TimesTen Praxiserfahrung – Do's and Dont's

Jan Ott und Roland Stirnimann
Trivadis AG
CH-Glattbrugg

Schlüsselworte:

TimesTen, In-Memory Datenbank, DataStore, Cache Groups, Performance

Einleitung

Die Oracle TimesTen In-Memory Datenbank geniesst aktuell bei weitem nicht den grossen Bekanntheitsgrad wie das Oracle RDBMS. Trotzdem konnte Trivadis im Rahmen eines Kundenprojekts umfangreiche und wertvolle Erfahrung mit TimesTen sammeln. Basierend auf den gewonnenen Erkenntnissen soll dieser Artikel die folgenden Fragen beantworten: Welchen Mehrwert bringt TimesTen für Sie? Wo macht ein Einsatz Sinn? Existieren Limitationen, über die Sie als Kunde sich im Klaren sein sollten? Nebst Antworten auf diese Fragen, gibt es zu Beginn einen Überblick zur Architektur wie auch zu den Funktionen von TimesTen.

Begriffe rund um TimesTen

TimesTen bedient sich teilweise gleicher Begriffe wie die bekannte Oracle Datenbank. Doch nicht immer bedeuten diese das Gleiche. Deshalb werden einige Begriffe erklärt und gleichzeitig mit dem Oracle RDBMS gegenübergestellt.

TimesTen	Oracle
Instance: TimesTen Main Daemon	**n/a**
DataStore: Einer oder mehrere laufen unter der gleichen Instanz. Jeder hat seinen eigenen Sub Daemon.	**Database/Instance**: Es ist kein Main Daemon vorhanden, jedoch hat jede Instanz ihre eigenen Background-Prozesse.
Checkpoint Files: „Dirty Blocks" werden aus dem Memory durch den Checkpointer Thread in diese Dateien geschrieben. Stellen ein Speicherabbild dar.	**Datafiles**: DBWR Prozess schreibt „dirty Blocks" hinein. Es befinden sich nur Teile von Daten im Memory.
Transaction Logs: Enthalten alle Transaktions-Daten seit dem letzten Checkpoint. Nach jedem Checkpoint werden sie gelöscht	**Redologs/Archivelogs**: Enthalten alle Trans-aktionsdaten, werden aber nicht automatisch gelöscht.
n/a	**Controlfiles**: Enthalten Metadaten und Struktur-Informationen zur Datenbank
sys.odbc.ini: Parameter Datei	**init.ora**: Parameter Datei
n/a	Tablespaces
TimesTen Server Process: Nimmt Anfragen vom Client entgegen.	**Listener**: Nimmt Clientanfragen entgegen.

Architektur von TimesTen

Im Zentrum der TimesTen Datenbank steht das Shared Memory Segment (blau in Abb. 1). TimesTen lädt alle Daten von der Festplatte in dieses Shared Memory. Dies geschieht einmalig beim Starten der Datenbank, so dass zu einem späteren Zeitpunkt kein zeitintensives Nachladen von der Festplatte stattfinden muss.
Weiter zeigt die Abbildung die wichtigsten Prozesse, die zu einer TimesTen Instanz gehören.

- **Main Daemon**: Pro TimesTen Instanz gibt es genau einen Main Daemon. Dieser startet auch alle anderen Prozesse bei Bedarf, welche nachfolgend noch genannt werden. Der Standard Port kann je nach Plattform variieren beziehungsweise per Konfigurationsdatei explizit definiert werden.
    - **Sub Daemon**: Pro DataStore gibt es einen Sub Daemon. Bei Bedarf können jedoch mehrere DataStores pro Instanz definiert werden, wodurch automatisch mehr als ein Sub Daemon aktiv werden. Jeder Sub Daemon startet weitere Threads, die für diverse Aufgaben wie Aging oder die Persistenz (Checkpointer Prozess) auf die Festplatte verantwortlich sind.
    - **Main Server**: Dieser Prozess nimmt die Client-Anfragen entgegen und startet je nach Konfiguration weitere ttc Server Prozesse wie nachfolgend erläutert.
        - **ttc Server**: Diese Prozesse verwalten eine oder mehrere Client Verbindungen.
    - **Cache Agent**: Der Prozess kommuniziert mit einer Oracle Datenbank, falls TimesTen entsprechend konfiguriert wurde. Er liest Daten aus Oracle in die TimesTen Datenbank bzw. in die Cache Groups. Mehr dazu später.
    - **Replication Agent**: Dieser Agent macht das Gegenteil vom Cache Agent. Er schreibt Daten aus TimesTen zurück nach Oracle.

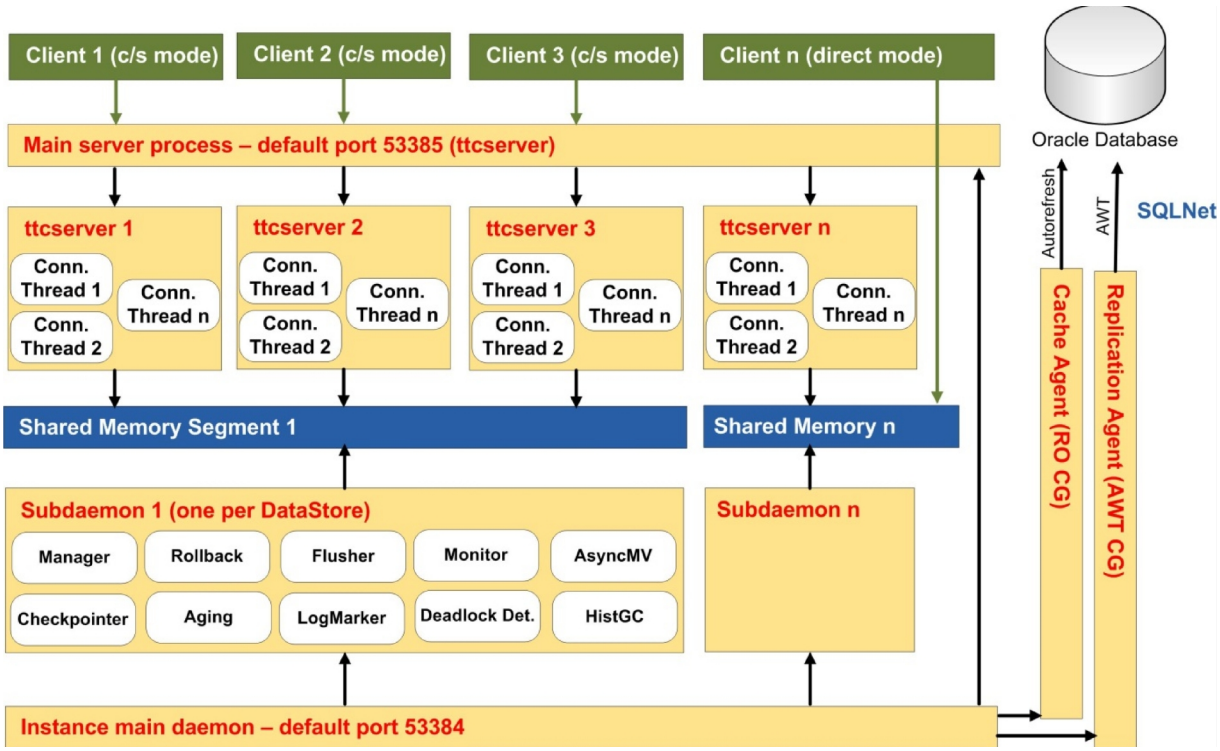

Abb. 1: TimesTen Architektur im Überblick (Quelle: Oracle)

Verbindungsarten in TimesTen

Für die Verbindung zu einem TimesTen DataStore existieren drei verschiedene Möglichkeiten.

- **Direct Driver Connection**: Applikation und DataStore müssen auf dem gleichen Host sein. Somit kann der Direct ODBC Driver den DataStore in das gleiche Memory Segment der Applikation laden. Es entsteht kein Zeitverlust durch Netzwerk- oder IPC-Kommunikation und somit ist dies die beste Variante im Bezug auf Performance.
- **Driver Manager Connection**: Unabhängiges Treiber Interface, z.B. ODBC. Die Applikation kann unabhängig von der Datenbank entwickelt werden.
- **Client/Server Connection**: DataStore und Applikation können auf unterschiedlichen Hosts installiert sein. Es werden drei weitere Typen unterschieden.
    - TCP/IP Kommunikation
    - UNIX Domain Sockets auf der gleichen Maschine
    - IPC (Inter Process Communication) auf der gleichen Maschine via eines Shared Memory Segments

IMDB Cache für Oracle Tabellen

Ein sogenannter TimesTen DataStore kann als eigenständige Datenbank betrieben werden. Interessant erscheint jedoch die Möglichkeit Tabellen aus einer Oracle Datenbank in TimesTen zu laden. Dieses Feature von TimesTen nennt sich IMDB Cache (In-Memory Database Cache). So genannte Cache Groups bilden die gewünschten Tabellen aus Oracle in TimesTen ab. Eine Cache Group ist nur ein logisches Gebilde, welches eine oder mehrere Tabellen aus einer Oracle Datenbank zusammenfasst. Für die Applikation sind die Tabellen unter dem gleichen Namen in TimesTen ansprechbar wie sich die Entwickler aus Oracle bereits gewohnt sind. Die folgende Abb. 2 zeigt die prinzipielle Funktionsweise einer Cache Group. TimesTen kann nur Tabellen aus einer Oracle Datenbank laden. Weitere Hersteller sind nicht unterstützt.

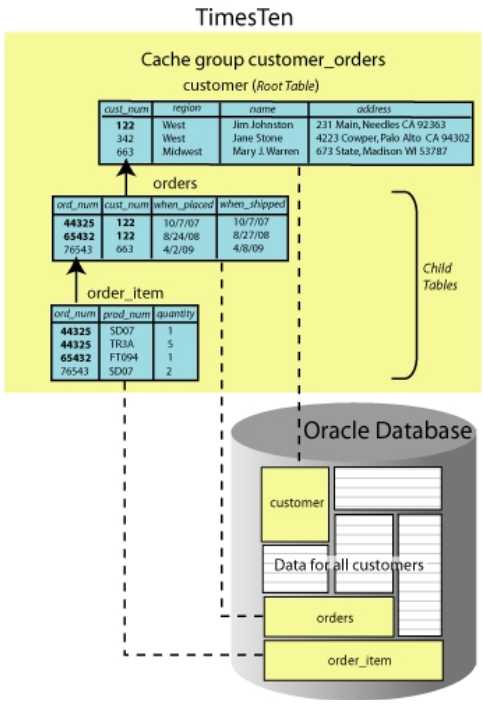

Abb. 2: Funktionsweise von Cache Groups in TimesTen
(Quelle: http://download.oracle.com/docs/cd/E13085_01/doc/timesten.1121/e13073/concepts.htm#BABFBIEC)

Tabellen innerhalb einer Cache Group müssen via Fremdschlüssel miteinander verbunden sein. Das heisst es können mehrere Tabellen zu einer Cache Group gehören, die untereinander eine Baumstruktur bilden. Es gibt immer nur eine Master-Tabelle.

TimesTen Installation

Die Installation von TimesTen verläuft ohne Probleme. Die Dokumentation enthält alle erforderlichen Schritte. Deshalb wird in diesem Artikel keine Schritt-für-Schritt Anleitung festgehalten. Lediglich ein paar Besonderheiten sind hervorgehoben.
Je nach erwarteter Datenbank Grösse sollten vor der Installation die Huge-Pages konfiguriert werden.

```
# Setzen von HugePages unter SLES 10 als root
vi /etc/sysctl.conf
vm.hugetlb_shm_group=202       # Group ID der dba Gruppe
vm.nr_hugepages=81920          # Anzahl von 2 MB Pages unter SLES10
```

Die Grösse einer Huge-Page lässt sich mit dem Kommando „*grep Huge /proc/meminfo*“ prüfen.

```
# Aktivieren der HugePages ohne Reboot
/sbin/sysctl -p
```

Falls das System nicht in der Lage ist alle Huge-Pages als zusammengehöriges Memory Segment zu allozieren, muss allenfalls trotzdem ein Neustart durchgeführt werden.
Wie von Oracle gewohnt müssen weiter diverse Kernel Parameter eingestellt werden, so zum Beispiel die maximale Grösse des Shared Memory Segments. Um die Frage nach dem Daemon Port gegenüber dem Installer zu beantworten, sollte sich der DBA im Voraus diesbezüglich ein paar Gedanken machen. In grossen und standardisierten Umgebungen verlangen die Richtlinien möglicherweise eine Trennung von Software Binaries, Datenfiles und Konfigurationsdateien. Entsprechend sind diese Speicherorte vor der Installation zu definieren.
Zwingend muss vor der Installation als Benutzer root ein File mit Namen „instance_info“ angelegt werden. Als Dateibesitzer muss anschliessend der Besitzer oder die Gruppe der TimesTen Binaries gesetzt werden unter jenem die Datenbank schlussendlich läuft.

```
touch /etc/TimesTen/instance_info
chgrp -R dba /etc/TimesTen
chmod 770 /etc/TimesTen
chmod 660 /etc/TimesTen/*
```

Nun kann das Setup-Script gestartet werden. Es führt interaktiv durch die Installation. Um das Leben in grossen Umgebungen zu erleichtern existiert auch eine „*silent*“ Variante. Dazu wird eine initiale Installation aufgezeichnet wobei der Installer ein Response File erstellt. Mit dieser Response Datei kann anschliessend eine Installation ohne Benutzerinteraktion durchgeführt werden.

```
# Aufzeichnen einer Installation
./setup.sh -install -record /opt/oracle/TimesTen_Server_112171.rsp
# Silent Installation durchführen
./setup.sh -install -batch /opt/oracle/TimesTen_Server_112171.rsp
```

Nach einer erfolgreichen Installation läuft der TimesTen Main Daemon. Dieser lässt sich auch manuell wie folgt starten und stoppen.

```
ttDaemonAdmin -stop
ttDaemonAdmin -start
```

Zu diesem Zeitpunkt existiert noch kein DataStore. Ein solcher kann in der zentralen *sys.odbc.ini* Datei definiert werden. Die Checkpoint und Log Files werden erst bei der ersten Verbindung zum DataStore auf der Festplatte angelegt.

```
vi $TT_HOME/info/sys.odbc.ini

# Inhalt
[MYDS]
Driver=/opt/oracle/TimesTen/tt11.2_b/lib/libtten.so
DataStore=/opt/oracle/dsn/myds      # Pfad mit Prefix für Checkpoint Files
LogDir=/opt/oracle/dsn              # Pfad für Log Files
PermSize=32                         # Grösse des DataStores in MB
TempSize=32                         # Grösse des Temp Bereichs in MB
PLSQL=1                             # PL/SQL Support aktivieren
DatabaseCharacterSet=AL32UTF8
OracleNetServiceName=DB02.world     # TNS Alias falls Cache Groups aktiv
```

Nun kann der Daemon neu gestartet werden und bei der ersten Verbindung wird der DataStore physikalisch angelegt.

```
ttDaemonAdmin –restart
ttIsql MYDS
```

Die zweite Zeile entspricht einer Verbindung „*as sysdba*“ unter Oracle. Nun kann der DBA Tabellen, Views, usw. anlegen oder aber auch Cache Groups. Darüber steht im nächsten Kapitel mehr.

TimesTen Cache Groups in Aktion

Cache Groups dienen dazu Tabellen aus einer Oracle Datenbank in TimesTen zu laden, sowie bei Bedarf nach Oracle zurück zu schreiben. Es werden folgende Cache Group Typen unterschieden:

- **Read Only Cache Group**: Daten werden nur nach TimesTen gelesen und können in TimesTen nicht verändert werden. Datenänderungen in Oracle können bei Bedarf automatisch in TimesTen aktualisiert werden (autorefresh Funktion).
- **Asynchronous Write Through Cache Group**: Daten können bi-direktional ausgetauscht werden, d.h. TimesTen lädt diese aus Oracle und schreibt Änderungen auch wieder nach Oracle zurück. Diese Vorgänge sind asynchron, so dass die Performance in TimesTen nicht beeinträchtigt wird.
- **Synchronous Write Through Cache Group**: Für maximale Datensicherheit in TimesTen schreibt dieser Typ alle Daten synchron nach Oracle. Somit muss TimesTen warten bis der „*commit*“ aus Oracle ankommt. Entsprechend kann die Applikation ausgebremst werden.
- **User Managed Cache Group**: Dieser Typ erlaubt ebenfalls lesen und schreiben von und nach Oracle. Jedoch können manuell auf Kommando diese Vorgänge gestartet werden (autorefresh und propagate).

Ein bestimmter DataStore kann immer nur zu einer Oracle Datenbank verbinden und folglich stammen auch die Cache Groups daraus. Dies wird in der Datei *sys.odbc.ini* definiert. Vor dem Anlegen der ersten Cache Group müssen initial einige Schritte durchgeführt werden (erstellen eines Schemas, ausführen von Scripts, usw.). Diese Aktionen sind ausführlich in der TimesTen Dokumentation beschrieben.
Folgendes Beispiel zeigt exemplarisch wie eine „Read Only" Cache Group mit zwei Tabellen erstellt wird. Dabei sind die Tabellen über den Schlüssel „*deptno*" verbunden.

```
create readonly cache group cg_deptartments
autorefresh interval 10 seconds
from
scott.dept (
 deptno number(2) not null, dname varchar2(14),
 loc varchar2(13),
 primary key (deptno)),
scott.emp (
 empno number(4) not null,
 ename varchar2(10), job varchar2(9), deptno number(2),
 primary key(empno),
 foreign key (deptno) references scott.dept(deptno));
```

PL/SQL und SQL Support

Das wichtigste Zuerst: SQL und PL/SQL in Oracle DB und TimesTen ist nicht das gleiche. Oracle arbeitet hart daran alle Befehle und Syntax nach TimesTen zu portieren, doch leider sind sie noch nicht ganz so weit.
Hier ein Auszug von Unterschieden:

- CONNECT BY wird nicht unterstützt
- WITH wird nicht unterstützt
- DISTINCT und SEQUENCES kombiniert wird in gewissen Fällen nicht unterstützt
- SELFREFERENZING FOREIGN KEYs werden nicht unterstützt
- PRIMARY KEY Syntax ist verschieden
- Und noch vieles mehr

Des Weiteren ist ein SQL in TimesTen nicht parallelisierbar. Da hat Oracle bei komplexen, aufwendigen SQL einen Vorteil und kann, durch die Möglichkeit die Arbeit auf mehrere Prozessoren zu verteilen, schneller sein.
Ein Beispiel wie das aussehen könnte:

```
ALTER TABLE emp
  ADD CONSTRAINT EMP_EMP_FK
    FOREIGN KEY (mgr) REFERENCES emp (empno);
 3000: self-referencing foreign keys are not allowed
```

Dies herauszufinden ist sehr zeitaufwändig und wird noch zusätzlich erschwert dadurch, dass der Compiler gewisse Syntax schon kennt und somit ohne Fehler kompiliert. Doch dann zur Laufzeit einen Fehler aufwirft. Dieser ist meist nicht selbsterklärend.
Die Aufwandschätzung in ein paar Stunden den Code von Oracle auf TimesTen zu portieren, wurde in der Realität zu einem Projekt von mehreren Tagen.

Performance und Analyse

Zu aller erst TimesTen ist schneller. Wenn Oracle im Millisekunden-Bereich liegt, dann ist TimesTen im Mikrosekunden-Bereich. Nun kommt jedoch ein „Aber".
Parallel Query funktioniert in TimesTen nicht. Ein SQL muss immer von einem und nur einem Prozess abgearbeitet werden. Damit bekommt natürlich Oracle einen Vorteil, wenn die Queries grösser und komplexer sind.
TimesTen ist schneller, doch wenn das Netzwerk die langsame Stelle ist, ist der Vorteil von TimesTen nicht mehr markant. Ein kleines Rechnungsbeispiel: Oracle benötigt 0.01 sec. TimesTen benötigt 0.001 sec. Das Netzwerk benötigt 0.2 sec. Daraus ist ersichtlich, dass die TimesTen Variante mit 0.201 zu 0.21 sec nur unwesentlich schneller ist.
Dies sieht anders aus, wenn der Direct Driver von TimesTen verwendet wird. Alles läuft auf einem Server. Das Netzwerk ist somit nicht involviert. Da ist eine Oracle DB Lösung massiv weniger performat.
Noch ein paar Worte zum Analysieren der Performance und finden von Ressourcenfressern. Dies ist in TimesTen bis zum heutigen Tag rudimentär implementiert und benötig viel Zeit. Oracle hat ein über Jahre gereiftes System von Reports, System Views und den Enterprise Manager um Probleme zu analysieren. Oracle verspricht das in TimesTen auszubauen.

Fazit

TimesTen ist schnell, sehr schnell vor allem kombiniert mit Direct Driver.
Die Frage ist nun will man in ein neues Produkt investieren? Neues Wissen aufbauen? Neue Backup/Recovery Szenarien aufbauen, da nicht die gleichen Mechanismen wie bei Oracle verwendet werden können?
All das kostet, doch vielleicht löst schon schnellere Hardware das Problem. Wenn das jedoch nicht der Fall ist, ist TimesTen eine gute Lösung.. Die Integration mit Oracle ist sehr gut gelöst und Oracle arbeitet laufend daran die Lücken, die noch bestehen, zu schliessen. Das sieht man schon heute darin, dass die ersten Oracle Dictionary Views in TimesTen vorhanden sind sowie SQL und PL/SQL zu 90% kompatibel ist.
Die Autoren würden auf jeden Fall das Wagnis wieder eingehen.

Kontaktadresse:

Jan Ott / Roland Stirnimann
Trivadis AG
Europa-Strasse 5
CH-8152 Glattbrugg

Telefon: +41 (0) 44-808 7020
Fax: +41 (0) 44-808 7021
E-Mail jan.ott@trivadis.com / roland.stirnimann@trivadis.com
Internet: www.trivadis.com