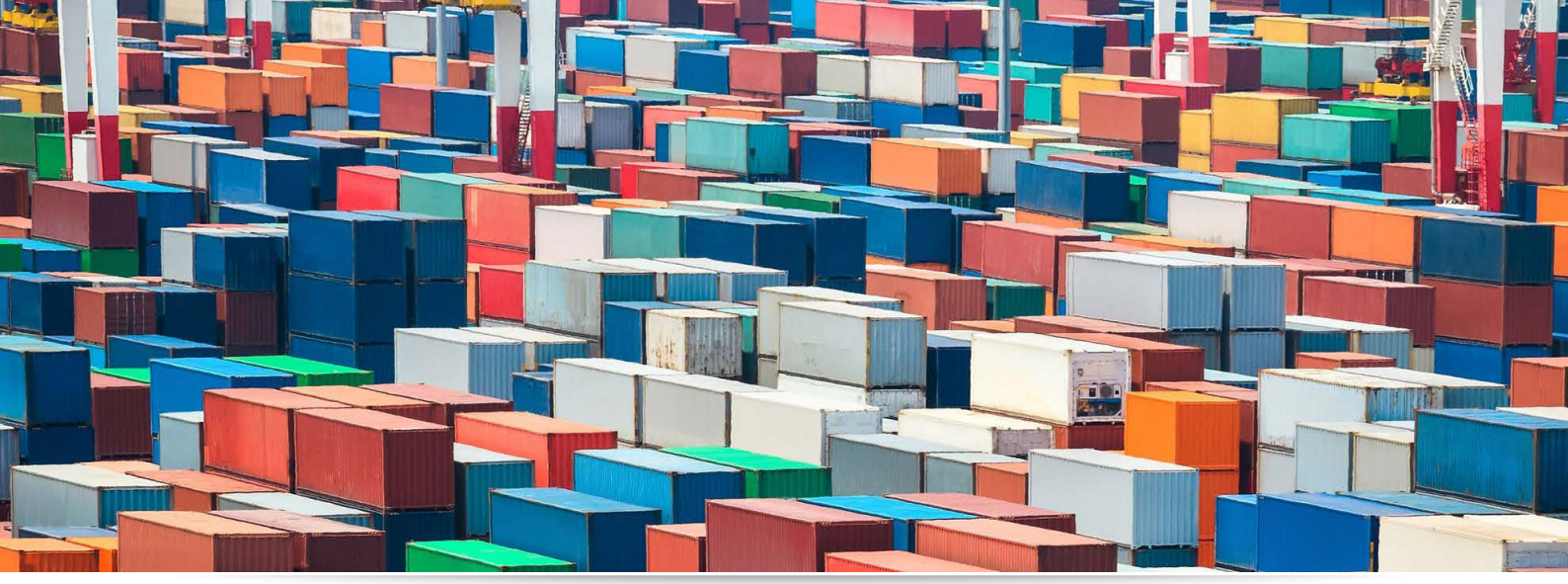

Daten-Virtualisierung in Oracle 12c mit Docker-Containern

Franck Pachot, dbi-services

In diesem Jahr dreht sich alles um die Daten-Virtualisierung. Die Zeiten, in denen der Datenbank-Speicher für jede Preproduction, jeden Test, jede Entwicklung etc. vervielfältigt wurde, sind vorbei. Die Bereitstellung muss flexibler sein und es gibt bereits entsprechende Tools dafür. Für Anwendungs-Umgebungen ist Docker ein mögliches Tool. Es stellt Daten-Container über das Copy-on-Write-Dateisystem und Linux-Container bereit. Warum sollte man es also nicht auch für Datenbanken nutzen?

Achtung: Die beschriebene Vorgehensweise ist für die Produktion nicht geeignet. Es geht hier nicht um eine ausgereifte Lösung, sondern um einige Prototypen, die zeigen, was machbar ist, und praktische Erfahrungen liefern. Von daher freut sich der Autor über Feedback („*franck.pachot@dbi-services.com*").

Der Artikel beschreibt nicht, was Docker ist. Dafür gibt es unter *„www.docker.com"* eine eigene Website. Hier geht es um eine Oracle-Installation in einem Docker-Container anhand eines Beispiels. Dazu wird eine Stand-by-Datenbank erstellt, die sich laufend aus einer Quell-Datenbank aktualisiert. Darüber hinaus lassen sich neue virtuelle Datenbanken sehr einfach erstellen: Ein neuer Docker-Container stellt den Speicher für Daten bereit, die nicht aktualisiert werden.

Docker und Oracle installieren

Man kann Docker entweder unter Linux installieren oder die „boot2docker"-VirtualBox-VM verwenden. Hier werden „boot2docker" und VirtualBox auf einem Windows-Laptop installiert. Dadurch entsteht eine VirtualBox VM; über das Verzeichnis „/users" ist das Windows-Verzeichnis von Docker aus im Zugriff. Dazu werden die Oracle-Software-Installationsdateien gestartet. Hinweis: Die entsprechenden Befehle und Listings stehen in der englischsprachigen Version des Artikels unter *„www.doag.org/go/News/201503/Pachot"*.

Mit Docker beginnt man nie ganz von vorn. Man startet in einem Container, der aus dem Repository („Pull") heruntergeladen wird. Im öffentlichen Verzeichnis fehlt allerdings das Image mit dem bereits installierten Oracle. Grund dafür sind die Lizenz-Bedingungen, die eine VM mit bereits installierter Software nicht erlauben. Es gibt allerdings eine Ausnahme: Oracle XE. In der Tat enthält das Verzeichnis Docker-Images mit Oracle XE.

Hier kommt jedoch eine vollständige Oracle-Datenbank 12c mit Data Guard Stand-by zum Einsatz, die selbst zu installieren ist. Die Software, die von der Oracle-Website *(siehe „http://www.oracle.com/technetwork/database/enterprise-edition/downloads")* heruntergeladen wird, enthält die beiden Zip-Dateien „linuxamd64_12102_database_1of2.zip" und „linuxamd64_12102_database_2of2.zip". Diese werden in einem Verzeichnis abgelegt, das mit dem Docker-Container verbunden ist. Weil hier „boot2docker" zum Einsatz kommt, wird das Verzeichnis „/users" in der Boot2docker-VM verwendet. Dann startet man das „breed85/Oracle-12c"-Image mit dem Tag „12101" und verbindet es unter „/users" mit der Boot-2docker-VM. Wenn das „breed85/Oracle-12c"-Image fehlt, wird es heruntergeladen. In diesem Fall muss man dem Aufruf sowohl „unzip" als auch das „preinstall"-Package hinzufügen.

Im Yum-Cache wird etwas Platz geschaffen und das „/oracle"-Verzeichnis erstellt, in dem alles liegt, auf das Oracle zugreifen

möchte. Die Installation erfolgt im Silent-Modus, wofür eine Response-Datei zu erstellen ist. Anschließend wird der Oracle-User erzeugt. Man ignoriert eine Menge System-Voraussetzungen, weil die Konfiguration ja nicht im Support ist. Die Installation ist nach etwa fünf Minuten abgeschlossen und man kann die „root.sh" ausführen.

Da es sich um einen Test handelt, ist der Platz in der Docker-VM nicht besonders groß, sodass man die nicht mehr benötigten entpackten Dateien löscht. Das neue Image wird abgespeichert und der Container entfernt.

Die Datenbank duplizieren

Ziel ist es, die Virtualisierung einer vorhandenen Datenbank zu simulieren. Dazu wird eine vorhandene Datenbank mit RMAN dupliziert und als physische Stand-by-Datenbank eingerichtet. Die einfachste Quell-Datenbank ist mit „dbca" und allen Standard-Optionen schnell erstellt. Um die Sache einfach zu halten, ist es keine Container-Datenbank. Sie heißt „DEMO111" und ist auf „192.168.78.111" gehostet. Das „SYS"-Passwort lautet „Oracle"; „dg_brokerstart" ist auf „true" eingestellt, weil die Stand-by-Datenbank mit Data Guard erstellt werden soll. Die Docker-VM hat die IP-Adresse „192.168.59.104". Wer die Listings kopiert und einfügt, muss diese Werte ändern.

Zunächst wird eine Shell in einem Container in dem Image gestartet, das zuvor erstellt wurde. Der Host-Name lautet „DEMO111_SBY". Das Verzeichnis „/users" ist jetzt nicht mehr notwendig; ab sofort wird der Port „1521" auf „9000" umgeleitet. Damit ist der gestartete Listener auf Port „1521" im Container („listening to //DEMO111_SBY:1521 into the container") extern mit der Adresse „//192.168.59.104:9000" verbunden.

Jetzt wird die Umgebung eingerichtet. Der Name der Instanz ist „DEMO111", die erstellte Passwort-Datei identisch mit der Primär-Datenbank. Der Listener erhält einen statischen Service und wird gestartet. Die Verzeichnisse entstehen per OMF; für die Instanz ist eine „init.ora" erforderlich. Nachdem daraus „spfile" erzeugt und die Instanz gestartet wird, lässt sich die Netzwerk-Verbindung testen.

Wenn alles korrekt läuft, kann man jetzt die Quelle mit dem Ziel verbinden und eine RMAN-Duplizierung starten. Es ist zwar empfohlen, den Verbindungs-String in einer „tnsnames.ora" zu deklarieren, mit dem „ezconnect"-String geht es allerdings schneller, solange es weniger als 64 Zeichen sind.

Die Stand-by-Datenbank konfigurieren und aktualisieren

Über den Data-Guard-Broker ist die Stand-by-Datenbank schnell aufgesetzt. Es ist zu bedenken, dass es nur darum geht, eine synchronisierte Stand-by-Datenbank zu haben. Es wird nie in den Docker-Container gewechselt. Um die Konfiguration behalten zu können, werden die interaktive Shell beendet, das Image abgespeichert und anschließend der Container entfernt. Das neue Image hat jetzt zusätzlich 2,13 GB für die Datenbank.

Um die Stand-by-Datenbank laufend zu aktualisieren, wird ein Container ausgeführt, der permanent läuft. Er kann im Hintergrund sein, zum Testen befindet er sich jedoch in einem Fenster. Der Listener, die Instanz sowie Docker werden in einem anderen Fenster gestartet.

Die virtuelle Datenbank bereitstellen und öffnen

Die erstellen Images sind aufeinander aufgebaut und der „refresh"-Container läuft. Damit gibt es jetzt einen zusätzlichen virtuellen Datenbank-Server, der über 10,2 GB (System + Oracle-Software + Datendateien + archivierte Protokolle) verfügt, aber nur 1,79 GB Speicherplatz benötigt – genau die Größe des „vdb1"-Layers.

Um die neue virtuelle Datenbank zu verwenden, wird einfach das Image in einem neuen Container ausgeführt und Port „1521" an einen anderen Port umgeleitet. Die „rm"-Option entfernt den Container, wenn die Shell endet. Grund dafür ist, dass er nur für eine kurze Zeit genutzt wird.

Der Zugang zur virtuellen Datenbank erfolgt über den umgeleiteten Port „9001", der Verbindungs-String lautet demnach „//192.169.59.104:9001/DEMO111_SBY". Jetzt wird der Listener gestartet. Dabei ist zu beachten, dass die „listener.ora", die für einen anderen Host-Namen konfiguriert wurde, entfernt ist. Nach dem Start der Instanz wird die Stand-by-Datenbank als „primary" aktiviert, damit man sie mit „read/write" als sogenannten „Failover" öffnen kann.

Es kann eine gute Idee sein, den Container nach dem Start zu isolieren, um sicherzugehen, dass dieser neue „Primary" nicht versucht, mit der tatsächlichen „Primary" und der Stand-by-Datenbank zu kommunizieren. Wir könnten auch den Datenbank-Namen ändern, aber Container bieten bereits die notwendige Isolation. Letztendlich soll nur die Umleitung von Port „1521" nach Port „9001" erfolgen.

Als Ergebnis laufen jetzt zwei Container und man kann so viele virtuelle Datenbanken starten, wie man will. Jede wird als Docker-Container verwaltet. Dort laufen alle Instanzen und alles liegt in Containern isoliert auf demselben Server. Jetzt lässt sich die „vdb1" mit jedem beliebigen Client verbinden, der Zugriff auf die Docker VM über das Netz hat.

Fazit

Der Artikel zeigt die Konzepte der Daten-Virtualisierung, die man hier anhand eines Beispiels anschaulich erfahren kann. Die zugrunde liegenden Technologien (Copy-on-Write-Dateisystem, Virtualisierung, Isolierung) sind vorhanden, man kann sie mit einfachen Mitteln auf dem Laptop einsetzen. Das Beispiel ist so aufgebaut, dass man die Befehle *(siehe „www.doag.org/go/News/201503/Pachot")* direkt kopieren und einfügen kann, lediglich die IP-Adressen sind zu ersetzen.

Für die Produktion wird man natürlich robuste und unterstützte Software einsetzen; viele Hersteller bieten auch gute Lösungen für Daten-Virtualisierung, einfache Snapshots und Clones an. Diese Do-it-yourself-Übung schafft die Grundlagen für die Auswahl und die Herausforderungen im Umgang mit kommerziellen Lösungen.

Hinweis: Ins Deutsche übersetzt von global syntax Language Management Services. Die englische Fassung des Artikels kann unter *„www.doag.org/go/News/201503/Pachot"* heruntergeladen werden.

Franck Pachot
franck.pachot@dbi-services.com