABLA	COMANDO LDD ALTER TABLE

```
--1. Verificar las columnas de la tabla Boleta
SP_COLUMNS BOLETA
GO

--2. Modificar la columna Numero de Boleta
ALTER TABLE BOLETA
    ALTER COLUMN NUMEROBOLETA CHAR(20) NOT NULL
GO
```

En el punto uno se verifica las precisiones de las columnas inicialmente definidas, la imagen siguiente muestra que la precisión de la columna **numeroBoleta** es 15.

	TABLE_QUALIFIER	TABLE_OWNER	TABLE_NAME	COLUMN_NAME	DATA_TYPE	TYPE_NAME	PRECISION	LENGTH
1	METROPOLITANO	dbo	BOLETA	numeroBoleta	1	char	15	15
2	METROPOLITANO	dbo	BOLETA	fechaEmision	-9	date	10	20
3	METROPOLITANO	dbo	BOLETA	monto	3	money	19	21

Luego de modificar la columna numeroBoleta con la instrucción **ALTER TABLE** usted podrá notar el cambio en la precisión ejecutando nuevamente la instrucción **SP_COLUMNS** así lo muestra la siguiente imagen:

	TABLE_QUALIFIER	TABLE_OWNER	TABLE_NAME	COLUMN_NAME	DATA_TYPE	TYPE_NAME	PRECISION	LENGTH
1	METROPOLITANO	dbo	BOLETA	numeroBoleta	1	char	20	20
2	METROPOLITANO	dbo	BOLETA	fechaEmision	-9	date	10	20
3	METROPOLITANO	dbo	BOLETA	monto	3	money	19	21

CASO DESARROLLADO Nº 1.9

Implementar un script que permita modificar el **TRIGGER TX_MENSAJE** que adicione al mensaje anterior a la fecha actual sólo cuando se inserta o actualiza la tabla **BOLETA**.

MODIFICAR UN TRIGGER	COMANDO LDD ALTER TRIGGER

```
ALTER TRIGGER TX_MENSAJE
ON BOLETA
FOR INSERT, UPDATE
AS
   PRINT 'LA BOLETA SE ACTUALIZO CORRECTAMENTE'
   PRINT 'FECHA'+CAST(GETDATE() AS VARCHAR(10))
GO
```

En el script se implementa la modificación del desencadenador **TX_MENSAJE** que permite mostrar un mensaje cuando el usuario registra una Boleta nueva o cuando se actualiza algún campo de la tabla Boleta además de mostrar la fecha de dicha actualización.

La función CAST permite la conversión de cualquier tipo y cuenta con el siguiente formato:

```
CAST(VALOR AS TIPODATOS)
```

La diferencia entre **STR** y **CAST** es que el primero convierte directamente a cadena y por tanto imprime espacios en el lado izquierdo de la conversión. CAST convierte a cualquier tipo de datos y no incorpora espacios en blanco en ninguno de los lados.

Para probar el Trigger debe agregar un nuevo registro a la tabla Boleta con el siguiente código:

```
INSERT INTO BOLETA
   VALUES('000000011515453','07/08/2012',20.50)
```

o una actualización a la tabla Boleta con el siguiente código:

```
UPDATE BOLETA
   SET FECHAEMISION='07/08/2012',
       MONTO=20.50
   WHERE NUMEROBOLETA
```

```
LA BOLETA SE ACTUALIZO CORRECTAMENTE
FECHA: Aug 10 2012  5:41PM

(1 row(s) affected)
```

CASO DESARROLLADO Nº 1.10

Implementar un script que permita modificar el procedimiento almacenado **SP_TOTALBOLETAS** de la tabla **BOLETA**, donde muestre la columna MES agrupando el número de boletas por cada mes y al final mostrar el total de boletas registradas.

MODIFICANDO UN PROCEDIMIENTO ALMACENADO	COMANDO LDD ALTER PROCEDURE

```
ALTER PROCEDURE SP_TOTALBOLETAS
AS
BEGIN
    SELECT  YEAR(FECHAEMISION) AS [AÑO],
                    MONTH(FECHAEMISION) AS [MES],
                    COUNT(*) AS [TOTAL]
                    FROM BOLETA
                    GROUP BY ROLLUP(YEAR(FECHAEMISION),MONTH(FECHAEMISION))

END
GO
```

En el script se implementa la modificación del procedimiento almacenado donde se muestran el total de boletas según el año y un mes respectivo y al final de cada cambio se calcula el total de boletas por año, note que en la fila 6 se muestra 2007 NULL 5, esto significa que el total de boletas registradas en el año 2007 son 5 lo mismo se aplica al año 2008 en la fila 13 y en el año 2012 en la fila 15, tenga en cuenta que la fila 16 sólo muestra el consolidado de boletas registradas que será el resultado del total de boletas de los años 2007 (5 boletas), 2008 (7 boletas) y 2012 (2 boletas).

	AÑO	MES	TOTAL
1	2007	1	1
2	2007	3	1
3	2007	5	1
4	2007	10	1
5	2007	11	1
6	2007	NULL	5
7	2008	1	1
8	2008	4	1
9	2008	8	1
10	2008	9	1
11	2008	10	1
12	2008	12	2
13	2008	NULL	7
14	2012	8	2
15	2012	NULL	2
16	NULL	NULL	14

Finalmente, observe la fila 14, la interpretación sería que en el mes de agosto del año 2012 se registraron 2 boletas y; por lo tanto, el total de boletas en ese año sería también 2, el cual es mostrado en la fila 15.

CASO DESARROLLADO Nº 1.11

Implementar un script que permita modificar la función **FN_TOTALBOLETAS** en la cual se permita añadir el parámetro mes y muestre el total de boletas según el año y mes ingresados a la tabla **BOLETAS**.

MODIFICANDO UNA FUNCIÓN ESCALAR	COMANDO LDD ALTER FUNCTION

```
ALTER FUNCTION FN_TOTALBOLETAS(@AÑO INT,@MES INT)
RETURNS INT
AS
BEGIN
    DECLARE @TOTAL INT
    SELECT  @TOTAL=COUNT(*)
            FROM BOLETA
            WHERE YEAR(FECHAEMISION)=@AÑO AND MONTH(FECHAEMISION)=@MES
            GROUP BY YEAR(FECHAEMISION)
    RETURN @TOTAL
END
```

En el script se implementa la modificación de la función **FN_TOTALBOLETAS** que de acuerdo a un mes y año ingresado por el usuario debe mostrar el total de boletas registradas en dichos parámetros. Para poder ejecutar la función puede tomar 2 formas mostradas a continuación:

Usando una consulta:

```
SELECT dbo.FN_TOTALBOLETAS(2012,3) AS [TOTAL DE BOLETAS]
```

	TOTAL DE BOLETAS
1	4

O usando el comando Print:

```
DECLARE @AÑO INT=2012,@MES INT=3
PRINT 'EL TOTAL DE BOLETAS DEL MES '+CAST(@MES AS CHAR(2))+
            'DEL AÑO '+CAST(@AÑO AS CHAR(4))+
            ' ES: ' + CAST(dbo.FN_TOTALBOLETAS(@AÑO,@MES) AS CHAR(4))
```

```
EL TOTAL DE BOLETAS DEL MES 3 DEL AÑO 2012 ES: 4
```

CASO DESARROLLADO Nº 1.12

Implementar un script que permita modificar la vista **VBOLETAS** donde muestre sólo las columnas Número de Boleta y Fecha de Emisión de la tabla **BOLETAS**, además de encriptar la información de la implementación.

MODIFICANDO UNA VISTA	COMANDO LDD ALTER VIEW

```
ALTER VIEW VBOLETA
WITH ENCRYPTION
AS
    SELECT B.NUMEROBOLETA,B.FECHAEMISION
           FROM BOLETA B
```

En el script se implementa la modificación de la vista **VBOLETA** que solo muestra las columnas Numero de Boleta y Fecha de Emisión además de encriptarlo:

```
SELECT * FROM VBOLETA
```

Para visualizar la implementación de la vista VBOLETA se debe colocar el siguiente script:

```
SP_HELPTEXT VBOLETA
```

La activación de la encriptación dentro de la vista no permite visualizar el script que implementa dicha vista, esta es una forma de proteger el script mostrando el siguiente mensaje:

```
The text for object 'VBOLETA' is encrypted.
```

```
SP_HELPTEXT VBOLETA
```

Si en la implementación de la vista no se coloca la cláusula **WITH ENCRYPTION** entonces al visualizar el script se mostrara:

	Text
1	CREATE VIEW VBOLETA
2	AS
3	SELECT B.NUMEROBOLETA,B.FECHAEMISION
4	FROM BOLETA B

1.12. SENTENCIA DROP

COMANDO DE ELIMINACIÓN DE OBJETOS DE UNA BASE DE DATOS	DROP

```
DROP OBJETO NOMBREOBJETO
```

La sentencia **DROP** permite la eliminación de un objeto asociado a una base de datos.

CASO DESARROLLADO Nº 1.13

Implementar un script que permita eliminar la base de datos **METROPOLITANO**.

ELIMINANDO UNA BASE DE DATOS	COMANDO LDD DROP DATABASE

```
USE MASTER
GO

DROP DATABASE METROPOLITANO
GO
```

En el script se implementa la eliminación de la base de datos METROPOLITANO, para este caso la base de datos no debe estar en uso ya que ocurrirá un error y mostrará el siguiente mensaje:

```
Msg 3702, Level 16, State 3, Line 1
Cannot drop database "METROPOLITANO" because it is currently in use.
```

Para que la base de datos METROPOLITANO no sea la base activa se tiene que activar la base de datos maestra con el siguiente script **USE MASTER**.

CASO DESARROLLADO Nº 1.14

Implementar un script que permita eliminar la tabla **BOLETA** de la base de datos **METROPOLITANO**.

ELIMINANDO UNA TABLA	COMANDO LDD DROP TABLE

```
DROP TABLE BOLETA
GO
```

En el script se implementa la eliminación de la tabla BOLETA asociada a la base de datos METROPOLITANO, tenga en cuenta que al eliminar una tabla también se eliminan los datos contenidos en la tabla, índices, triggers, constrains y permisos especificados en la tabla.

CASO DESARROLLADO Nº 1.15

Implementar un script que permita eliminar el trigger **TX_MENSAJE** asociado a la tabla BOLETA de la base de datos **METROPOLITANO**.

ELIMINANDO UN TRIGGER	COMANDO LDD DROP TRIGGER

```
DROP TRIGGER TX_MENSAJE
GO
```

En el script se implementa la eliminación del trigger TX_MENSAJE, la eliminación de un trigger obedece cuando ya no necesite estar asociado a una tabla, porque SQL permite habilitar e inhabilitar un trigger con la sentencia **Disable**.

Impreso en los Talleres Gráficos de

Surquillo
☎ 719-9700