The Next Generation of Extreme OLTP Processing with Oracle TimesTen

Tirthankar Lahiri
Oracle
Redwood Shores, California, USA

Keywords: TimesTen, velocity scaleout elastic inmemory relational database cloud

Introduction

Traditional database dogma would have it that in order to achieve high levels of horizontal scalability one needs to give up many of the desirable characteristics of relational database systems such as:

- SQL and Joins
- ACID Transactions
- Full Consistency
- Normalised Data Models

And further that you will need to:

- Manually shard your database
- Re-write your applications to fit this brave new world.

With the Oracle TimesTen Velocity Scale In-Memory Database, we refute these suggestions and show that you can retain all of these desirable capabilities while still achieving extreme performance at scale either on-premises or in the cloud.

Fully distributed scale-out in-memory database

TimesTen Velocity Scale builds on the strong foundations of 'classic' TimesTen to deliver a fully distributed in-memory database with a shared nothing architecture. A Velocity Scale grid consists of multiple database instances each hosted in a separate machine, VM or container. The grid can host one or more databases; each database consists of multiple database elements each one hosted in one of the grid instances. The data stored the database tables is automatically distributed across all the elements that make up the database.

Single database image with data location transparency

Although a Velocity Scale database is distributed across the elements that make up the database it presents a single database image to applications. Applications may connect to any of the elements that make up the database and they will be able to access and transact against all the data within the database using standard SQL. Applications do not need to know or care about the exact storage location of any specific item of data within the database.

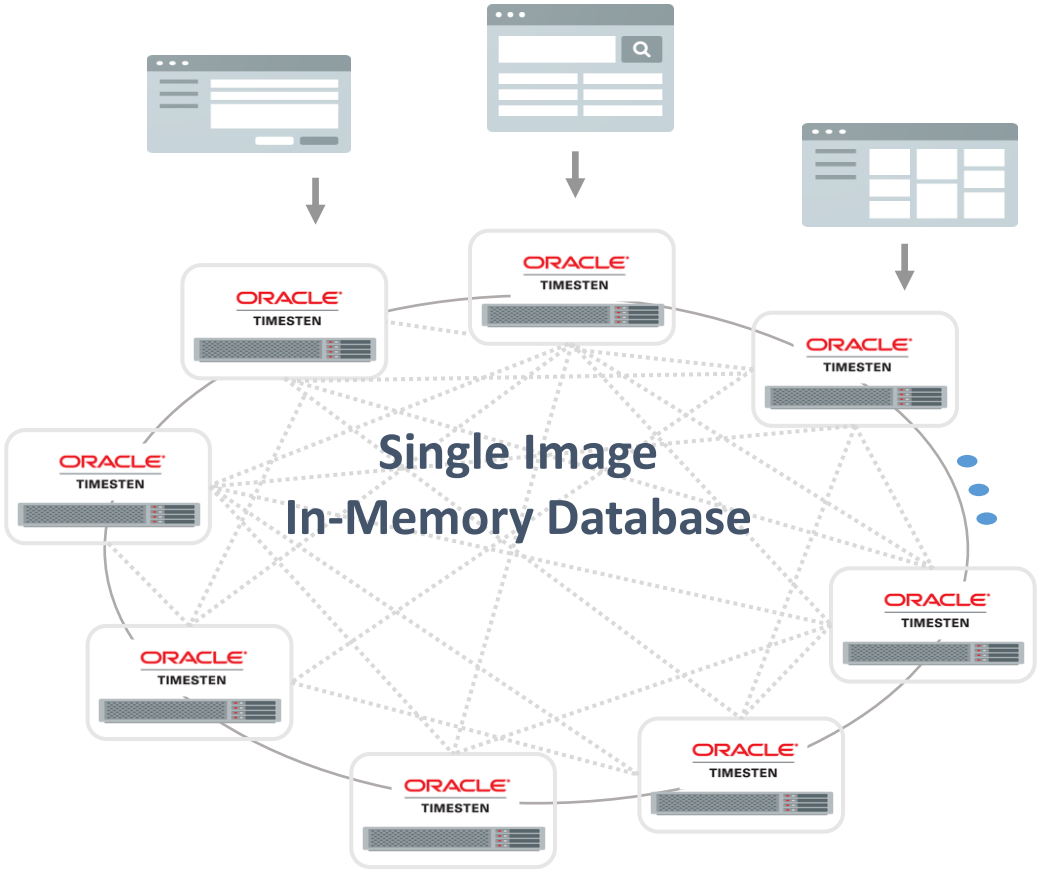

Illustration. 1: A TimesTen Velocity Scale Database Grid

Data distribution

Table data is automatically distributed across all the elements that make up the database. The exact method of distribution is chosen by the user on a table by table basis at table creation time. Three distribution methods are currently provided:

Hash	Rows are placed in elements based on the result of applying a consistent hash algorithm to one or more column values.
Reference	In a parent-child hierarchy defined by an explicit primary key – foreign key relationship the child rows are placed in the same element as their parent row. This is a useful performance optimisation for cases when the parent and child rows are often joined or when the parent and child rows are manipulated within the same transaction. By co-locating the parent and child rows inter-element network traffic can be reduced or eliminated.
Duplicate	A full copy of the table is stored in every element. This is another performance optimisation; all reads against the table are local so queries and joins against this table will not incur any network overhead.

Automatic High-Availability and Fault Tolerance

TimesTen Velocity Scale Database provides automatic high availability by way of K-safety. This involves maintaining multiple copies (the 'k' factor) of the data distributed across database elements hosted on different machines within the grid. In the event of a failure making one machine/database element unavailable the application workload will automatically continue using one of the surviving copies of that element. Restart and recovery of the failed element is automatic and once it has recovered and resynchronized it will re-join the grid.

Automatic failover for client connections is also provided. If an element fails, any client connections to that element are automatically moved to a surviving element.

Elastic Scalability

Velocity Scale allows you to grow or shrink the database grid by adding or removing new machines, instances and database elements while the grid and its databases remain operational. This elastic scalability allows you to quickly adapt data storage and processing capacity to keep pace with business demands.

Easy to Deploy and Manage

By providing a single point for installation, administration and monitoring Velocity Scale makes it simple to deploy and manage even very large database grids. Flexibility is further enhanced by the ability to deploy and manage the database using either powerful command line tools or a feature rich GUI (SQL Developer 4.2). The power and simplicity of Velocity Scale means that it is equally at home when deployed on-premises or in the cloud.

Easy Application Development

By maintaining a high degree of compatibility with both Classic TimesTen and Oracle Database, Velocity Scale facilitates the migration of existing applications and the development of new ones.

High Performance

Velocity Scale combines the key features of traditional relational databases with the high performance of in-memory technology and the scalability of a distributed database. In like for like testing against Cassandra using the YCSB benchmark Velocity scale greatly outperforms Cassandra for both throughput and scalability while also delivering order of magnitude better response times.

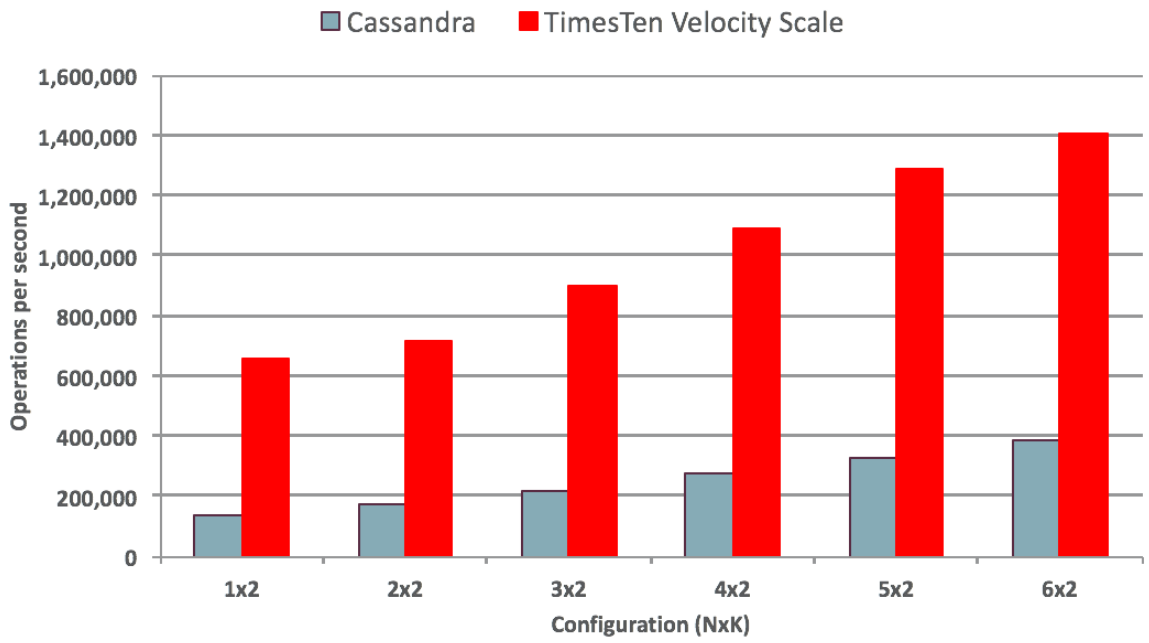

Illustration.2:YCSB Workload B; Cassandra versus TimesTen Velocity Scale; Throughput and Scalability

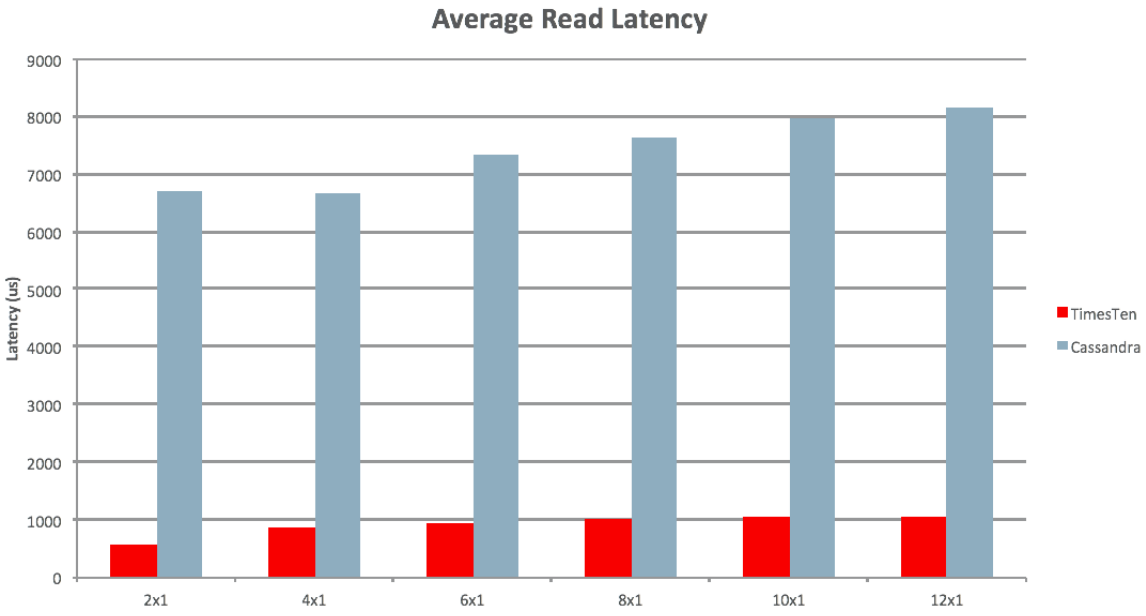

Illustration.2:YCSB Workload B; Cassandra versus TimesTen Velocity Scale; Average Read Latency

Contact address:

Chris Jenkins
Oracle
Oracle Parkway Thames Valley Park 520
Reading, RG6 1RA
Phone: +44 118 924 5243
Email chris.jenkins@oracle.com

Tirthankar Lahiri
Oracle
400 Oracle Parkway, Redwood Shores,
Redwood City, CA 94065, USA

Phone: +1 (650) 506-6279
Email tirthankar.lahiri@oracle.com
Internet: http://www.oracle.com/technetwork/database/database-technologies/timesten/overview/index.html