ESP-JETA
ESP Journal of Engineering & Technology Advancements
ISSN: 2583-2646 / Volume 5 Issue 3 July 2025 / Page No: 175-179
Paper Id: JETA-V5I3P123 / Doi: 10.56472/25832646/JETA-V5I3P123

Original Article

Oracle Globally Distributed Database: Scaling Intelligence for Large-Scale Data Analytics

Sarvesh Kumar Gupta
Western Governors University, USA.

Received Date: 10 July 2025 Revised Date: 29 August 2025 Accepted Date: 26 September 2025

Abstract: *This paper provides an in-depth exploration of Oracle's globally distributed database architecture and its transformative role in enabling petabyte-scale analytics. Designed for enterprises operating at a planetary scale, Oracle's globally distributed database integrates strong consistency, low-latency replication, horizontal scalability, and autonomous operations. This paper highlights the core architectural features, engineering advancements, and real-world use cases that demonstrate how Oracle is redefining the future of distributed data analytics.*

Keywords: *Oracle Database, Distributed Systems, Global Scalability, Data Analytics, Cloud Infrastructure, High Availability, Automation.*

I. INTRODUCTION

Enterprises today operate across geographies with ever-growing volumes of data generated in real time. To support global operations and decision-making, there is a need for a unified database system that can handle transactional consistency, analytical performance, and scale. Oracle's globally distributed database is built to address these demands.

Oracle Globally Distributed Database functions as a single logical database composed of multiple Oracle Database shards. Each shard is an independent Oracle Database instance that stores a portion of the overall dataset. These shards can be deployed within a single Oracle Cloud Infrastructure (OCI) availability domain to optimize performance, across multiple availability domains or regions to ensure high availability, or distributed across global OCI regions to address data residency and compliance requirements.

The architecture is orchestrated through a central shard catalog, which facilitates automated shard provisioning, centralized management, and execution of cross-shard queries via an intelligent query coordinator. Shard directors further enhance performance by intelligently routing client connections to the appropriate shard based on the provided sharding key, using Oracle's native database drivers.

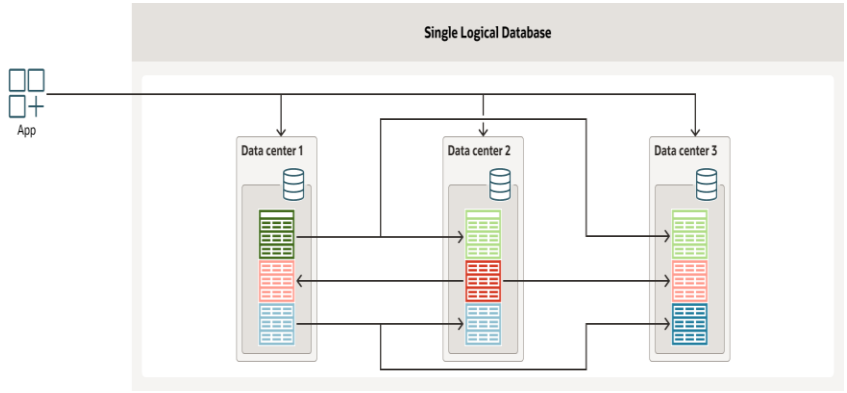

Figure 1 : Oracle Distributed Database, Source[1]

II. ARCHITECTURAL FOUNDATIONS OF ORACLE'S DISTRIBUTED DATABASE

- Shared-nothing architecture with intelligent data partitioning
- Global data replication with strong consistency
- Separation of compute and storage
- Support for multi-region, multi-cloud deployments

- Advanced indexing and query optimization across geographies

III. SHARDING TYPES

It comes with the flexibility to choose the right sharding method based on application & data-model:

- System Managed — Partition data across shards by CONSISTENT HASH. Data sharded/re-sharded automatically. Data is evenly distributed across shards.
- User Defined (Range, List) — Partition data across shards by RANGE or LIST. Ranges or lists of sharding key values are assigned to shards by the user. User needs to maintain balanced data distribution. More efficient range queries
- Composite — Data is partitioned across Shardspaces by LIST or RANGE on super-sharding key (e.g. geography). Within each shardspace, data is partitioned across shards by CONSISTENT HASH using sharding key (e.g. customer id). For geo-distribution or hybrid clouds with linear scalability

IV. AUTOMATED DATA DISTRIBUTION

- Oracle Globally Distributed Database automatically places data on the desired shard, saving time and eliminating manual data preparation.
- Multiple sharding methods (system-managed and user-defined)
- Composite sharding, which allows two levels of sharding with different sharding methods and keys
- Parallel data ingestion and parallel queries on all shards.

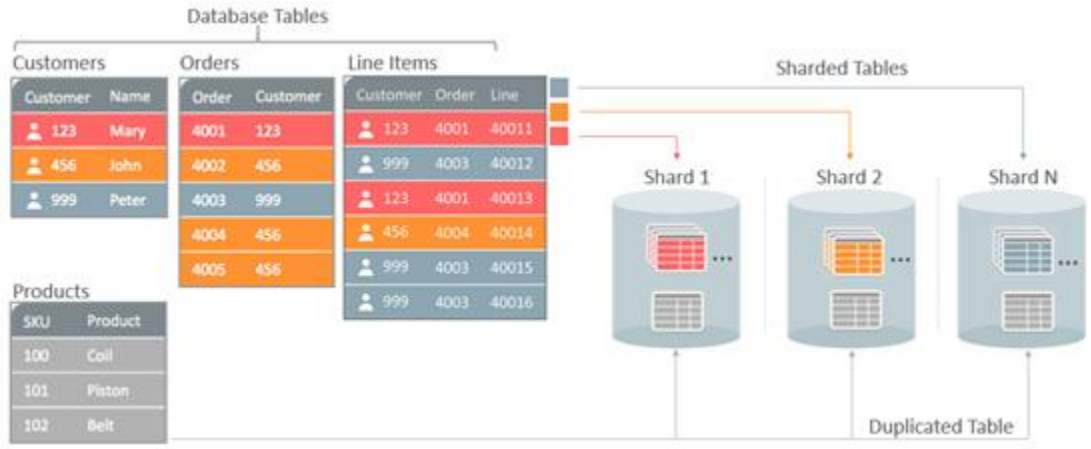

Figure 2 : Sharded & Duplicated Tables, Source[2]

V. DATA MODEL

Sharded table partitions are distributed across shards at the tablespace level, based on a sharding key. Examples of keys include customer ID, account number, and country ID.

Sharding keys must adhere to the following characteristics.

- The sharding key should be very stable; its value should almost never change.
- The sharding key must be present in all of the sharded tables. This allows the creation of a family of equi-partitioned tables based on the sharding key.
- Joins between tables in a table family should be performed using the sharding key.

Sharding Keys for System-Managed Distributed Databases

For the system-managed data distribution method, the sharding key must be based on a column that has high cardinality; the number of unique values in this column must be much bigger than the number of shards. Customer ID, for example, is a good candidate for the sharding key, while a United States state name is not.

A sharding key can be a single column or multiple columns. When multiple columns are present, the hash of the columns are concatenated to form the sharding key.

Schema design:

The following examples create a sharded table called Customers and specify that columns cust_id and name form the sharding keys for the table.

```
CREATE SHARDED TABLE customers
(cust_id     NUMBER NOT NULL
, name        VARCHAR2(50)
, address     VARCHAR2(250)
, region      VARCHAR2(20)
, class       VARCHAR2(3)
, signup      DATE,
CONSTRAINT cust_pk PRIMARY KEY(cust_id, name))
PARTITION BY CONSISTENT HASH (cust_id,name)
PARTITIONS AUTO
TABLESPACE SET ts1;

CREATE SHARDED TABLE Orders
( OrderNo   NUMBER NOT NULL
, CustNo    NUMBER NOT NULL
, Name      VARCHAR2(50) NOT NULL
, OrderDate DATE
, CONSTRAINT OrderPK PRIMARY KEY (CustNo, Name, OrderNo)
, CONSTRAINT CustFK  FOREIGN KEY (CustNo, Name) REFERENCES Customers(Cust_ID, Name)
)
PARTITION BY REFERENCE (CustFK);
```

VI. ENABLING ANALYTICS AT SCALE

- Massively parallel query execution across global nodes
- Support for mixed workloads (OLTP + OLAP)
- Integrated machine learning and analytics functions
- In-memory acceleration and tiered storage
- Version 23 ai has vector search enhancement to support similarity search.

Figure 3 : Oracle GDD runs everywhere , Source[3]

VII. AUTOMATION AND RESILIENCE

- Auto-sharding and rebalancing
- Self-healing clusters and adaptive query optimization
- Real-time failover and zero-downtime maintenance
- Autonomous capabilities for tuning, patching, and scaling

VIII. REAL-WORLD APPLICATIONS

- Financial institutions running risk models across continents
- Global e-commerce platforms handling transactions and personalization in real time
- Telecommunications providers managing distributed subscriber analytics
- Government and research agencies with geo-distributed sensor data

IX. ROUTING

A. Direct and Proxy routing for OTLP & Reporting requirements

Direct Route (Single Shard Query):

- Fast path for sharding key based access (OLTP)
- Connection pool maintains the shard topology cache
- Upon first connection to a shard-
- Connection pool retrieves all sharding key ranges in the shard
- Connection pool caches the key range mappings
- DB request for a key that's in any of the cached key ranges goes directly to the shard (i.e. bypasses shard director)

Proxy Route (Multi-shard query):

- Multi-shard query or query without sharding key
- Applications connect to Query coordinator/Shard Catalog
- Coordinator rewrites the query to do most processing on the shards
- Supports shard pruning and scatter-gather
- Final aggregation performed on the coordinator

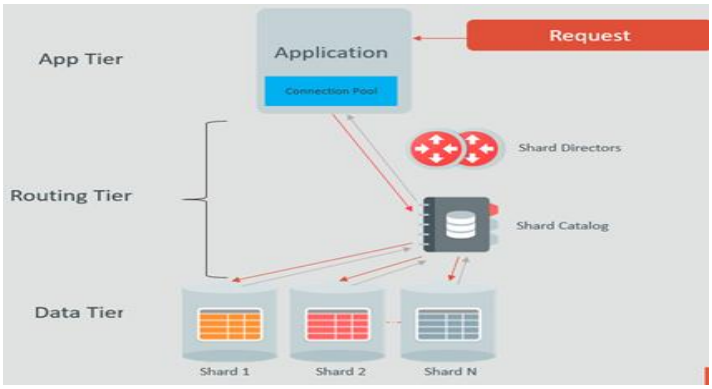

Figure 4 : Routing , Source[4]

X. PERFORMANCE AND BENCHMARKING

- Sub-second query responses over petabyte-scale datasets
- Proven linear scalability under increasing workloads
- Cloud-native performance enhancements on Oracle Cloud Infrastructure (OCI)

XI. CHALLENGES AND ENGINEERING SOLUTIONS

- Data gravity and cross-region latency mitigated via optimized replication
- Ensuring consistency and availability through Paxos-based protocols
- Handling schema evolution and versioning in globally distributed schemas

XII. CONCLUSION

Oracle's globally distributed database offers a powerful solution to modern enterprises seeking to scale analytics globally without compromising consistency, performance, or reliability. By leveraging deep integration of automation, resiliency, and intelligent data services, it sets a new standard for distributed data management.

XIII. REFERENCES

[1] https://www.oracle.com/autonomous-database/distributed-autonomous-database/
[2] Sander Almekinders https://trusharborse.medium.com/internet-scale-application-using-oracle-database-sharding-b8017d3df2a1
[3] https://docs.oracle.com/en/database/oracle/oracle-database/23/shard/sharding-keys.html#GUID-2341646A-5039-4202-A8BF-D0FD082004FD
[4] Trushar Borse https://trusharborse.medium.com/internet-scale-application-using-oracle-database-sharding-b8017d3df2a1
[5] https://docs.oracle.com/en/database/oracle/oracle-database/23/shard/oracle-globally-distributed-database-guide.pdf