Práctica 11. Creación de un contenedor Docker con SQL Server Express

ASIR / DAW - (Gestión de) Bases de datos

JOSÉ JUAN SÁNCHEZ HERNÁNDEZ

IES Celia Viñas (Almería) - 2022/2023

Índice general

Índice de cuadros

Índice de figuras

Capítulo 1

Creación de un contenedor Docker con SQL Server Express

1.1 ¿Qué es SQL Server?

SQL Server es un sistema de gestión de bases de datos relacionales desarrollado por la empresa Microsoft.

Actualmente existen diferentes versiones para una infraestructura **On-Premise**.

- **Enterprise**. Es la versión más completa de todas.
- **Developer**. Es una edición gratuita que incluye todas las características que se puede usar como base de datos de desarrollo y pruebas en un entorno que no sea de producción.
- **Standard**. Es una versión limitada destinada a servidores con menos recursos.
- **Express**. Es una edición gratuita ideal para el desarrollo y la producción de aplicaciones de escritorio, aplicaciones web y pequeñas aplicaciones de servidor.

También existe una solución para infraestructura **Cloud**.

- **SQL Server en Azure**. Es la versión que está disponible en la nube pública de Azure, donde el usuario sólo paga por el uso del servicio.

1.2 Requisitos

Para poder ejecutar contenedores Docker es necesario tener instalado Docker Community Edition (CE) en nuestro equipo.

En la web oficial encontrará la información necesaria para realizar la instalación de Docker CE sobre Windows, macOS, Ubuntu, Debian, Fedora y CentOS.

1.3 Cómo crear un contenedor SQL Server 2019 Express sin persistencia de datos

Un contenedor Docker que no tiene persistencia de datos quiere decir que cuando finalice la ejecución perderá todo el contenido que hayamos creado durante la ejecución. Es decir, si durante la ejecución del contenedor hemos creado varias bases de datos en SQL Server Express, éstas se perderán cuando el contenedor se detenga.

El comando que podríamos usar para lanzar nuestro contenedor Docker con SQL Server 2019 Express sin persistencia de datos podría ser el siguiente:

```
docker run -d --rm --name sql_server_2019 -e ACCEPT_EULA=Y -e SA_PASSWORD='
    yourStrong(!)Password123' -e MSSQL_PID='Express' -p 1433:1433 mcr.microsoft.com
    /mssql/server:2019-latest
```

- `docker run` es el comando que nos permite crear un contenedor a partir de una imagen Docker.
- El parámetro `-d` nos permite ejecutar el contenedor en modo *detached*, es decir, ejecutándose en segundo plano.
- El parámetro `--rm` hace que cuando salgamos del contenedor, éste se elimine y no ocupe espacio en nuestro disco.
- El parámetro `--name` nos permite asignarle un nombre a nuestro contenedor. Si no le asignamos un nombre Docker nos asignará un nombre automáticamente.
- El parámetro `-e` es para pasarle al contenedor variables de entorno. En este caso le estamos pasando tres variables de entorno `ACCEPT_EULA`, `SA_PASSWORD` y `MSSQL_PID`.
  - A la variable `ACCEPT_EULA` le estamos pasando el parámetro `Y` para indicarle que aceptamos los términos de uso de la licencia.
  - La variable `SA_PASSWORD` contiene el valor de la contraseña del administrador del sistema (userid = «sa»). Debe tener en cuenta que la contraseña debe tener al menos 8 caracteres y debe incluir caracteres en mayúscula, minúscula y números.
  - La variable `MSSQL_PID` indica el *Product ID* (PID) o la edición del contenedor que queremos ejecutar. En nuestro caso utilizarmos la edición `Express`, pero es posible utilizar otras ediciones. Si no se indica ningún valor en la variable `MSSQL_PID` utilizará por defecto la opción `Developer`. La lista de posibles valores que podemos encontrar en la documentación oficial.
- El parámetro `-p` nos permite mapear los puertos entre nuestra máquina local y el contenedor. En este caso, estamos mapeando el puerto `1433` de nuestra máquina local con el puerto `1433` del contenedor.
- `mcr.microsoft.com/mssql/server:2019-latest` es el nombre de la imagen y la versión que vamos a utilizar para crear el contenedor. Si no se indica lo contrario buscará las imágenes en el repositorio oficial Docker Hub.

1.4 Cómo comprobar que el contenedor está en ejecución

Una vez que hemos iniciado el contenedor podemos comprobar que se está ejecutando con el siguiente comando:

```
docker ps
```

Deberíamos obtener una salida similar a esta.

```
CONTAINER ID        IMAGE                                  COMMAND
                    CREATED             STATUS              PORTS
                      NAMES
c370b426a379        microsoft/mssql-server-linux:latest    "/opt/mssql/bin/…sqls"
      3 seconds ago       Up 1 second         0.0.0.0:1433->1433/tcp
    sql_server_2019
```

1.5 Cómo conectarnos al contenedor desde la línea de comandos con la utilidad `sqlcmd`

Una vez que hemos creado la instancia del contenedor que está ejecutando SQL Server 2019 Express, podemos conectarnos a él para utilizar la herramienta `sqlcmd`.

Para conectarnos al contenedor en primer lugar tenemos que conocer cuál es su ID. Para obtenerlo podemos hacer uso del comando `docker ps`.

```
docker ps

CONTAINER ID        IMAGE                                  COMMAND
                    CREATED             STATUS              PORTS
                      NAMES
c370b426a379        microsoft/mssql-server-linux:latest    "/opt/mssql/bin/…sqls"
      3 seconds ago       Up 1 second         0.0.0.0:1433->1433/tcp
    sql_server_2019
```

En la primera columna podemos ver cuál es el `CONTAINER ID`. Una vez localizado el identificador ejecutamos el siguiente comando para conectarnos a él con la herramienta `sqlcmd`.

```
docker exec -it <container_id|container_name> /opt/mssql-tools/bin/sqlcmd -S
    localhost -U sa -P yourStrong(!)Password123
```

Ejemplo:

```
docker exec -it c370b426a379 /opt/mssql-tools/bin/sqlcmd -S localhost -U sa -P
    yourStrong(!)Password123
```

Donde:

- El parámetro `c370b426a379` indica el identificador del contenedor.
- El parámetro `/opt/mssql-tools/bin/sqlcmd` es el comando que vamos a ejecutar en el contenedor que estamos creando.
- El parámetro `-S localhost` indica el servidor al que queremos conectarnos, que en este caso se está ejecutando en la misma máquina desde la que estamos lanzando este comando.
- El parámetro `-U sa` indica que vamos a conectarnos con el usuario `sa`.
- El parámetro `-P yourStrong(!)Password123` indica el password del usuario con el que nos vamos a conectar.

1.6 Cómo detener el contenedor

Para detener el contenedor en primer lugar tenemos que conocer cuál es su ID. Para obtenerlo podemos hacer uso del comando `docker ps`.

```
docker ps

CONTAINER ID        IMAGE                                  COMMAND
                     CREATED             STATUS              PORTS
                       NAMES
c370b426a379        microsoft/mssql-server-linux:latest    "/opt/mssql/bin/…sqls"
      3 seconds ago       Up 1 second         0.0.0.0:1433->1433/tcp
    sql_server_2019
```

En la primera columna podemos ver cuál es el `CONTAINER ID`. Una vez localizado el identificador ejecutamos el comando `docker stop` y le pasamos como parámetro el identificador del contenedor que queremos detener.

Para el caso anterior deberíamos ejecutar:

```
docker stop c370b426a379
```

1.7 Herramientas recomendadas para trabajar con SQL Server

A continuación se muestran algunas de las herramientas que Microsoft ha desarrollado para trabajar con SQL Server. Puede encontrar un listado más completo de herramientas en la web oficial.

Herramientas con interfaz de usuario (GUI)

- Azure Data Studio (Windows, macOS, Linux).
- SQL Server Management Studio (SSMS) (Windows).
- SQL Server Data Tools (SSDT) (Windows).
- Visual Studio Code con la extensión `mssql` (Windows, macOS, Linux).

Herramientas para la línea de comandos

- `sqlcmd` (Windows, macOS, Linux).
- SQL Server PowerShell. (Windows, macOS, Linux).

Herramientas para realizar migraciones y otras tareas

- Data Migration Assistant.
- SQL Server Migration Assistant para Oracle.

1.8 ¿Qué es Azure Data Studio?

Azure Data Studio es una herramienta multiplataforma para trabajar con bases de datos que usan plataformas de datos locales y en la nube. Está disponible para Windows, macOS y Linux.

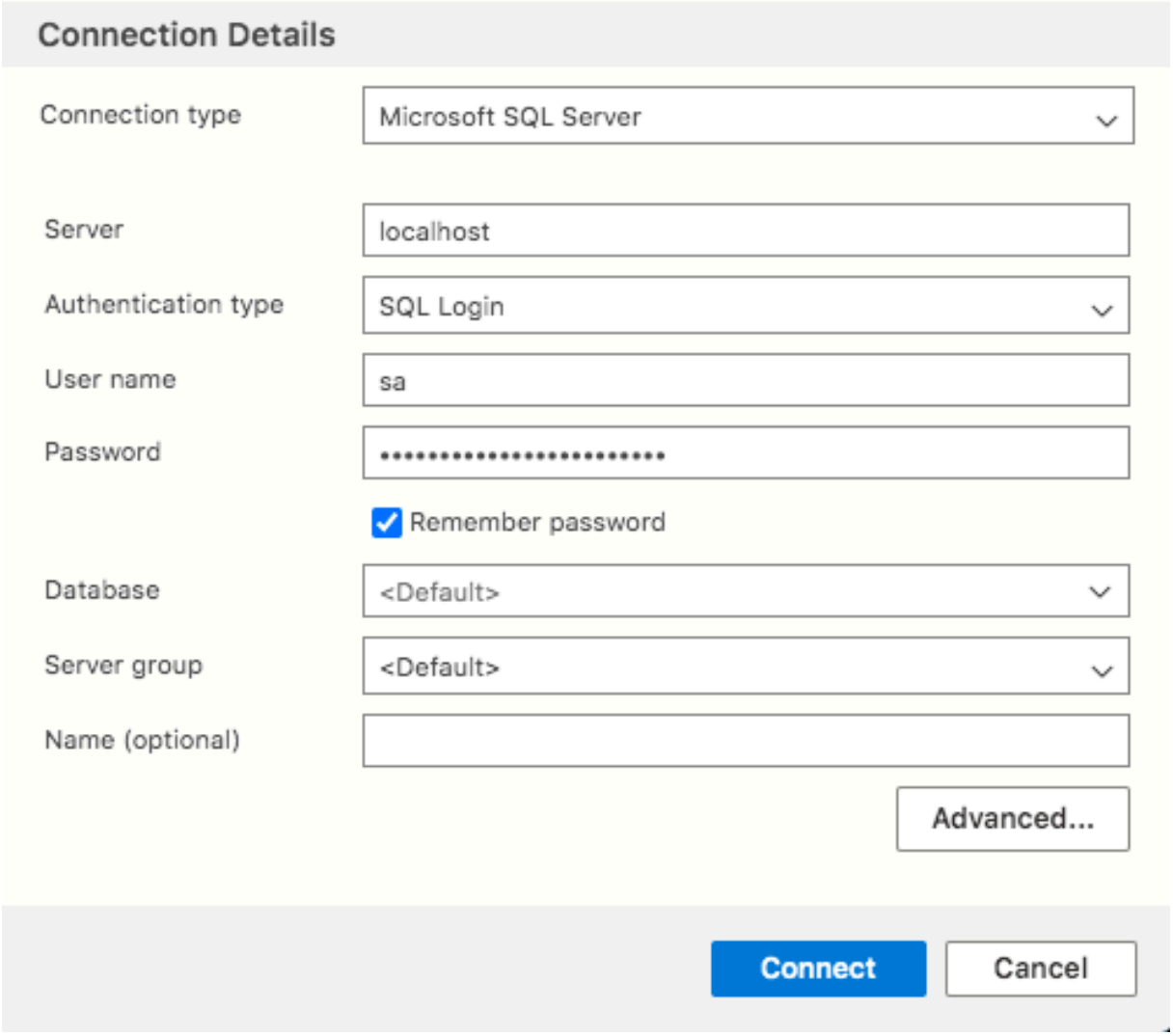
Más información en la documentación oficial.

1.9 Descargar e instalar Azure Data Studio

Más información en la documentación oficial.

1.10 Configurar Azure Data Studio para conectar con SQL Server

A continuación se muestra un ejemplo de cómo sería la configuración de Azure Data Studio para poder conectar con la instancia de SQL Server que se está ejecutando en nuestra máquina.



1.11 Referencias

- Web oficial de SQL Server
- Imagen de SQL Server Express en Docker Hub
- Microsoft SQL Server. Wikipedia.
- Web oficial de Docker
- Instalación de Docker Community Edition (CE)
- Instalación de Docker Community Edition (CE) en Windows
- Instalación de Docker Community Edition (CE) en macOS
- Instalación de Docker Community Edition (CE) en Ubuntu
- Instalación de Docker Community Edition (CE) en Debian
- Instalación de Docker Community Edition (CE) en Fedora
- Instalación de Docker Community Edition (CE) en CentOS
- Docker Hub

Capítulo 2

Licencia