Frequently Asked Questions
Oracle Database 12*c* Release 2 – Oracle Sharding

Introduction

Oracle Sharding is a scalability and availability feature for custom-designed OLTP applications that enables distribution and replication of data across a pool of Oracle databases that share no hardware or software. The pool of databases is presented to the application as a single logical database. Applications elastically scale (data, transactions and users) to any level, on any platform, simply by adding additional databases (shards) to the pool. Scaling up to 1000 shards is supported in the first release with Oracle Database 12.2.0.1.

Oracle Sharding provides superior run-time performance and simpler life-cycle management compared to homegrown deployments that use a similar approach to scalability. It also provides the advantages of an enterprise DBMS, including: relational schema, SQL, and other programmatic interfaces, support for complex data types, online schema changes, multi-core scalability, advanced security, compression, high-availability, ACID properties, consistent reads, developer agility with JSON, and much more.

Product Overview

Q: Why would a customer be attracted to Oracle Sharding?

A: Oracle Sharding is a response to customer demand for a database architecture that can deliver the combination of linear scalability and complete fault isolation without requiring shared storage or compromising on the enterprise qualities of the Oracle Database: strict consistency, the full power of SQL, developer agility with JSON, security, high availability, backup and recovery, life-cycle management, etc.

OLTP applications designed for Oracle sharding can elastically scale (data, transactions and users) to any level, on any platform, simply by deploying new shards on additional stand-alone servers. The unavailability or slowdown of a shard due to either an unplanned outage or planned maintenance affects only the users of that shard, it does not affect the availability or performance of the application for users of other shards. Each shard may run a different release of the Oracle Database as long as the application is backward compatible with the oldest running version – making it simple to maintain availability of an application while performing database maintenance.

Oracle Sharding also does more than just extend the enterprise qualities of Oracle to a sharded database architecture. Oracle Sharding uses automation to simplify life-cycle management, advanced partitioning methods to address a wide array of use-cases, and data-dependent routing for superior runtime performance. Collectively these capabilities provide customers substantial advantages compared to competitive sharding solutions or custom deployments.

Q: What new development is Oracle delivering with Oracle Sharding?

A: Oracle Sharding replaces homegrown approaches for sharding with a comprehensive solution for deploying and managing a sharded database architecture. Oracle sharding provides new automation for distributing table partitions across shards, automated deployment of a sharded database including replication for HA and data protection, high performance routing, load balancing, and complete life-cycle management.

Oracle Sharding also optimizes time-proven enterprise capabilities of the Oracle Database explicitly for a sharding use-case. These include: Oracle Partitioning, Data Guard, Active Data Guard, GoldenGate, JDBC/OCI/ODP.NET connection pools, DBCA, OEM, Oracle RAC, and more.

Q: Can you summarize the benefits of Oracle Sharding?

A: Sharding with Oracle Database 12c Release 2 provides a number of benefits:

- Linear scalability with complete fault isolation. OLTP applications designed for Oracle sharding can

elastically scale (data, transactions and users) to any level, on any platform, simply by deploying new shards on additional stand-alone servers. The unavailability or slowdown of a shard due to either an unplanned outage or planned maintenance affects only the users of that shard, it does not affect the availability or performance of the application for users of other shards. Each shard may run a different release of the Oracle Database as long as the application is backward compatible with the oldest running version – making it simple to maintain availability of an application while performing database maintenance.

- Global data distribution for data proximity to bring data to the consumers and data sovereignty to meet data privacy regulations.
- Simplicity via automation of many life-cycle management tasks including: automatic creation of shards and replication, system managed partitioning, single command deployment, and fine-grained rebalancing.
- Superior run-time performance using intelligent, data-dependent routing.

All of the advantages of sharding without sacrificing the capabilities of an enterprise RDBMS, including: relational schema, SQL, and other programmatic interfaces, complex data types, online schema changes, multi-core scalability, advanced security, compression, high-availability, ACID properties, consistent reads, developer agility with JSON, and much more.

Q: What is Oracle Sharding? - A more complete description

A: Oracle Sharding is a scalability and availability feature for custom-designed OLTP applications that enables distribution and replication of data across a pool of discrete Oracle databases that share no hardware or software. Each database in the pool is referred to as a shard. The pool of shards is presented to an application as a single logical Oracle database (a sharded database or SDB).

Oracle sharding distributes data across shards using horizontal partitioning. Horizontal partitioning splits a database table across shards so that each shard contains the table with the same columns but a different subset of rows.

From the perspective of a database administrator, an SDB consists of multiple databases that can be managed either collectively or individually. However, from the perspective of an application developer, an SDB looks like a single database: the number of shards and the distribution of data across them are completely transparent to database applications. SQL statements issued by an application do not refer to shards nor are they dependent on the number of shards and their configuration.

OLTP applications must be explicitly designed for a sharded database architecture in order to realize the benefits of scalability and availability. This is different from an HA architecture based upon Oracle Real Application Clusters (Oracle RAC) where scalability and availability are achieved transparent to an application. Applications that use a sharded database must have a well-defined data model and data distribution strategy (consistent hash or composite) that primarily accesses data via a sharding key. Examples of a shard key include customer_id, account_no, country_id, etc. Oracle Sharding also supports data placement policies (rack and geo awareness) and all deployment models: on-premises and public or hybrid clouds.

Transactions that require high performance must be single-shard transactions that typically access 10s or 100s of rows. For example, lookup and update of a customer's billing record, lookup and update of a subscriber's documents etc. There is no communication or coordination between shards for high performance transactions. Multi-shard operations and non-shard key access are also supported but with a reduced expectation for performance. Such transactions include simple aggregations, reporting, etc. - typically less than 10% of total workload.

In return for these design considerations, applications that run on a sharded database architecture can achieve even higher levels of scalability and availability. Performance scales linearly as shards are added to the pool because each shard is completely independent from other shards. Each shard typically uses local storage, flash, and memory offering customers a further opportunity to optimize performance at relatively low cost. The first release of Oracle Sharding is designed to scale up to 1,000 shards. Isolation between shards also means that outages or poor performance of one shard does not impact the availability or performance of transactions executing at other shards.

High Availability (HA) for individual shards is provided by automatic deployment of database replication. Simple, one-way Data Guard physical replication with automatic database failover is the default configuration. Active Data Guard (copies open read-only) or Oracle GoldenGate (bi-directional replication with all copies open read-write) may also be automatically deployed. Shards may be replicated within and across data centers. Replication is data-center and rack aware using data placement policies supported by Oracle Sharding. Optionally, Oracle RAC may be manually configured to provide Shard HA.

Shards are front-ended by a set of replicated listeners called Shard Directors that act as routers. Oracle clients (JDBC, OCI, and ODP.net) and the Oracle Universal Connection Pool (UCP) have been enhanced to recognize shard keys specified in a connection string and to insure availability by controlling the maximum number of connections allowed per shard. A shard routing cache in the connection layer (populated by the initial request to a shard) is used to route requests directly to the shard where the data resides for optimal runtime performance. The shard routing cache is automatically refreshed if there is any change made to the sharded database (e.g. automatic rebalancing or add/delete of shards).

Application Requirements & Suitability

Q: What is required for an application to utilize Oracle Sharding?

A: Not all applications can take advantage of a sharded database. Oracle Sharding is intended for OLTP applications that are explicitly designed for a sharded database architecture.

- Applications must have a well-defined data model and data distribution strategy (consistent hash or composite) that primarily accesses data via some key. Examples of keys include customer ID, account number, country_id, etc.
- All transactions that require high performance must be single-shard transactions. These transactions provide a shard key for high performance routing and typically access 10s or 100s of rows. For example, lookup and update of a customer's billing record, lookup and update of a subscriber's documents etc.
- Multi-shard operations or non-shard key access are supported but with a reduced level of performance. Such transactions include simple aggregations, reporting, etc. - typically less than 10% of total workload.

Q: Can COTS applications such as SAP or Oracle E-Business Suite use sharding?

A: No. Sharding is explicitly intended for applications that are purpose-built to take advantage of a sharded database architecture. There are no application providers that have certified their applications for Oracle Sharding.

Q: Are there examples of target customers and applications for sharding?

A: The target customer for Oracle sharding can come from any industry vertical. Examples include:

- Mass Media and Financial Information Services providers who need massive scalability with high availability for online storage and retrieval of information.
- Airline ticketing systems whose main driver for sharding is fault isolation. They want to shard across tens of independent databases. Failure of a database only makes 1/N of the data momentarily unavailable.
- Social Media companies who may wish to allocate different shards for different classes of users/customer profiles, at different price levels.
- Online Payment Systems that shard for linear scalability and fault isolation, and who may need to satisfy regulatory requirements for storing user data in the country of citizenship.
- Financial and Tax preparation companies who shard by customer id to scale users, workload and transactions. Sharding provides these companies with elasticity required when demand for service peaks during tax filing season.
- Large billing systems where each customer can be identified by a customer ID, phone number, or user ID.

Q: Is sharding a new concept?

A: Sharding is a well established, but special purpose practice used for one or more of the following reasons:

- To massively scale both read-write and read-only workloads.
- For high availability through complete fault isolation. Since shards can be configured so that they share no hardware or software, an outage or slowdown of an individual shard affects only the users of that shard, it does not affect the users of other shards or the availability of the overall application.
- For scalability and availability when using servers with locally attached non-shared storage, (like PCI-based flash) or when a shard can fit completely in-memory. A sharded database can be deployed across a pool of such servers with no clusterware or shared storage.

Companies have not waited for Oracle Sharding to deploy sharded database architectures. Major Internet companies shard. There are also high profile Oracle customers who have developed their own approaches to partitioning their Oracle data and workloads across multiple independent Oracle databases for scalability and availability. Sharding is also used by NoSQL databases as a means to achieve scalability and HA.

Technical Aspects

Q: How is data distributed across shards?

A: Oracle Sharding uses horizontal partitioning to split a database table across shards (discrete physical databases) so that each shard contains the table with the same columns but a different subset of rows.

Distribution of partitions across shards is done at the tablespace level. Each partition of a sharded table resides in a separate tablespace and each tablespace is associated with a specific shard. The table partitions on each shard are no different from regular partitions used in non-sharded Oracle database.

Even though the partitions of a table reside in multiple databases, to the application developer the table looks and behaves exactly the same as a regular partitioned table in a single database. SQL statements issued by an application never refer to a shard, nor depend on the number of shards and their configuration.

Oracle Sharding uses the familiar SQL syntax for table partitioning to specify how table rows are partitioned across shards. A partitioning key for a sharded table is also the sharding key. For example, the CREATE SHARDED TABLE statement is used to create a sharded table.

Oracle Sharding provides a choice of partitioning methods where the distribution of data can be determined automatically or by the user, or a combination of both.

Q: What types of sharding methods are supported?

A: Oracle Sharding supports two methods of sharding in 12.2.0.1: system-managed and composite.

- **System-managed sharding** does not require the user to specify mapping of data to shards. Data is automatically distributed across shards using partitioning by consistent hash. The partitioning algorithm evenly and randomly distributes data across shards. Such distribution is intended to eliminate hot spots and provide uniform performance across shards. Oracle Sharding automatically maintains balanced distribution of data when shards are added to or removed from an SDB. System-managed sharding uses a consistent-hash partitioning strategy that is optimized for Oracle Sharding. System-managed sharding is the most used form of sharding.
- **Composite sharding** - With this sharding method, data is first partitioned by list or range and then further partitioned by consistent hash. The two levels of sharding make it possible to map data to a set of shards, and then automatically maintain balanced distribution of data across that set of shards.

Q: How are transactions contained within a single-shard?

A: For many applications, a high percentage of single shard operations can be achieved by combining horizontal partitioning with replication of a small number of read-only or read-mostly tables across all shards. Replication of complete tables is a good choice for relatively small tables that are often accessed together with sharded tables. A table with the same contents in each shard is called a duplicated table. For example a Customers–Orders–Line Items schema may also include a Products table. This table contains data shared by all customers and cannot be sharded by the customer number. Instead, the entire table is duplicated on all databases to prevent multi-shard queries during order processing. In the

example of the Products table, it is created using the CREATE DUPLICATED TABLE statement.

Oracle Sharding uses Materialized View Replication to synchronize contents of duplicated tables. A duplicated table on each shard is represented by a read-only materialized view. The master table for the materialized views is located in a special database called the Shard Catalog. The materialized views on all shards are automatically refreshed with configurable frequency. CREATE DUPLICATED TABLE automatically creates the master table, materialized views and other objects required for materialized view replication.

Q: How does an application know which shard it must connect to at runtime?

A: Applications must specify a sharding key to achieve high performance using a sharded database architecture. An online banking application, for example, can be designed to use the customer_id generated when users log-in as the sharding key. The application passes the sharding key to the connection layer when a database transaction is processed:

- Oracle JDBC, OCI, and ODP.net clients are able to recognize sharding keys specified in the connection string for high performance data dependent routing. A shard routing cache in the connection layer is used to route the request directly to the shard where the data resides.
- The Oracle Universal Connection Pool (UCP) for JDBC clients is also able to recognize sharding keys specified in the connection URL. A shard routing cache is used to route the connection directly to the shard where the data resides. Oracle UCP also enables non-Oracle application clients such as Apache Tomcat, IBM WebSphere, etc. to work with Oracle Sharding.

The UCP shard routing cache, for example, contains a mapping of the sharding keys ranges to the shards. When the application passes the sharding key to check out a connection, UCP looks up the corresponding shards on which this key exists from its routing cache. If a matching connection is available in the pool then UCP returns a connection to one of these shards by applying its internal connection selection algorithm. If there is no matching connection, a new connection is created by forwarding the request with the sharding key to the Shard Director. Upon first connection to a shard, the connection pool retrieves all key ranges in the shard and adds them to its routing cache so that subsequent connections go directly to the shard (i.e., bypasses Shard Director).

If a shard becomes unavailable, client connections are automatically re-directed to a copy of the shard for HA.

Q: How is the routing cache updated if data is rebalanced or shards added/deleted?

A: All rebalance and add/delete shard operations are executed by the SDB management tier (Shard Catalog & Shard Director). The SDB remains available and online during all such operations. Sharding routing caches are invalidated once the rebalance is complete and automatically refreshed the next time a connection is routed to a shard.

The Oracle Universal Connection Pool (UCP), for example, keeps its routing cache in-sync with changes to a shard during resharding events such as chunk move or split. This is achieved by subscribing to new ONS (Oracle Notification Service) events that are sent by the shard during resharding. Upon the first connection to the shard, UCP subscribes to receive all chunk events for all services "eventType=database/event/<chunk>" (a 'chunk' is the unit of data movement in an SDB – for more info see the FAQ, How do I rebalance workload across shards?). For any case where there are Chunk Down or Chunk Split events, UCP updates the Chunk placement table and the shard routing cache.

Q: Can I run reports that must summarize data stored in multiple shards?

A: Oracle Sharding supports routing and processing of queries and transactions that access data stored on multiple shards to enable simple aggregation of data and reporting across shards. Oracle Sharding uses a coordinator node in such cases, similar to other sharding solutions. The user must accept a reduced level of performance compared to single-shard OLTP transactions (the primary use-case for Oracle Sharding).

Q: Can an application access data without specifying a shard key?

A: Oracle Sharding supports routing for queries that do not specify a shard key. This allows the flexibility for any database application to execute SQL statements (including SELECT and DML) in a system where tables are sharded or replicated without the need to specify the shards where the query should execute. Oracle Sharding uses a coordinator node in such

cases, similar to other sharding solutions. The user must accept a reduced level of performance compared to OLTP transactions that specify the sharding key (the primary use-case for Oracle Sharding).

Q: How do I elastically scale a sharded database?

A: Oracle Sharding supports the online addition of shards for additional scalability.

Q: How do I rebalance workload across shards?

A: Data migration across shards is required in the following cases:

- When one or multiple shards are added to or removed from an SDB
- When there is skew in the data or workload distribution across shards

The process of redistributing data between shards triggered by a change in the number of shards is called resharding. Automatic resharding provides uniform data distribution across an SDB. To understand how this is done it is necessary to understand how data is physically partitioned across shards.

Distribution of partitions across shards is achieved by creating partitions in tablespaces that reside on different shards. To minimize the number of multi-shard joins, corresponding partitions of all tables in a table family are always stored in the same shard. Each partition of a sharded table is stored in a separate tablespace. Thus a tablespace is a physical unit of data distribution in an SDB.

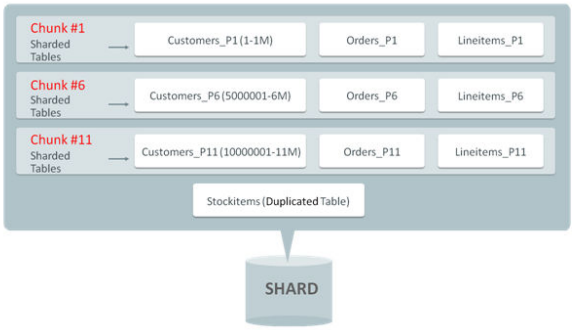

Figure 1: A shard having 3 chunks of horizontally partitioned tables and a duplicated table

The unit of data migration between shards is chunk. A chunk is a set of tablespaces that store corresponding partitions of all tables in a table family. A chunk contains a single partition from each table of a table family. This guarantees that related data from different sharded tables are moved together. The number of chunks within each shard is specified when the SDB is created. Figure 1 shows a shard with 3 chunks. Each chunk contains a set of partitions that contain related data from multiple sharded tables. Also shown is a duplicated table that resides in a non-sharded tablespace (Stockitems) that is present on all shards.

When a shard is added to or removed from an SDB, multiple chunks are migrated to maintain a balanced distribution of data and workload across shards. Depending on the sharding method, resharding happens automatically or is directed by the user.

A particular chunk can also be moved from one shard to another when data or workload skew occurs without any change in the number of shards. In this case, chunk migration is initiated by the DBA to eliminate the hot spot. Alternatively, Oracle Sharding also supports online split of a chunk. A split is required when chunks become too big, or only part of a chunk has to be migrated to another shard.

RMAN Incremental Backup, Transportable Tablespace and Oracle Notification Service technologies are used to minimize impact of chunk migration on application availability. A chunk is kept online during chunk migration. There is a short period of time (a few seconds) when data stored in the chunk is available for read-only access only. The process of migrating chunks is automated once initiated by the administrator.

FAN-enabled clients receive notifications when a chunk is about to become read-only in the source shard and when it's fully available in the destination shard on completion of chunk migration. When clients receive the 'chunk read-only' event, they can either repeat connection attempts until the chunk move is completed, or access the read-only chunk in the source chunk. In the latter case, an attempt to write to the chunk will result in a run-time error.

Q: How does sharding deliver linear scalability?

A: Linear scalability is achieved by eliminating any dependency between shards. Each shard is an independent Oracle Database that shares no hardware or software. Transactions that require high performance and scalability only access data contained in a single shard. OLTP applications designed for Oracle sharding can elastically scale (data, transactions and users) to any level, on any platform, simply by deploying new shards on additional stand-alone servers. Each shard typically uses local storage, flash, and memory offering

opportunities to further optimize performance at relatively low cost.

Single shard transactions use a combination of tables that are partitioned according to the shard key (all data related to a given key is located in the same shard), and duplicated tables (read-only or read-mostly tables containing reference data that are replicated across all shards, e.g. a products table).

Workload is directed to the appropriate shard using the combination of the shard key provided by the application and high performance data-dependent routing supported by Oracle clients (JDBC, OCI, and ODP.net) and the Oracle Universal Connection Pool (UCP). Examples of shard key include: customer_id, account_no, country_id, etc.

Q: How does sharding achieve fault isolation?

A: By definition, each shard is an independent Oracle database that shares no hardware or software with other shards. An outage or slowdown of one or more shards does not impact the availability of the application on other shards. Replication used for HA at the shard level quickly restores availability should a shard experience an outage. Likewise, planned maintenance – hardware, O.S., database maintenance and upgrades, to one shard does not affect the availability of other shards.

The additional components of a sharded deployment – shard directors and the shard catalog database are also designed to eliminate single point of failure. A shard director is a very lightweight process that does not include an Oracle database instance. Redundant shard directors are deployed within each region where shards are deployed to insure continuous application access to the SDB. The shard catalog database uses Oracle Data Guard replication with automatic failover to provide high availability. The catalog database has no effect on the routing of connections at runtime – client connections use a shard routing cache for high performance data-dependent routing. The momentary unavailability of the catalog database during a Data Guard automatic failover causes only a short brownout for shard maintenance operations or multi shard queries.

Q: Can I perform a database upgrade on a shard and not affect other shards?

A: Each shard may run a different release of the Oracle Database as long as the application is backward compatible with the oldest running version – making it simple to maintain availability of an application while performing database maintenance.

Individual shards may be upgraded in rolling fashion using Data Guard with minimal downtime, or online using Oracle GoldenGate bi-directional replication. In addition, if Oracle RAC is used for shard HA then all of the customary RAC-Rolling maintenance may be performed with zero downtime for the shard.

Q: How do I implement HA and disaster recovery for a sharded database?

A: All of the customary Oracle high availability solutions used for any Oracle Database are also used to provide HA, backup and recovery, and disaster recovery for a sharded database.

- Data Guard with automatic database failover is the default HA configuration for unplanned outages and planned maintenance and is automatically deployed for each shard
- Optionally the administrator may automatically deploy Active Data Guard (all shard replicas open read-only) or Oracle GoldenGate bi-directional replication for shard HA.
- Alternatively, an administrator can manually configure Oracle RAC for shard HA
- Oracle Recovery Manager (RMAN) and Flashback provide backup and recovery and fast point-in-time recovery at a shard level.
- The Zero Data Loss Recovery Appliance (Recovery Appliance) offers efficient enterprise backup and recovery. The recovery appliance can perform real-time backups thus protecting every transaction in an SDB.
- Additional remote copies of each shard are maintained by Data Guard, Active Data Guard or Oracle GoldenGate and provide real-time disaster recovery and data protection against site outages.
- Edition-Based Redefinition provides for online patching of shards when deploying new versions of an application that modify back-end database objects.

Q: What are the compatibility requirements for each shard in an SDB?

A: Each shard is an independent Oracle Database. Oracle Sharding requires a minimum release of Oracle Database 12.2.0.1 and Oracle Client 12.2.0.1. For the first release of Oracle Sharding, Oracle recommends using the same OS for all the shards. This recommendation will be lifted in future releases. Beyond this, there are no additional compatibility requirements across shards that comprise an SDB. There are, however, compatibility requirements between individual shards and their replicas that are determined by the approach selected for shard-level HA:

- Shards that use Data Guard or Active Data Guard must have corresponding Data Guard standbys that are at the same database version and patch set level (except when performing a database upgrade). A Shard and its standbys have the same compatibility requirements as apply to any Data Guard configuration as described in My Oracle Support Note 413484.1.
- Shards that use Oracle GoldenGate have the same requirements for compatibility between replicas as for non-sharded deployments.
- Shards that use Oracle RAC have the same requirements for compatibility across RAC nodes as for non-sharded deployments.

Cloud

Q: Is Oracle Sharding supported on Oracle Cloud?

A: Oracle Database Cloud Services (PaaS) do not currently provide turnkey automation for deploying a sharded database. Automated deployment of Oracle Sharding using Oracle Database Cloud Services is on the development roadmap for end of calendar 2018. Hybrid Cloud and all-in Cloud deployment models are planned to be supported as well as both single data center and multi data center deployments.

Till then, Oracle Sharded databases can be deployed on Oracle Cloud (DBCS instances) using the cookbook.

Licensing

Q: How is Oracle Sharding Licensed for Oracle Cloud (PaaS)?

A: With DBCS EE and DBCS EE-High Performance (HP), use is limited to three primary shards. (No limit on number of standby shards)

With DBCS EE-Extreme Performance (EP) and Exadata Cloud Service (ECS), no limit on number of primary shards or standby shards. Ref: Licensing Information Manual

Q: How is Oracle Sharding Licensed for On-premises?

A: For a sharded database (SDB) with 3 or fewer primary shards, Oracle Sharding is included with EE (includes Data Guard). No limit on number of standby shards.

For an SDB with more than 3 primary shards, in addition to EE, all shards must be licensed either for Active Data Guard, Oracle GoldenGate or RAC.

So, If your ULA covers EE and one of the HA options – Active Data Guard or Oracle GoldenGate or RAC, you can use Oracle Sharding at no additional cost.

Q: What is the BYOL model for Oracle Cloud (IaaS)?

A: For a sharded database (SDB) with 3 or fewer primary shards, Oracle Sharding is included with EE (includes Data Guard). No limit on number of standby shards. For the 3 primary shards, if Active Data Guard, Oracle GoldenGate or RAC are required for these shards, they ought to be licensed accordingly.

For an SDB with more than 3 primary shards, in addition to EE, all shards must be licensed either for Active Data Guard, Oracle GoldenGate or RAC.

Positioning

Q: How does Oracle Sharding compare to Oracle RAC?

A: The combination of Oracle RAC and Active Data Guard provide transparent scale-out and availability for OLTP and analytic applications and can fulfill the requirements of 99+% of use cases. With Oracle RAC, all transactions can act on any data within the database, there is no need to partition data or concern for the performance of multi shard operations, all RAC instances share direct access to the same physical database.

Oracle Sharding compromises on transparency in return for providing massive scalability and high availability for custom-designed OLTP applications. With Oracle Sharding, applications are designed so that update transactions act on data within a single shard. Transactions that span multiple shards do not benefit from the same level of performance and scalability as single-shard transactions.

Q: When would I use Oracle Sharding instead of Oracle RAC?

A: If it is an existing application and the customer is meeting service levels for scalability and availability with Oracle RAC, then there is no need for Oracle Sharding. For new applications your default assumption unless proven otherwise should be that Oracle MAA using the combination of Oracle RAC and database replication (Active Data Guard or Oracle GoldenGate) provides the optimal solution for providing application-transparent scalability and high availability.

Oracle Sharding becomes the preferred solution when your Oracle Enterprise Architect or the customer has decided that Oracle RAC cannot address current or future scalability and availability requirements (either objectively as a result of scalability and fault isolation tests using projected workloads or subjectively due to customer perception that cannot be changed) and when the application can be explicitly designed for a sharded database.

Note that sharding and Oracle RAC are not mutually exclusive. A shard may use Oracle RAC or RAC One node for HA in place of replication. The only caveat with the first release of Oracle Sharding is that RAC must be configured manually; Oracle Sharding does not provide auto-deployment of a RAC configuration using the DEPLOY command as it does for Data Guard/Active Data Guard and Oracle GoldenGate. Auto-deployment of a RAC configuration is under consideration for a future release

Q: How does Oracle Sharding relate to Oracle MAA?

A: Oracle Sharding is a natural evolution of Oracle MAA designed to address a special use case for massive scalability and complete fault isolation for custom-developed OLTP applications. It adds another tool to the MAA portfolio to address customer scalability and availability requirements. All Oracle MAA principles that apply to an Oracle Database also apply to individual shards that comprise an SDB. Oracle MAA best practices will be extended to address any unique considerations for configuration and management of an SDB.

Q: How does Oracle Sharding relate to Oracle Engineered Systems?

A: Oracle Sharding is hardware platform agnostic as are most other capabilities that fall under the Oracle MAA umbrella of scalability and availability solutions.

Oracle Sharding will benefit from any advantages for database workloads offered by the underlying hardware platform. This means that all of the usual benefits for database performance, availability, and manageability provided by Oracle Engineered Systems also benefit a sharded database running on such systems.

Q: How does Oracle Sharding compare to Oracle Multitenant?

A: Oracle Multi-tenant is a consolidation solution for SaaS and other database applications; Oracle Sharding is a scalability and availability solution for high volume OLTP systems.

Oracle Multitenant supports DBaaS for SaaS applications and for private, public and dev/test clouds. It provides the infrastructure for federating databases together into a logical database to enable high performance querying across all data transparent to the application. Oracle Multitenant provides unique advantage for reducing capital and operational expense by enabling the highest consolidation density for Oracle Database and the simplicity of managing many as one.

Oracle Sharding provides massive scalability and high availability specifically for custom designed OLTP applications. Update transactions must specify a shard key and act upon data within a single shard in order to benefit from the high performance and availability promised by a sharded database architecture. Multi-shard operations or non-shard key access are supported but with a reduced level of performance. Such transactions include simple aggregations, reporting, etc. - ideally less than 10% of total workload for a sharded database.

Q: Does Oracle Sharding support Oracle Multitenant?

A: The initial release of Oracle Sharding (12.2.0.1) does not support Oracle Multitenant. Sharding support for single-tenant container databases is planned for a future release, with support for multi-tenant container databases to follow.

Q: Are there examples of use-cases that include both Oracle Sharding & Multitenant?

A: Yes. When sharding support for Oracle Multitenant is delivered, each shard will be a PDB and will benefit from the same advantages for agility that Multitenant offers any other PDB. Example use-cases include:

- The simplicity of upgrading a shard in a multitenant architecture simply by unplugging from its current CDB and plugging it into a CDB at a higher version.
- The simplicity of migration to and from the Oracle Cloud. An on-premises shard that is a PDB can be unplugged and then plugged into the Oracle Cloud.
- PDB cloning of shards.

Q: Can you shard with Exadata?

A: Exadata has an important play with Oracle Sharding in the following use cases:

- Data Sovereignty and Data Proximity via Geographic Data Distribution: Sharding makes it possible to locate a separate Exadata system as a shard in each country or regions – thus satisfying regulatory requirements where data has to be located in a certain jurisdiction.
- Exadata as a Shard: Each shard can be an Exadata (e.g. 1/4th rack or 1/8th rack) as well. With the inherent scaling advantage of Exadata, a sharded database using Exadata can reduce the number of shards - for example, 10s of shards instead of having 100s of shards. This is a big manageability advantage.
- Shards on Exadata: If you do use Exadata as shards, we recommend sharding across multiple Exadata Machines. For example, instead of placing 100 shards on a single Exadata, you can improve fault isolation by placing 20 shards on each of five Exadata Machines.

Q: What are the advantages of using Oracle Sharding for geo-distribution over deploying separate (non-sharded) databases in each region and have applications/users access databases only in a given region?

A: There are 3 key advantages while doing geo-distribution with Oracle Sharding:

- Manage multiple databases as one database: Oracle Sharding automatically maintains the same database schema in all regions, instead of requiring you to manually make the changes in each region. Also if you want to add another region, you just need to add shards in that region; the schema will be automatically propagated to the new shards. With sharding, the shard catalog knows your logical schema, executes DDLs across shards, adds/removes shards/auto rebalance, etc. And the application has a clean abstraction layer to access the data. The application does not need to know which shards the sharding keys map to, etc.
- Multi-shard Queries: With Oracle Sharding, you have the capability to perform multi-shard queries across all the databases from a central coordinator acting as a proxy. Applications can still access each region directly.
- Automatic replication setup and management: You just need to specify region, shardgroups, shards and the replication factor, and Oracle Sharding automatically configures the replication pipelines and maintains the number of replica chunks. Without sharding, the application and customers would have to maintain and manage the distributed logical database.

In summary, Oracle sharding adds a layer of functionality that makes shards and replicas more like a single logical database - from the perspective of both management and applications.

Q: How does Oracle Sharding compare to Microsoft Azure Elastic Database?

A: Microsoft also offers a sharded database architecture having many of the same objectives as Oracle Sharding. The Microsoft product is cloud-only and is called Azure Elastic Database. Oracle Sharding can be deployed either on-premises or in the Oracle cloud. Oracle Sharding has a number of strengths compared to Microsoft Azure, examples include:

- Microsoft Azure handles multi shard queries by treating shards as external tables and performing all query processing on their coordinator node. This design makes it difficult to scale and requires the movement of a large amount of data on the network between shards and the coordinator node. In contrast, Oracle Sharding multi-shard queries are pushed down to each shard, improving performance and reducing the amount of data transmitted on the network.
- Microsoft Azure also requires the customer to explicitly configure and maintain the list of shards that are used in a multi-shard query. If the configuration changes, it's up to the customer to update the mapping. No such list need be maintained by Oracle Sharding since it maintains this metadata automatically for the customer.
- Microsoft Azure is limited to read-only queries on replica shards. Updates must be directed via database links back to the coordinator database.

Oracle Sharding supports read-only queries to any table and DML updates to temporary tables on Active Data Guard shards and full read-write access to all tables on Oracle GoldenGate shards.

Q: How does Oracle Sharding compare to NoSQL data stores?

A: NoSQL data stores are unable to provide application transparent scalability as Oracle can do for any application using Oracle RAC and Active Data Guard. Similar to Oracle Sharding, however, NoSQL data stores can also use a sharded architecture to achieve the combination of scalability and high availability. Customers who prefer to explicitly design their application for a sharded architecture may perceive that they have the option of using Oracle Sharding with Oracle Enterprise Edition or the alternative of using a NoSQL data store.

Customers will choose Oracle Sharding with Oracle Enterprise Edition for the following reasons:

- NoSQL data stores lack all of the capabilities of an enterprise RDBMS, including: relational schema, SQL, and other programmatic interfaces, support for complex data types, online schema changes, multi-core scalability, advanced security, compression, extensive high-availability features, ACID properties, consistent reads, the developer agility of JSON, and much more. For example, Oracle's inherent support for transactions means concurrent updates/reads never get inconsistent results; NoSQL data stores cannot do this. Oracle also extends read consistency to multi-shard operations as well using global consistent read, something that is not possible with NoSQL data stores.
- Oracle Sharding combines Oracle Enterprise Edition with a comprehensive solution for deploying a sharded architecture that includes automation to simplify many aspects of life-cycle management, advanced partitioning methods for increased flexibility, and intelligent data-dependent routing for superior run-time performance.

The combination of Oracle Sharding and the Oracle RDBMS provide customers with the best of both worlds: the ability to massively scale using a sharded database architecture without the compromises of a NoSQL data store.

Q: When would I use Oracle Sharding versus Oracle NoSQL?

A: Customers evaluating simple key-value NoSQL data stores should:

- Choose Oracle Sharding if they see value in the combination of Oracle Enterprise Edition and a comprehensive set of capabilities for deploying a sharded architecture for extreme scalability and availability.
- Choose Oracle NoSQL if they do not place value on the capabilities of Oracle Enterprise Edition and instead are seeking the lower cost of ownership of a NoSQL solution designed to provide highly reliable, scalable and available data storage across a configurable set of systems that function as storage nodes.

More Information

Q: Where can I find more information on Oracle Sharding?

A: For more information, please see the Oracle Sharding page on Oracle Technology Network - www.oracle.com/goto/oracle/sharding . A variety of helpful information is available online including white papers, cookbooks, customer presentations, end-user documentation, and links to blog sites etc.

Do follow us on Twitter @OracleSharding

Oracle Corporation, World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065, USA

Worldwide Inquiries
Phone: +1.650.506.7000
Fax: +1.650.506.7200

ORACLE

CONNECT WITH US

blogs.oracle.com/blogs
facebook.com/oracle
twitter.com/oracle
oracle.com

Integrated Cloud Applications & Platform Services