Oracle® AI Database

Oracle AI Database New Features

Release 26ai
G43730-13
October 20, 2026

ORACLE

Oracle AI Database Oracle AI Database New Features, Release 26ai

G43730-13

Copyright © 2022, 2026, Oracle and/or its affiliates.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software, software documentation, data (as defined in the Federal Acquisition Regulation), or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software," "commercial computer software documentation," or "limited rights data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle®, Java, and MySQL are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

1 Introduction

Oracle AI Database 26ai is the next long-term support release of Oracle Database. It includes over 300 new features with a focus on artificial intelligence (AI) and developer productivity.

About

About Oracle AI Database 26ai

Oracle AI Database 26ai is the next long-term support release of Oracle AI Database. It includes over 300 new features with a focus on artificial intelligence (AI) and developer productivity. Features such as AI Vector Search enable you to leverage a new generation of AI models to generate and store vectors of documents, images, sound, and so on; index them and quickly look for similarity while leveraging the existing analytical capabilities of Oracle AI Database. This combined with the already extensive set of Machine Learning algorithms enables you to quickly create sophisticated AI-enabled applications. Oracle AI Database 26ai also uses AI to optimize many of the key database functions to make more accurate estimates on timings and resource costings.

New developer-focused features now make it simpler to build next-generation applications that use JSON or relational development approaches or both interchangeably. New microservice and messaging functionality improves upon Oracle AI Database's extensive support for this key design methodology. If you need to distribute or shard your database because of regulatory or performance requirements, Oracle AI Database 26ai adds new RAFT protocol support to make it easier than ever before.

Oracle AI Database 26ai also includes significant improvements to SQL and PL/SQL, introducing new data types and language enhancements to create new or improve existing OLTP or analytical applications. While Oracle AI Database is widely regarded as the most secure database in the industry, many new capabilities, such as SQL Firewall, enable you to control exactly what SQL is executed against your database.

To help DBAs, Oracle AI Database 26ai further refines many of the key management tasks, reducing their complexity and improving their performance as well as introducing new functionality to simplify tasks, such as reclaiming free space in tablespaces. Oracle AI Database also adds new performance improvements both at an infrastructural level (with technologies like True Cache) and at the SQL level, ensuring some statements will execute many times faster.

Note: For information about desupported features, see Oracle AI Database Changes, Desupports, and Deprecations.

Feature Highlights

AI Vector Search

Oracle AI Vector Search is designed for Artificial Intelligence (AI) workloads and allows you to query data based on semantics, rather than keywords.

See Oracle AI Vector Search User's Guide.

JSON Relational Duality

Data can be transparently accessed and updated as either JSON documents or relational tables.

Developers benefit from the strengths of both, which are simpler and more powerful than Object Relational Mapping (ORM).

See Overview of JSON-Relational Duality Views.

Operational Property Graphs in SQL

Developers can now build real-time graph analysis applications against operational data directly in the Oracle AI Database, utilizing its industry leading security, high availability and performance capabilities.

See Support for the ISO/IEC SQL Property Graph Queries (SQL/PGQ) Standard.

Microservice Support

Alongside Oracle's already comprehensive support for microservices, new functionality makes it simpler to implement cross-service transactions.

See Microservices.

Lock-Free Reservations

Lock-free column value reservations allow applications to reserve part of a value in a column without locking the row; for example, reserve part of a bank account balance

or reserve an item in inventory without locking out all other operations on the bank account or item.

See Lock-Free Reservation.

Kafka APIs for TxEventQ

Kafka applications can now run directly against the Oracle AI Database with minimal code changes, leveraging high performance Transaction Event Queues (TxEventQ).

See Kafka APIs for Oracle Transactional Event Queues.

JavaScript Stored Procedures

Developers can now create stored procedures using JavaScript in the database. This functionality also allows developers to leverage the huge number of JavaScript libraries.

See JavaScript.

Priority Transactions

Low priority transactions that block high priority transactions can be automatically aborted. This feature reduces the administrative burden on the DBA while maintaining high transaction throughput.

See Priority Transactions.

Data Use Case Domains

Data use case domains allow developers to declare the intended usage of data (columns) in a centralized and light-weight manner. For example, you can declare a column to hold an email, URL, password, currency, and so on. Applications can use data use case domains to automatically generate code or verify values.

See Data Use Case Domains.

Many Data Type and SQL Enhancements

The following features are among the many data type and SQL enhancements:

- SQL BOOLEAN Data Type
- Direct Joins for UPDATE and DELETE Statements

- Unicode 15.0 Support
- SELECT Without FROM Clause
- GROUP BY Column Alias or Position

Improved Machine Learning Algorithms

New improvements to Oracle In-Database Machine Learning algorithms make it simpler to categorize text and data while offering better performance and flexibility.

See Machine Learning - Enhancements.

Sharding Enhancements

New functionality makes it simpler to create and manage shard replicas. New sharding models also improve the distribution of data for shard keys with few unique values.

See Using Raft Replication in Oracle Globally Distributed Database.

Schema Privileges to Simplify Access Control

System privileges can now be granted at the schema level. This feature simplifies the privilege management process and as a result, makes it easy to secure databases.

See Managing Schema Privileges.

Developer Role for Application Developers

A new role allows administrators to quickly assign developers only the privileges they need to design, build, and deploy applications for the Oracle AI Database.

See Use of the `DB_DEVELOPER_ROLE` Role for Application Developers.

Oracle SQL Firewall Included in Oracle AI Database

Included in Oracle AI Database, SQL Firewall provides real-time protection against common database attacks by monitoring and blocking unauthorized SQL and SQL injection attacks, no matter the SQL execution path.

See Overview of Oracle SQL Firewall.

Azure AD OAuth2 Integration

New functionality enables single sign-on to Oracle AI Database service instances or on-premises Oracle AI Databases from Microsoft Azure Cloud.

See JDBC Support for OAuth 2.0 Including OCI IAM and Azure AD.

Oracle AI Database as a Redis Server

Oracle AI Database provides Redis server–compatible functionality. It supports storing and caching Redis objects as database objects and automatically synchronizing them when the underlying database objects change. The Oracle Redis Server can be configured on Oracle AI Database, Oracle Active Data Guard, and Oracle True Cache, offering added flexibility.

See Using Oracle True Cache as a Redis Server.

2 AI Vector Search

Oracle AI Vector Search is designed for Artificial Intelligence (AI) workloads and allows you to query data based on semantics, rather than keywords.

General

AI Vector Search: JDBC Support Completion for SPARSE Vectors with a Binary Format

An `oracle.sql.VECTOR.SparseBooleanArray` interface is added to represent `SPARSE BINARY VECTOR` data.

- Instances of `SparseBooleanArray` can be passed to `setObject` methods of `PreparedStatement` to create `SPARSE BINARY VECTOR` bind values.
- Instances of `SparseBooleanArray` can also be returned by `getObject` methods of `ResultSet`. This conversion is supported for all `VECTOR` dimension types, not just `BINARY`.

This feature brings productivity and performance to Java applications using `VECTOR` with a `SPARSE BINARY` format.

View Documentation

Vector Data Type

Vector Data Type

This feature provides a built-in VECTOR data type that enables vector similarity searches within the database.

With a built-in VECTOR data type, you can run AI-powered vector similarity searches within the database instead of having to move business data to a separate vector database. Avoiding data movement reduces complexity, improves security, and enables searches on current data. You also can run far more powerful searches with Oracle AI Vector Search by combining sophisticated business data searches with AI vector similarity search using simple, intuitive SQL and the full power of the converged database - JSON, Graph, Text, Spatial, Relational and Vector - all within a single query.

View Documentation

AI Vector Search: SQL Execution

AI Vector Search SQL Execution adds SQL execution support for vector indexes built on vector columns inside the database. In addition, it provides support for SQL Functions related to the vector type and allow for row level restriction capabilities in SQL queries for partitions.

This feature allows you to more easily build with the vector data type, enabling the rapid development of AI-driven applications.

View Documentation

Sparse Vectors

Sparse Vectors are vectors that typically have large number of dimensions, but only a few dimensions have non-zero values. These vectors are often produced by sparse encoding models such as SPLADE and BM25. Conceptually, each dimension in a sparse vector represents a keyword from a specific vocabulary. For a given document, the non-zero dimension values in the vector correspond to the keywords (and their variations) that appear in that document.

Sparse vectors, such as those generated by models like SPLADE and BM25, often outperform dense vectors from models such as BERT, in terms of keyword sensitivity and effectiveness in out-of-domain searches. This superior performance is especially valuable in applications where precise keyword matching is crucial, like in legal or academic research. Additionally, sparse vectors are often used for Hybrid Vector Search, where they can be used alongside dense vectors to combine semantic searches and keyword searches, and provide more relevant search results.

View Documentation

AI Vector Search: Arithmetic and Aggregate Operations

Just as you can add (+), subtract (-), or multiply (*) dates, timestamps, intervals, and numbers, you can now apply these arithmetic operators to vectors. The arithmetic operation is performed at each dimensional element of the vectors. It's also possible to calculate the SUM or AVG of a set of vectors.

Arithmetic operations on vectors allow AI systems to manipulate and combine abstract concepts, which enhances their ability to understand and process language or data in more sophisticated ways.

View Documentation

AI Vector Search: Optimizer

This functionality adds support to the Optimizer to use indexes built on the new Vector data type rather than doing full table scans.

The support for vector indexes being used by the Optimizer allows for efficient computation of vector queries enabling developers to build the next generation of AI-powered solutions.

View Documentation

AI Vector Search: PL/SQL

This functionality adds a new vector type to the PL/SQL type system, along with a set of vector operations useful for performing similarity searches on sets of vectors.

Support for the new vector data type in PL/SQL opens up new possibilities for developers to build robust and efficient AI-driven applications.

View Documentation

BINARY Vector Dimension Format

BINARY is a new dimension format that can be used with the `VECTOR` data type. Each dimension of a BINARY vector can be represented with a single bit (0 or 1). A BINARY vector itself is represented as a packed UINT8 array, for example, a single UINT8 value represents 8 dimensions of the BINARY vector. BINARY vectors can be generated using embedding models provided by Cohere (for example, embed v3), Hugging Face Sentence Transformers, and so on.

BINARY vectors offer two key benefits compared to FLOAT32 vectors:

1. The storage footprint of BINARY vectors is 32X lesser, and
2. Distance computations on BINARY vectors can be up to 40X faster, which accelerates Vector Search

BINARY vectors can provided reduced accuracy compared to FLOAT32 vectors. But, evaluations on various datasets have shown that they can still achieve 90% or higher accuracy of FLOAT32 vectors.

View Documentation

Client Support for SPARSE Vectors

Sparse vectors are now supported by JDBC and OPD.NET client drivers.

Sparse vectors are vectors that typically have a large number of dimensions, but only a few of those dimensions have non-zero values. Because sparse vectors only store non-zero values, their use can improve efficiency and save storage space. ODP.NET and JDBC now supports sparse vector types enabling efficient interaction with sparse vector data through these client drivers.

View Documentation

JDBC Support for Vector Data Type

This feature adds the necessary components to the JDBC drivers to support the AI Vector Search data type including SQLType, DatabaseMetaData, ResultSetMetaData and ParameterMetaData, VectorMetaData, Java to SQL Conversions with PreparedStatement and CallableStatement, SQL to Java Conversions with CallableStatement, SQL to Java Conversions with CallableStatment and ResultSet, and VECTOR Datum class.

JDBC Support for the Vector data type enables developers to build robust, scalable, and high-performance Java applications with Artificial Intelligence focus.

View Documentation

New SQL Functions: VECTOR_ORIGIN, VECTOR_RANDOM, and VECTOR_RANDOM_PER_ROW

The `VECTOR_ORIGIN` SQL function creates a vector with all of the dimension values set to zero. This represents the origin for a vector space with n-dimensions. The `VECTOR_ORIGIN` SQL function is useful as a bind variable for prototyping as it will always work for any vector dataset.

The `VECTOR_RANDOM` SQL function generates a vector with random values for all dimensions for all rows associated with the SQL statement.

The `VECTOR_RANDOM_PER_ROW` SQL function generates different random values for every row. This is useful for generating random vector data.

These SQL functions make it easy to generate random vector data and vector bind variables for proof of concept development.

View Documentation

Oracle Call Interface Support for Vector Type

Oracle Call Interface (OCI) now supports Vector data type. Applications that use OCI can now take advantage of the new Vector data type in the Oracle AI Database.

This feature ensures that you can leverage the full capabilities of the Oracle AI Database through OCI-based applications to create the next generation of AI-powered solutions.

View Documentation

PL/SQL BINARY Vector Support

PL/SQL supports `BINARY` as a new dimension format for the vector type, in line with SQL.

`BINARY` vectors are frequently used to represent whether some entity, such as a textual document, does or does not include certain features, list words, or terms. The advantages of the `BINARY` format are two-fold. The storage footprint of vectors can be reduced 32X compared to the default `FLOAT32` vectors and distance computations on `BINARY` vectors are up to 40X faster.

Support for `BINARY` vectors means that PL/SQL is able to handle a binary vector in the same way that it supports a vector of any other dimension format.

View Documentation

Set COMPATIBLE to 23.6.0 to Enable New AI Vector Search Features in 26ai

The new AI Vector Search features are available in Oracle AI Database 26ai, Release Update 23.6 or later, provided the `COMPATIBLE` parameter is set to 23.6.

Regarding the `COMPATIBLE` parameter:

1. New customers installing Oracle AI Database 26ai, Release Update 23.6 directly:
    - A fresh installation of Release Update 23.6 and using db create comes with the `COMPATIBLE` parameter set to 23.6.0 by default.

    - Post installation, you will be unable to downgrade `COMPATIBLE` parameter to a lower value later.
2. Customers on Oracle AI Database 26ai, Release Update 23.4 and 23.5:
    - Before patching, the `COMPATIBLE` parameter can be 23.0.0 or 23.4.0 or 23.5.0.
    - When you patch to Release Update 23.6, to access the new AI Vector Search features, you must manually set the `COMPATIBLE` parameter to 23.6.0.
    - Note: In Release Update 23.6. updating the `COMPATIBLE` parameter requires downtime. It is not automatically done as part of patching. You must choose to update the database `COMPATIBLE` setting.
3. Customers on Oracle Database 19c or 21c:
    - When you upgrade the database to Oracle AI Database 26ai, the `COMPATIBLE` parameter remains as previously set in the source database.
    - If you want to use the new AI Vector Search features, you must manually set the `COMPATIBLE` parameter to 23.6.0.
    - Note: The minimum `COMPATIBLE` setting permitted for upgrading to 26ai is 19.0.0. In other words, you cannot upgrade directly from Oracle Database 12c or 18c to Oracle AI Database 26ai.

View Documentation

Sharding Support for AI Vector Search

With Globally Distributed Database support for Vector Search, tables containing vectors are automatically distributed and replicated across a pool of Oracle databases that share no hardware. Similarity searches are automatically parallelized across shards or directed to a specific shard if the sharding key is provided.

Globally Distributed Database AI Vector Search offers several benefits, including greater scalability by allowing vectors to be distributed across multiple machines, improved performance by parallelizing vector searches across shards, and improved data resilience because if one shard goes down, the other shards can continue to operate. It also allows vector search to be deployed as part of a distributed database, where a single logical database is distributed over multiple geographies.

View Documentation

Sparse Vector Support in PL/SQL

Sparse vectors are now supported in PL/SQL.

Sparse vectors are vectors that typically have a large number of dimensions, but only a few of those dimensions have non-zero values. Because sparse vectors only store non-zero values, their use can improve efficiency and save storage space. Native support in PL/SQL allows sparse vectors to be created and used directly from within PL/SQL.

View Documentation

Support of Vector Data Type in JSON Type (OSON)

This functionality extends the standard JSON scalar types, to include the new Vector data type. It is fully supported by all Oracle JSON constructs, and a vector scalar JSON value is convertible to/from a JSON array of numbers.

Embedding vector values in JSON-type data is important for interoperability between SQL values and JSON values. For example, a table with a VECTOR column can be exposed in JSON data without a loss of data-type information allowing developers to create the next generation of AI applications.

View Documentation

User-Defined Vector Distance Functions

AI vector search supports custom, user-defined distance metrics. Proprietary and/or domain-specific distance metrics can be used in addition to the standard built-in distance metrics.

Vector search operations are often based on standard distance metrics such as Euclidean, Cosine, and Dot Product. However, there are situations where domain-specific or proprietary metrics may be required. User-defined vector distance functions allow users to create their own custom metrics using JavaScript functions.

View Documentation

Vector Data Type Support in External Tables

Vector data stored outside the database (on a file system or cloud object store) can be easily accessed to perform similarity searches using external tables.

Vector embeddings created outside the database can be loaded quickly and easily using an external table and standard SQL. It is also possible to run similarity searches

on vector embeddings stored outside the database and seamlessly combine those searches with data that is stored inside the database.

View Documentation

Vector Indexes

Hybrid Vector Index

The Hybrid Vector Index (HVI) is a new index that allows users to easily index and query their documents using a combination of full text search and semantic vector search to achieve higher quality search results.

The Hybrid Vector Index (HVI) simplifies the process of transforming documents into forms that are amenable both for vector similarity search and for textual search through a single index DDL.

The HVI provides a unified query API that allows users to run textual queries, vector similarity queries, or hybrid queries that leverage both of these approaches. This lets users easily customize the search experience and enhances search results.

View Documentation

Vector Indexes

Vector Indexes are a class of specialized indexing data structures that are used to efficiently store and search high-dimensional vector data. A vector index organizes vector data in a manner such that similar items (where similarity is defined by distance between two vectors) are grouped together, thus, making the search process extremely efficient. Unlike traditional database indexes, vector indexes are commonly used on large datasets to perform approximate similarity searches that can trade-off between query accuracy and query performance depending on the application's requirements.

This functionality enables efficient similarity searches and faster query performance for AI-driven applications. In addition, vector indexes scalability and support for high-dimensional data improve analytical insights and can lead to informed decision-making and a competitive business advantage.

View Documentation

Partition-Local Neighbor Partition Vector Index

This feature enables LOCAL indexing for Neighbor Partition Vector Indexes, optimizing search performance for partitioned tables. This feature conceptually creates a dedicated vector index for each partition, allowing queries with partition key filters to search only the relevant index partitions. As a result, vector searches are more efficient, leading to significantly lower response times when querying large partitioned datasets.

Large enterprise datasets are frequently partitioned by relational attributes to optimize performance. By enabling LOCAL Neighbor Partition Vector Indexes, users benefit from enhanced scalability and accelerated query performance through partition pruning. This approach also ensures more efficient data lifecycle management, making it ideal for handling large-scale enterprise workloads.

View Documentation

Persistent Neighbor Graph Vector Indexes

The HNSW vector index is an inmemory resident multi-layered graph index. The time taken to recreate the inmemory graph on a restart can be improved by having a disk checkpoint image of the graph. This feature adds the checkpoint format as well as the framework to take a disk checkpoint and then use it to recreate the inmemory resident graph structure.

Getting an index access plan after a restart can take a long time. A higher priority disk checkpoint based reload execution improves the time taken to get an index access plan after a restart.

View Documentation

Transactional Support for Neighbor Graph Vector Indexes

HNSW Index is an in-memory hierarchical graph index for vector data. In Oracle AI Database 26ai, Release Update 23.4 and 23.5, DMLs were not allowed on tables that have HNSW index built on their vector column(s). This feature enables transactions to be executed on such tables. Moreover, vector search queries that use the HNSW Index will see transactionally consistent results, based on their read snapshot. Transactional consistency is guaranteed even on Oracle RAC where the HNSW Index is duplicated on all instances in the Cluster, DMLs occur on one or more instances in the Cluster, and search queries can be executed on any instance in the Cluster.

HNSW Index is the fastest vector search index that Oracle offers in Oracle AI Database 26ai. Thus, customers want to use HNSW Index for search queries, while also issuing DML modifications on relational or vector columns in the underlying table. Since DMLs may render the in-memory HNSW Index structures stale, special protocols are added in this project to guarantee transactionally consistent results for customers.

View Documentation

AI Vector Search: Scalar Quantized IVF Indexes

Scalar quantization for a vector index is a method of compressing or approximating high-dimensional data vectors by independently quantizing each dimension (or scalar component). Scalar quantization is supported on a global IVF index.

Scalar quantization over an IVF index drastically reduces storage requirements while maintaining essential data fidelity to retain semantic relevance. It is typically used when simplicity and low storage requirements are priorities.

View Documentation

Additional Predicate Support with Hybrid Vector Search

Hybrid vector search queries can now have additional `WHERE` clause predicates on columns other than the indexed columns.

Hybrid vector search combines vector-distance and text-based search in a single query. There are situations where it is beneficial to add an additional filter predicate on columns that are not covered by the vector or text-based indexes. The `FILTER_BY` field provides a method to supply additional filter predicates using standard SQL operators.

View Documentation

Auto IVF Vector Index Reorganization

IVF vector indexes can be automatically reorganized after the database system has determined that a rebuild is necessary. This can be done without the need for human intervention.

Over time, the performance of IVF vector indexes can degrade as the number of datapoints mapped to a centroid increases, or the mapping of the data to centroids becomes skewed. In order to overcome this situation, the IVF vector index should be

reorganized. This feature ensures that you do not have to manually determine the thresholds to determine if a rebuild is necessary; instead, the system does it for you.

View Documentation

DBMS_SEARCH Autocomplete for Hybrid Vector Indexes

Autocomplete uses Oracle Text indexing to provide low latency, high quality matches on a designated column (or JSON field), and can optionally use fuzzy matching.

- Faster UX: Suggests results after only a few characters.
- Higher relevance: Uses Oracle Text ranking to return the best matches first.
- Flexible data sources: Works on regular table columns or JSON fields.
- Two search modes: General text completion or name-style completion.
- Optional fuzzy matching: Captures near-miss spellings.

Autocomplete helps users discover content faster by returning relevant suggestions as they type. It's ideal for search bars, name lookups, product catalogs, and any UI that requires suggestions instantly.

View Documentation

Distributed HNSW Indexes with RAC

Distributed HNSW indexes on Real Application Cluster databases can now seamlessly scale across the total memory available in all instances of a RAC database.

HNSW vector indexes reside in the Vector Memory Pool in the Database's SGA, which is limited by the amount of memory available in a physical database server. With RAC, the SGA and CPU count for a database can be scaled across multiple instances. With distributed HNSW indexes in RAC, the vector memory pool can now span multiple RAC instances. This means larger HNSW indexes are supported, and the number of CPUs available for parallel operations also increases.

View Documentation

Duplicated HNSW Vector Indexes on RAC

HNSW Vector Indexes are now supported on RAC environments through full duplication on all instances of the cluster that have sufficient memory in the Vector Pool. On Oracle Autonomous AI Database Serverless deployments, the Vector Pool is autonomously managed.

All copies of the HNSW index across different RAC instances share the same ROWID-to-VID mapping table on disk. However, each instance builds its in-memory neighbor graph independently, and hence, its possible to get different results for approximate searches depending on which RAC instance is used to serve the query.

Enterprise customers often deploy Oracle AI Database in RAC environments. This feature enables creation of HNSW vector indexes for RAC through full duplication across all instances of the cluster. Queries directed at any instance of the RAC cluster can take advantage of HNSW vector index plans resulting in ultra-fast similarity searches.

View Documentation

Hybrid Vector Index for JSON

Hybrid Vector Indexes allow document retrieval by integrating full-text search capabilities with semantic vector search techniques, resulting in higher-quality search results. This powerful feature has now been extended to support JSON columns, offering greater flexibility in data indexing and querying.

Creating a Hybrid Vector Index on a JSON column offers a unified query API that enables users to execute various types of searches:

- Textual queries
- Vector similarity queries
- Hybrid queries that leverage both approaches

This versatile functionality allows users to:

- Easily customize the search experience
- Significantly enhance the quality and relevance of search results

View Documentation

IVF Index Online Reorganization

The quality of an IVF index may degrade over time if updates to the base table alter the general vector distribution. It is now possible to reorganize an IVF index while it remains available for DMLs and Queries.

IVF indexes can become unbalanced if the source table changes significantly from when the index was originally created. This can potentially impact the performance and quality of the index. With IVF Index online reorganization, it is possible to reorganize the structure of the index while the index remains online and available for DMLs and queries.

View Documentation

Included Columns Support for Distributed HNSW Indexes

Distributed HNSW queries may need to revisit the table after finding the nearest neighbors to retrieve requested attributes. Including commonly used columns in the index lets queries return those values directly, reducing extra table lookups and speeding up responses.

This helps AI applications retrieve relevant records and the business data they need in one efficient search.

View Documentation

Included Columns Support for JSON, BLOB and CLOB Data Types

Included Columns in IVF (Neighbor Partition) vector indexes can now be of type `JSON`, `BLOB` or `CLOB`.

Included Columns permit additional non-vector columns in a base table to be stored in an IVF (Neighbor Partition) vector index. By storing the extra columns in the index, the query execution no longer needs an additional table access to retrieve the underlying columns from the base table.

View Documentation

Included Columns in Neighbor Partition Vector Indexes

Included columns in vector indexes facilitate faster searches with attribute filters by incorporating non-vector columns within a Neighbor Partition Vector Index. This feature optimizes query execution by removing the need to access these columns from the base table.

Sophisticated workloads often combine business data search on relational columns with vector similarity search. Having included columns in Neighbor Partition Vector

Indexes significantly enhances enterprise search capabilities by integrating attribute filters with vector-based similarity searches.

This integration facilitates efficient execution of complex queries by:

1. Directly evaluating attribute filters in tandem with vector searches
2. Eliminating the need for base table access through expensive join operations.

Moreover, when the index includes all columns required for a query as covering columns, data can be retrieved directly from the index, thereby accelerating query performance.

View Documentation

Included Columns with HNSW Indexes

HNSW vector indexes can store additional non-vector columns. By storing the extra columns in the index, table access to retrieve the underlying columns from the table is eliminated when performing similarity search queries.

Included columns permit additional non-vector columns in a base table to be stored in a HNSW (In-Memory Neighbor Graph) vector index. By storing the extra columns in the index, the query execution no longer needs an additional table access to retrieve the underlying columns from the base table, resulting in faster similarity search queries.

View Documentation

Incremental Snapshots Support for Distributed HNSW Indexes

Incremental snapshots are now supported on distributed HNSW indexes.

Distributed HNSW indexes currently rely on full repopulation as the refresh mechanism. Full repopulation is expensive and gets progressively slower as DMLs accumulate, and limits the ability to perform fast index searches.

Incremental snapshot support addresses this gap by only applying the changes since the last snapshot. This enables faster refreshes of the index, and is critical for supporting continuously updated AI workloads.

View Documentation

Local HNSW Indexes

Local Partitioned HNSW Vector Indexes deliver all the benefits of enhanced scalability and accelerated query performance through partition pruning. This approach also ensures more efficient data lifecycle management and is well suited for handling large-scale enterprise workloads.

Large datasets are frequently partitioned to optimize performance. Local Partitioned HNSW Indexes deliver a logical HNSW index with an independent HNSW graph per partition. Query performance benefits from partition pruning, while management operations, such as Add, Merge, Split, Exchange, Drop, and Rebuild, add flexibility with minimal performance impact.

By distributing the indexing workload across partitions, Local Partitioned HNSW ensures seamless scalability and efficient resource utilization for large-scale datasets.

View Documentation

Multi-Vector Search with HNSW Indexes

Multi-Vector Search with HNSW Indexes extends Oracle Database approximate vector search to hierarchical queries. Instead of returning one flat top-K list, it finds the best rows within the best partitions across multiple levels—for example, the best sentences from the best paragraphs in the best books. It adds HNSW-based access paths to partitioned row-limiting queries, improving search efficiency while preserving existing SQL syntax. `PARTIAL` and `FULL` promotion modes control how partitions qualify and how many rows are returned. HNSW indexes must include partition-by keys. The implementation reuses existing filtering, visibility, quantization, and rescoring capabilities without on-disk or dictionary changes.

This feature enables applications to express grouped semantic searches directly in SQL, avoiding multiple queries and client-side regrouping. HNSW acceleration supports partition-aware traversal and early termination, improving potential latency and resource efficiency. Reusing existing syntax and infrastructure simplifies adoption, reduces implementation risk, and produces results aligned with entity-level business needs.

View Documentation

Online Index Rebuild

It is possible to create or rebuild a vector index online while the underlying base table also remains open and updatable during the operation.

Vector indexes on tables with frequent updates can become unbalanced and require a rebuild to recalibrate their accuracy. However, for many businesses, the tables and indexes must remain online as they cannot afford any downtime when rebuilding their vector indexes. With online index rebuild, the tables and corresponding vector indexes (HNSW and IVF) can remain open for updates while the vector indexes are created or rebuilt.

View Documentation

Partition Maintenance Operations and Direct Load with Global IVF and HNSW Indexes

Partition Maintenance Operations can now be performed on partitioned tables that have global IVF and HNSW Indexes. These operations can be applied to tables that have been partitioned using various methods, including `RANGE`, `LIST`, `HASH`, and `COMPOSITE`.

Partition maintenance operations such as adding, dropping, merging, and splitting partitions can be performed on tables with global IVF and HNSW Indexes. One of the key benefits of partitioning is the added flexibility of being able to perform maintenance operations on a subset (or partition) of a table in isolation, without impacting the rows in neighboring partitions. This includes tables with Vectors.

View Documentation

Product Quantized HNSW Indexes

Product quantized HNSW indexes can be used to reduce memory requirements and support efficient similarity search on large vector datasets. Product quantization compresses high-dimensional vectors by dividing each vector into smaller subvectors and representing each subvector with the closest centroid from a trained codebook, so that compact centroid identifiers are stored instead of the original vector values. With Oracle AI Database, product quantization can be specified when creating an HNSW index by selecting a desired compression ratio, while the database determines the corresponding quantization parameters. During similarity search, the HNSW index operates on the compressed vector representations, and an optional rescore factor can be used to refine the top-K results using the original, unquantized vectors.

Product quantized HNSW indexes help reduce the memory footprint of vector search workloads, enabling you to index and search larger vector datasets with the same available memory. By storing compact quantized representations instead of full vectors in the HNSW graph, product quantization can provide substantial memory savings while retaining fast approximate nearest neighbor search. You can choose the desired compression ratio based on your workload requirements and use rescoring to improve result quality when needed, providing a flexible trade-off between memory consumption, search performance, and accuracy.

View Documentation

Scalar Quantization Support for Distributed HNSW Indexes

Distributed HNSW indexes support scalar quantization. Scalar quantization enables the compression of vectors into a more compact representation while preserving similarity characteristics. Hence, scalar quantized distributed HNSW indexes can be used to reduce memory requirements.

Distributed HNSW indexes store full-precision vectors, which increases memory usage and limits the amount of data that can be efficiently searched in-memory. As datasets grow, this leads to higher infrastructure costs and reduced query performance due to memory pressure and cache misses.

Scalar quantization addresses this by compressing vectors into a more compact representation while preserving similarity characteristics. This reduces the memory footprint of distributed HNSW indexes, allowing more vectors to reside in-memory and improving search latency.

By enabling efficient storage and faster traversal of the index, scalar quantization is critical for scaling vector search to large, distributed AI workloads without sacrificing performance or cost efficiency.

View Documentation

Scalar Quantized HNSW Indexes

HNSW indexes can be compressed using scalar quantization. Scalar quantization is a technique used to compress or approximate high-dimensional data vectors by independently quantizing each dimension (or scalar component) of the vector. This approach reduces the storage requirements for HNSW indexes and can improve efficiency without significant loss of accuracy.

HNSW vector indexes reside in the vector memory pool of the database SGA. The pool is limited by the amount of memory available in a physical database server. Scalar quantization provides an efficient approach to compress vectors to significantly decrease their storage footprint while also improving efficiency without significant loss of accuracy.

View Documentation

Distance Functions

AI Vector Search: New Vector Distance Metric

AI Vector Search was extended in Oracle AI Database 26ai, Release Update 23.6, to include a new vector distance metric called Jaccard distance.

This feature allows users the option to use another distance metric between vectors.

View Documentation

PL/SQL JACCARD Distance Support

The `JACCARD` distance metric is now supported in PL/SQL, which provides a similarity measure as a value from 0 to 1 between two `BINARY` vectors.

Jaccard distance is a common similarity measure for binary vectors. Support for `JACCARD` as a new metric in the PL/SQL `VECTOR_DISTANCE` operator means that PL/SQL code can make use of the `JACCARD` distance metric from within PL/SQL instead of calling out into SQL.

View Documentation

ONNX Support

Support for ONNX-Format Models as First-Class Database Objects

The Open Neural Network Exchange (ONNX) is an open format to represent machine learning models. It faciliates the exchange of models between systems and is supported by an ONNX runtime environment that enables using models for scoring/inference.

You can import ONNX-format models to Oracle AI Database for the machine learning techniques classification, regression, clustering, and embeddings.

The models will be imported as first-class MINING MODEL objects in your schema. Inference can be done using the family of OML scoring operators, including PREDICTION, CLUSTER, and VECTOR_EMBEDDING.

You can import and use third-party ML models, possibly built in other environments or from other sources, to leverage the database as an ML scoring platform.

Users can invoke these models from SQL queries using the same scoring operators as native in-database models.

While ONNX format models can already be imported to OML Services on Oracle Autonomous AI Database Serverless, you can now use ONNX-format models from Oracle AI Database.

View Documentation

Enable Larger ONNX-Format Models

This feature enables importing Open Neural Network Exchange (ONNX) format models larger than 1 GB for use with the in-database ONNX Runtime. Models include transformers (embedding models) that take text and images as input and produce vectors. This is enabled using "external initializers," which allow referencing model weights (constants) separate from the ONNX graph representation. OML4Py facilitates converting models from Hugging Face or from a local directory, which may contain fine-tuned models.

The ability to use larger models can offer greater accuracy and capture subtle relationships between concepts. It also allows users to leverage a broader set of readily available models and fine-tuned models. External initializers will also enable sharing initializers across database processes for greater memory efficiency. OML4Py automates and simplifies the production of ONNX-format models with external initializers.

View Documentation

Support for Image Transformer Models with AI Vector Search Using the In-Database ONNX Runtime

This feature enables importing and using image transformer models with the in-database ONNX Runtime engine available in Oracle AI Database 26ai. Image transformer models must be in ONNX format and include the required image decoding and preprocessing as part of the ONNX pipeline.

The in-database ONNX Runtime engine with imported transformers enables you to vectorize text and image data directly in the database, eliminating the need to provision, configure, and maintain a separate environment to generate embeddings. Further, it eliminates the need to move data from the database to a separate environment and return vectors to the database. This enhancement expands the set of use cases that you can develop using Oracle AI Database to include image data. This feature is integrated for use with Oracle AI Vector Search for semantic similarity search.

View Documentation

Vector Utilities

Vector Utility API

The Vector Utility API provides a SQL function `VECTOR_CHUNKS` which processes text into pieces (chunks) in preparation for the generation of embeddings to be used with a vector index. The API is configurable in terms of size of chunks and rules for splitting chunks.

While it is possible for you to create your own chunking algorithms, utilizing this functionality could save you time and aid in faster development with a pre-packaged SQL function.

View Documentation

Chainable Utility Functions for Vectors

DBMS_VECTOR provides a set of utility functions for processing text for the creation of vector indexes. These functions may be chained together such that the output from one function is used as the input for the next.

This feature offers a straightforward yet very customizable method for you to turn textual content, like a PDF document or VARCHAR2 database field, into the embeddings necessary for a vector index. This capability enables you to seamlessly develop with vectors, facilitating the creation of the next generation of Artificial Intelligence applications with ease.

View Documentation

Vector Format Output for Feature Extraction Algorithm

Feature extraction algorithms produce a set of features that represent projections in a lower dimensional latent space. The output is typically numerical and dense. VECTOR type representation is the natural choice.

Feature extraction algorithms represent a principled approach to vectorizing relational data. The vectorized representation can be used for similarity search.

View Documentation

In-Database Algorithms Support for VECTOR Data Type Predictors

This feature enables users to include one or more columns of `VECTOR` data type as predictors along with structured enterprise data to in-database machine learning algorithms.

Vector representations of unstructured data can be a powerful input to traditional machine learning algorithms. They enable efficient data processing on text and image data, helping to speed data-driven decision making. Providing vectors as input to machine learning models enables handling a broader class of use cases.

View Documentation

JSON_TEXTCONTAINS and JSON_EXISTS Support in DBMS_HYBRID_VECTOR.SEARCH

The `DBMS_HYBRID_VECTOR.SEARCH` API now supports a way for users to specify optional `JSON_EXISTS` and `JSON_TEXTCONTAINS` clauses as part of their hybrid search.

Writing complex queries requires the ability to further constrain hybrid (semantic and textual) search by `JSON_EXISTS` and `JSON_TEXTCONTAINS` when working with complex JSON data. This feature lets you design sophisticated applications that better satisfy users' search criteria.

View Documentation

Vector Memory Management

Elastic Vector Memory Management

The vector memory pool can be configured to dynamically grow or shrink to accomodate new or changing HNSW indexes.

HNSW vector indexes reside in the Vector Memory Pool in the Database's SGA. Creating an HNSW Index is dependent upon the pool being adequately sized to accommodate the index, and subsequent changes to the underlying base-table can result in the size of the HNSW index changing. With Elastic Vector Memory Management, the pool automatically resizes to accommodate all HNSW Indexes dynamically.

View Documentation

Vector Memory Pool Automatic Management

When using Oracle Autonomous AI Database services (ADB), the Vector Pool dynamically grows and shrinks when HNSW indexes are created or dropped respectively.

Because you cannot explicitly set any SGA-related memory parameters when using Oracle Autonomous AI Database services, this feature automatically maintains the necessary amount of Vector Pool memory needed for your HNSW indexes.

View Documentation

3 Select AI

Oracle Autonomous AI Database Select AI is Oracle's built-in feature that lets you interact with your data using natural language instead of writing SQL, and then uses AI to generate, run, and even explain the query for you.

Extend Select AI Translate for Google, AWS, and Azure Translation Services

Select AI extends the set of translation services supported by the translate action to include services from Google, AWS, and Microsoft Azure. This expands the existing support for the OCI translation service (OCI Language).

This feature enables developers to easily incorporate language translation capabilities into applications for use cases such as translating global news articles, social media content, product documentation, and other content. Translation can be pipelined with other Select AI functionality to handle LLM limitations or customize output for individual users or regions.

View Documentation

Select AI Capabilities

Through the `DBMS_CLOUD_AI` package, Select AI serves as an AI gateway enabling users, developers, and applications to interact with Oracle AI Database using natural language supporting multiple generative AI use cases. Select AI provides functionality supporting natural language to SQL, including automatically generating, running, and explaining SQL; supporting automated vector index creation and the retrieval augmented generation (RAG) workflow; and through the `DBMS_CLOUD_AI_AGENT` package, providing a framework for building, managing, and running AI agents in the database. It also supports synthetic data generation and text summarization and translation. Select AI provides PL/SQL and Python APIs.

Together with the broader foundational Cloud Developer packages, they enable you to integrate AI capabilities directly from the database and support end-to-end workflows that deliver greater data insight with AI.

Select AI enables business users and developers to use a wide range of AI providers and models supporting many AI-enabled use cases and application scenarios. Select AI helps to simplify the adoption of AI in enterprises and organizations across both on-premises and cloud systems. Users get insights faster using natural language without

waiting on manual SQL query generation or workflows implementing retrieval augmented generation. This can reduce IT bottlenecks and accelerate AI-enabled application development. Using Select AI, enterprises benefit from database security mechanisms and protocols.

View Documentation

Select AI Support for NL to Graph Queries

Select AI extends table-based natural language to SQL generation to include native database property graphs. This enables answering questions like "Search for persons or persons they know where both people have linked the same post" or "Find the names of people they know (via 3 to 7 other people)."

Select AI support for native database property graphs lets developers expose graph-based insights using familiar database and natural-language interfaces, reducing custom integration and query-development effort. Business users can explore complex relationships, dependencies, and patterns without learning graph query languages—accelerating analysis and improving decision-making while retaining database security and governance.

View Documentation

Select AI Supports Specifying Custom LLM via Private Endpoint in Customer-Hosted Environment

This feature enables you to run your own LLM on a VM controlled by your organization and configure its private endpoint in an AI Profile for use with Select AI.

You gain greater control over model choice, customization, data residency, and security while minimizing data movement outside your private environment. Developers can integrate organization-specific models through a familiar AI Profile configuration, and business users benefit from AI experiences tailored to their domain and policies.

View Documentation

4 Private AI Services Container

Oracle Private AI Services Container is a robust, secure AI infrastructure that helps regulated organizations, and those with stringent security requirements, address common compliance and auditability challenges of AI model deployments.

Offload Creating Vector Indexes to Remote GPU Container

The Private AI Services Container Vector Index Service allows you to offload the creation of Hierarchical Navigable Small World (HNSW) indexes from the Oracle AI Database to a container running on a remote machine with an NVIDIA GPU.

Accelerate time to insight and reduce database workload by offloading index creation to external GPU-powered processing. This delivers dramatically faster performance without adding complexity, all within the trusted, secure, and easy-to-use Oracle AI Database platform.

View Documentation

Private AI Container for Vector Embedding Generation

The Oracle Private AI Services Container gives Oracle AI Database customers a private, air-gap-capable, OpenAI-style inference layer that keeps embedding work off the database server while still fitting naturally into Oracle-native workflows.

The Private AI Services Container protects Oracle AI Database performance and keeps AI inference inside your security boundary.

View Documentation

Private AI Support for Large Language Models, Reranking, and Text Classification

The Private AI Services Container adds local LLM inference, reranking, and text classification services, enabling chat, RAG, agentic workflows, relevance-ranked retrieval, and label-based text analysis through containerized REST APIs deployable on CPU or GPU infrastructure.

You can build private, production-ready AI applications closer to enterprise data while keeping AI processing within your controlled infrastructure.

By offloading AI workloads from the database to dedicated compute, you can preserve database resources for core operations while adding scalable capabilities for chat, search, classification, routing, sentiment analysis, and other AI-driven workflows.

View Documentation

5 Application Development

Oracle AI Database provides the most comprehensive platform with both application and data services to make development and deployment of enterprise applications simpler.

JSON

JSON-Relational Duality

JSON Relational Duality Views are fully updatable JSON views over relational data. Data is still stored in relational tables in a highly efficient normalized format but can be accessed by applications in the form of JSON documents.

Duality views provide you with game-changing flexibility and simplicity by overcoming the historical challenges developers have faced when building applications using relational or document models.

View Documentation

JSON Schema

JSON Schema-based validation is allowed with the SQL condition `IS JSON` and with a PL/SQL utility function. A JSON schema is a JSON document that specifies allowed properties (field names) and the corresponding allowed data types, and whether they are optional or mandatory.

By default, JSON data is schemaless, providing flexibility. However, you may want to ensure that your JSON data contains particular mandatory fixed structures and typing, besides other optional and flexible components, which can be done via JSON Schema validation.

View Documentation

XML and JSON Search Index Enhancements

The Oracle Text XML search index syntax and JSON search index syntax are now consistent. Additionally, the performance of JSON and XML search indexes has been improved.

Using the same syntax for XML or JSON search indexes and better performance increase productivity.

View Documentation

GoldenGate Replication of JSON-Relational Duality Views

This feature allows developers to use Oracle GoldenGate technology to replicate JSON-relational duality view data as JSON documents, instead of relational tables, from an Oracle AI Database to a target Oracle or non-Oracle database.

Replication of JSON data is an important feature for high availability, fail-over, and real-time migration from a non-Oracle database to Oracle AI Database.

Oracle GoldenGate Replication to non-Oracle databases such as MongoDB (a document database) or Redis (a NoSQL key/value store) is simple and performant with the ability to replicate JSON documents from JSON-relational duality views.

Developers need not write complex JSON and SQL transformations to shape data for relational and document-based updates, or pay the cost of reconstructing application objects on the target database.

View Documentation

JSON-Relational Duality Views: Hidden and Generated Fields

You can map duality view data to *hidden fields*, absent from the view's documents. A *generated field* can use the value of hidden fields.

A document supported by a duality view can be simpler if it need not have a field for every underlying column, in particular for columns needed only for calculating other field values.

View Documentation

JSON-Relational Duality Views: Add Fields With Calculated Values

Duality views generate JSON documents containing objects whose fields map to columns of relational tables. You can add object fields whose values are calculated automatically.

When the database automatically augments objects by adding calculated fields, such calculation need not be done by multiple, separate client applications. And field

calculations can refer to stored data that is not exposed in the resulting objects - a feature similar to redaction.

View Documentation

JSON Collection Views

JSON collection views are special, read-only database views that expose JSON objects in a JSON-type column named DATA.

JSON collection views are conceptually close to JSON-relational duality views, but they have fewer restrictions because they are read only.

View Documentation

JSON Replication

New options in the `logical_replication_clause` of the `CREATE` and `ALTER TABLE` statements let you control how JSON changes are logged. Instead of always logging entire JSON documents, you can now choose to log only the modified portions (`PARTIAL JSON`) or continue logging full documents (`NO PARTIAL JSON`).

Partial JSON logging improves replication efficiency by capturing and transferring only the changed portions of JSON data. This reduces network and storage overhead, accelerates replication, and enables better performance on existing hardware—lowering overall infrastructure costs.

View Documentation

JSON Search Index Path Subsetting

When creating a JSON search index you can specify the fields to include or exclude from indexing: path subsetting.

Path subsetting can reduce the size of a search index and improve its performance.

View Documentation

Replication Support for JSON Collection Tables

JSON Collection Tables can be enabled for logical replication using GoldenGate. Replication is supported to and from JSON Relational Duality Views as well to and from third-party products, such as MongoDB.

Replication is a basic database functionality that works between Oracle databases. It is also used to facilitate online migration to Oracle AI Database from third-party databases, such as MongoDB.

View Documentation

ALTER INDEX Support for JSON Search Index

`ALTER INDEX` is supported for JSON search indexes, which means that you can change index path subsetting without dropping and recreating an index.

Not needing to drop an index and create a new one, to redefine its behavior, increases flexibility and agility for developers.

View Documentation

Additional Flexibility Defining JSON-Relational Duality Views

You can use `field _id` in document subobjects to identify a column that selects document, and use an identity column as an identifying column.

These additional possibilities when defining a duality view provide more kinds of documents that can be supported, and thus allow for more application use cases.

View Documentation

Array Limits for JSON-Relational Duality Views

Developers can limit the number of elements returned in a nested JSON array within a JSON-relational duality view. When the limit value is `1`, the limited nested object can be unnested or represented as a singleton subobject.

This helps applications expose a simpler and more predictable JSON document shape when only a fixed number of child rows is needed. It also supports compatibility scenarios where an older API expects a single nested object, while the relational model or newer API can still use an array-based structure.

View Documentation

Changes for JSON Search Index and Data Guide

JSON search index and JSON data guide are enhanced in these ways. The first two represent changes in the default behavior.

1. When creating a JSON search index, by default a data guide is not created.
2. By default, `DBMS_JSON` procedures `create_view`, `get_view_sql`, and `add_virtual_columns` resolve name conflicts; that is, the default value of parameter `resolveNameConflicts` is `TRUE`, not `FALSE`. This means that if a resulting field name exists in the same data guide then it is suffixed with a new sequence number, to make it unique.
3. Function `json_dataguide` is enhanced to detect ISO 8601 date-time string values, using flag option `DBMS_JSON.detect_datetime`.

When this option is present, field values that are strings in the ISO 8601 date and time formats supported by Oracle are represented in a data guide with the value of field `type` not as `string` but as `timestamp`.

The default changes improve usability and performance for JSON data guides.

View Documentation

Comparing and Sorting JSON Data Types

JSON data type can now be used directly in a WHERE, ORDER BY, and GROUP BY clause.

The broader applicability of the JSON data type in SQL constructs simplifies your application development and improves the performance of your applications by avoiding the need for explicit casts.

View Documentation

DBMS_AQ Support for JSON Arrays

You can use a JSON data type array as the payload for Advanced Queuing (AQ) message-passing functions, which process an array of messages as a single operation. This applies to the AQ interfaces for C (Oracle Call Interface), PL/SQL, and Java (JDBC).

Advanced Queuing can directly use JSON data for its bulk message passing. With JSON being an increasingly popular format for data exchange, this functionality provides more flexible application development and improves developer productivity.

View Documentation

Deletable Shared Tables for JSON-Relational Duality Views

Developers can use the `@delete` annotation to mark tables in a JSON-relational duality view as deletable. For shared tables, Oracle AI Database deletes a row only when no child row continues to reference it and the required `RESTRICT` foreign-key constraint is enabled.

This helps applications clean up unreferenced shared data through the duality view instead of requiring separate SQL cleanup logic. It preserves referential safety by deleting shared rows only when they are no longer used by related data.

View Documentation

Duality View Inline Write Augmentation

Duality View inline `WRITE` augmentation can automatically create additional fields, not present in a document being inserted or updated.

Automatic addition of document fields lets you provide default field values or transform incoming values. The fields can be added always or only if missing from a document, and their values can be derived from other fields in the same document.

View Documentation

EMPTY STRING ON NULL for JSON Generation

When generating JSON data from relational data, a SQL `NULL` input value results in a JSON `null` value by default.

In Oracle SQL, a SQL `NULL` value cannot be distinguished from an empty string value (`''`). This means that an empty SQL string input is treated the same as SQL `NULL`. This behavior can sometimes confuse users.

When using a SQL/JSON generation function such as `json_object`, for `NULL` input values of a SQL character data type, such as `CLOB` and `VARCHAR2`, a user can specify that an empty JSON string (`""`) be created. The same is true for function json_scalar.

With this feature, generating a JSON empty string (`""`) from an empty SQL string is easy and efficient. Without this feature, a user needs to use a complex `CASE` statement to do the same.

View Documentation

Enhancement to JSON_TRANSFORM

`JSON_TRANSFORM` is extended to support right-hand-side path expressions, nested paths, and arithmetic operations. A SORT operator is supported which allows sorting the elements in an array.

`JSON_TRANSFORM` is the main SQL operator for modifying JSON data, both for on-disk updates and transient changes in the `SELECT` clause of a query. This enhancement increases update capabilities, such as arithmetic calculations and operations on nested arrays and raises developer productivity.

View Documentation

Enhancements to GraphQL Syntax for Duality View Creation

These syntax enhancements are provided for using GraphQL with JSON-relational duality views:

1. Comments
2. New directives: `@cast`, `@object`, and `@array`
3. Enhanced directives: `@generated`, `@link`, and `@where`
4. Support for fully qualified table/column names
5. Query-By-Example (QBE) predicates for filtering (analogous to SQL `WHERE`)
6. Simplified QBE using query arguments as predicates

These enhancements allow more flexible use of GraphQL when creating a duality view.

View Documentation

Incremental Schema Inference for the JSON-to-Duality Migrator

The JSON-to-Duality Migrator enables users to generate an initial relational schema and duality views from a set of JSON documents, then incrementally extend the schema by introducing additional JSON document sets. This supports progressive onboarding of application collections while reusing the existing relational schema.

Incremental schema generation allows teams to migrate and onboard JSON collections in manageable stages, reducing disruption and avoiding redundant schema design. By reusing existing relational structures and duality views, organizations can preserve prior development investment, accelerate application modernization, and adapt their data model as new collection requirements emerge.

View Documentation

JSON Collections

JSON collections are special tables or views that store (or represent) only JSON documents in a document-store-compatible format, such as the Oracle AI Database API for MongoDB. JSON collections are integrated into the database and fully operable with SQL, from creation to manipulation and query processing. For example, it is possible to do a simple INSERT AS SELECT into a JSON collection table.

JSON collection tables complement JSON Duality Views, the marquee JSON collection views that provide the benefits of relational storage and JSON document processing with a single database structure.

Native JSON collections simplify working with JSON data stored in collections within the Oracle AI Database ecosystem. For example, with collections, you easily analyze your JSON documents with SQL while concurrently using those operationally with document-centric APIs, such as the Oracle AI Database API for MongoDB.

View Documentation

JSON Data Guide Format FORMAT_SCHEMA

Format `FORMAT_SCHEMA` produces a data guide that you can use to validate JSON documents.

You can produce JSON data guide documents that you can use to validate JSON documents.

View Documentation

JSON Type Modifier Enhancements: Data Size and Array Specifications

You can specify the maximum size in bytes for a JSON-type column.

For type modifier ARRAY, you can specify:

- A (single) scalar type for all of the array elements
- Whether array elements can be JSON null
- The maximum number of array elements
- Whether to store array elements sorted in ascending order

Specifying a size limit for a JSON-type column and constraining array size, element type, and storage order can lead to more efficient handling of the stored JSON data.

View Documentation

JSON Type Modifiers

You can limit the kinds of values stored in a `JSON`-type column to objects, arrays, scalars, or a combination, by adding modifier keywords `OBJECT`, `ARRAY`, and `SCALAR` to the column specification when creating a table. For example, `DATA JSON (OBJECT)` ensures that only JSON objects can be stored.

Using modifier `OBJECT` alone ensures that the column data is compatible with *documents* in Oracle JSON collections and in typical document databases.

View Documentation

JSON Type Support for External Tables

Support for access and direct-loading of JSON-type columns is provided for external tables. JSON data type is supported as a column type in the external table definition. Newline-delimited and JSON-array file options are supported, which facilitates importing JSON data from an external table.

This feature makes it easier to load data into a JSON-type columns.

View Documentation

JSON to Duality Migrator: Hints Configuration Field

The hints configuration field for duality views can map primary-key columns to subobject fields, map SQL data types to fields, and disable sharing of data underlying a subobject.

You can obtain more control over the mapping between the documents supported by duality view and their underlying relational data:

1. Which document fields should correspond to primary-key columns
2. Which SQL data types should underlie particular fields
3. Whether to normalize particular objects in a document

This feature enhances the ability of developers to leverage JSON documents for data access using the highly efficient relational data storage model, without compromising simplicity or efficiency.

View Documentation

JSON to Duality Migrator: Multi-Collection Import API

`DBMS_JSON_DUALITY.IMPORT_ALL` is a PL/SQL procedure that is designed to import multiple document collections into a JSON-relational duality view.

While the existing `DBMS_JSON_DUALITY.IMPORT` procedure is limited to importing a single collection, the ability to import multiple collections in a single PL/SQL call simplifies the process and helps prevent constraint violation errors. The `IMPORT_ALL` procedure enables the efficient import of multiple document collections into a JSON-relational duality view.

View Documentation

JSON to Duality Migrator: Schema Inference using JSON Schema

This feature allows the JSON to Duality Migrator to use user provided JSON schemas for relational schema inference. The JSON schemas can be provided in isolation or in conjunction with data for more accurate schema inference.

This feature makes the JSON to Duality Migrator more accessible to customers who may not have a large amount of JSON data for the Migrator to analyze or who do not want to send confidential data to the Migrator. As long as they have a JSON schema for each collection, they can still use the Migrator.

View Documentation

JSON to Duality Migrator: Validation of Schema and Data

The PL/SQL functions `DBMS_JSON_DUALITY.VALIDATE_SCHEMA_REPORT` and `VALIDATE_IMPORT_REPORT` are provided to validate the relational schema and data that are created and imported by the JSON-To-Duality Migrator.

There is a growing need for functions that would validate the recommended relational schema. To address these needs, the validation APIs have been developed that:

- Help users verify accuracy of the recommended relational schema
- Confirms that no data is lost when they migrate their document collections to duality views.

View Documentation

JSON-to-Duality Converter

Given an existing set of JSON collections as input, the JSON-To-Duality Converter creates a set of JSON-relational duality views, based on normalized relational schemas, that support the same document collections. This creation needs no user supervision, but users can override schema recommendations.

This feature provides one part of the JSON-to-Duality Migrator, which is a set of PL/SQL procedures to move document-centric applications and their JSON documents from a document database to duality views in Oracle AI Database.

View Documentation

JSON-to-Duality Importer

This feature imports application data from a set of JSON collections into JSON-relational duality views that have been created using the JSON-to-Duality Converter.

This feature provides one part of the JSON-to-Duality Migrator, which is a set of PL/SQL procedures to move document-centric applications and their JSON documents from a document database to duality views in Oracle AI Database.

View Documentation

JSON/JSON_VALUE will Convert PL/SQL Aggregate Type to/from JSON

The PL/SQL JSON constructor is enhanced to accept an instance of a corresponding PL/SQL aggregate type, returning a JSON object or array type populated with the aggregate type data.

The PL/SQL JSON_VALUE operator is enhanced so that its returning clause can accept a type name that defines the type of the instance that the operator is to return.

JSON constructor support for aggregate data types streamlines data interchange between PL/SQL applications and languages that support JSON.

View Documentation

JSON_ARRAY Constructor by Query

A subquery can be used as an argument to SQL/JSON function JSON_ARRAY to define the array elements. This functionality is part of the SQL/JSON standard.

This feature increases your developer productivity and higher interoperability with other SQL/JSON standard-compliant solutions.

View Documentation

JSON_BEHAVIOR Initialization Parameter to Control SQL/JSON Runtime Behavior

You can use the initialization parameter `JSON_BEHAVIOR` to change the default behavior of SQL/JSON operators within a database session.

You can change the default return data type, default type-compatibility, and default error behavior during a session for applicable SQL/JSON operators.

This lets you enforce consistent session-level JSON processing behavior, reducing the need for explicit overriding of default behaviors within individual SQL statements.

View Documentation

JSON_EXPRESSION_CHECK Parameter

A new parameter `JSON_EXPRESSION_CHECK` allows to enable/disable a JSON query check. The values are `on` and `off`. The default is `off`. For now, this parameter is limited

to JSON-relational duality views. An error is raised if a JSON path expression on a duality view does not match to an underlying column, for example if the path expression has a typo. The error is raised during query compilations.

This simplifies working with JSON-relational duality views, as incorrect JSON path expressions do not need to be debugged at runtime but instead are flagged at query compilation time (by raising an error).

View Documentation

JSON_ID SQL Operator

SQL operator `JSON_ID` generates a unique document-identifier value, for unique access to JSON documents in a collection. The argument to `JSON_ID` determines whether the value is a 12-byte `OID` or a 16-byte `UUID`. In Oracle JSON collections, `JSON_ID` is used to create (automatically or explicitly) the values for document-identifier field `_id`.

`JSON_ID` simplifies the generation of ID values to uniquely identify JSON documents.

View Documentation

JSON_TRANSFORM Operators ADD_SET and REMOVE_SET

Oracle SQL function `JSON_TRANSFORM` operators `ADD_SET` and `REMOVE_SET` work with JSON arrays as if they are *sets*; that is, as if their elements are unordered and unique (no duplicates).

- Operator `ADD_SET` adds a value to an array only if the value is not already an element.
- Operator `REMOVE_SET` removes all occurrences of a given value from an array.

Application code can more concisely update arrays that it uses as sets.

View Documentation

LOBs Returned by SQL Functions for JSON can be Value-Based

Wherever a SQL function for JSON returns a LOB value, the returning clause can specify that the LOB be value-based. By default, a LOB reference is returned instead. For example:

```
JSON_SERIALIZE (data returning CLOB VALUE)
```

Value-based LOBs are are easier to use because they do not need to be freed explicitly. The database fully manages the lifecycle of value-based LOBs and frees them when appropriate.

View Documentation

New JSON Data Dictionary Views

New dictionary views `*_JSON_INDEXES` and `*_TABLE_VIRTUAL_COLUMNS` have been added.

These new views provide better insight into the database objects that have been created to work with JSON data.

View Documentation

Notification Directives

Notification directives provide an efficient way to journal (log) and query the changes made by database transactions to JSON-relational duality views and their tables.

Logged notifications are ordered by the commit times of their transactions. They are consistent as of the commit time of a transaction.

Uses of notifications include the following:

- Replicate modified duality-view documents or table rows to another database, including document databases and other non-Oracle databases
- Maintain middle-tier caches of duality-view documents
- Generate change-notification events, to be consumed by microservices. These can initiate out-of-line microservice actions. For example, the creation of a new purchase-order document can initiate actions in fulfillment and payment microservices.

View Documentation

ORDERED in JSON_SERIALIZE

The SQL function `JSON_SERIALIZE` has an optional keyword `ORDERED`, which reorders the key-value pairs alphabetically (ascending only). It can be combined with optional keywords `PRETTY` and `ASCII`.

Ordering the result of serialization makes it easier for both tools and humans to compare values.

View Documentation

Precheckable Constraints using JSON SCHEMA

To avoid sending invalid data to the database, an application can often precheck (validate) it. PL/SQL function `DBMS_JSON_SCHEMA.describe` provides JSON schemas that apps can use to perform validation equivalent to that performed by database column-level check constraints, and it records constraints that have no equivalent JSON schema.

Applications can also check which columns are precheckable with a JSON schema by consulting static dictionary views `ALL_CONSTRAINTS`, `DBA_CONSTRAINTS`, and `USER_CONSTRAINTS`.

When you create or alter a table you can use keyword `PRECHECK` to determine whether column check constraints can be prechecked outside the database. If no equivalent JSON schema exists for a given `PRECHECK` column check constraint then an error is raised.

Early detection of invalid data makes applications more resilient and reduces potential system downtime. All applications have access to the same information about whether data for a given column is precheckable, and if so what JSON schema validates it.

View Documentation

Predicates for JSON_VALUE and JSON_QUERY

JSON path expressions with predicates can be used in `JSON_VALUE` and `JSON_QUERY`. The functionality is part of the SQL/JSON standard.

Applying JSON path expressions more widely for querying JSON data boosts your developer's productivity and simplifies code development.

View Documentation

SCORE Ancillary Operator for JSON_TEXTCONTAINS()

This feature allows you to return a score for your `JSON_TEXTCONTAINS()` queries by using the `SCORE()` operator.

You can also order the results by the score.

JSON_TEXTCONTAINS function gains a new parameter for use with the SCORE() function allowing for an improved development experience.

View Documentation

SODA Enhancements

Various extensions are made to the SODA API:

- **Merge and patch:** New SODA operations mergeOne and mergeOneAndGet.
- **Embedded Keys:** You can now embed the key of a document in the document itself. This is used for MongoDB-compatible collections.
- **Dynamic Data Guide:** The operation to compute a data guide on the fly is extended to other SODA languages, besides PL/SQL and C.
- **Sampling operation:** The sampling operation is extended to other SODA languages, besides PL/SQL and C.
- **Flashback:** The operation to use flashback is extended to other SODA languages, besides PL/SQL and C.
- **Hints and monitoring:** Hints and SQL monitoring are extended to other SODA languages, besides PL/SQL and C.
- **Explain plan:** Obtaining a SQL execution plan is extended to other SODA languages, besides PL/SQL and C.
- **Data Guard and Golden Gate:** You can now replicate SODA collections using Oracle Data Guard and Oracle GoldenGate.
- **Index Discovery:** You can now fetch all indexes for a given SODA collection.
- **Multivalue index creation:** New SODA APIs for PL/SQL, C, and Java to create multivalue indexes.

These extensions increase the usability and capabilities of SODA in general, thus improving developer productivity.

View Documentation

Simple Dot-Notation Syntax for JSON Indexes

You can now create single-value and multi-value indexes on JSON data using simple dot notation. This feature optimizes query performance for both Oracle API for MongoDB and SQL, with optional configurations to maximize index usage for common queries.

JSON indexes significantly improve query performance by eliminating full-table scans and lowering the cost of joins. This enables faster data retrieval and greater efficiency for complex queries. The simple creation syntax streamlines development and boosts productivity.

View Documentation

Tools to Migrate JSON Text Storage to JSON Type Storages

The new PL/SQL procedure, `DBMS_JSON.json_type_convertible_check`, checks whether existing data stored as JSON text can be migrated to JSON data type. There are several alternative ways to migrate the data after this check succeeds.

Leveraging the binary JSON data type format provides the best performance for processing JSON data. Providing a simple and easy way to ensure existing data can be transformed successfully to binary JSON format helps you to adopt the preferred storage format for JSON data.

View Documentation

User-Defined Identifying Keys for JSON-Relational Duality Views

Developers can specify the JSON field or fields that uniquely identify rows in tables used by a JSON-relational duality view. This can be used when a table has no primary-key or unique-key constraint, or when the application needs to override the key inferred by the database.

This feature lets applications use JSON document identities that match their data model, even when the relational schema does not expose the right database constraint. It reduces the need to change existing tables before creating or evolving duality views, and helps keep document updates mapped to the intended relational rows.

View Documentation

Validation Directives for JSON Relational Duality Views

A validation directive can ensure that the documents supported by a duality view, and the relational data underlying them, conform to business rules and constraints.

A validation directive provides a central, declarative way to manage data sharing across applications, and maintain data integrity and consistency. It can validate a

single document or all documents that share underlying data with a given document, within the same duality view or across all duality views. It can perform validation at DML time or transaction-commit time.

View Documentation

WHERE Clauses in JSON-Relational Duality Views

When creating a JSON-relational duality view you can use simple `WHERE` clauses to limit the rows from which to generate JSON data from underlying tables. As one kind of use case, you can create multiple duality views, whose documents contain different data depending on the values in a discriminating column. For example, with the same underlying table you can define views for data from different countries, using a `WHERE` clause that selects only table rows whose country-code column has a given value (for example, `FR` for France). The JSON documents supported by a country view reflect this requirement, and the requirement is enforced for updates.

`WHERE` clauses in view definitions allow fine-grained control of the data that is to be included in a JSON document supported by a duality view.

View Documentation

SQL

Schema Annotations

Schema annotations enable you to store and retrieve metadata about database objects. These are name-value pairs or simply a name. These are free-form text fields applications can use to customize business logic or user interfaces.

Annotations help you use database objects in the same way across all applications. This simplifies development and improves data quality.

View Documentation

Direct Joins for UPDATE and DELETE Statements

Join the target table in `UPDATE` and `DELETE` statements to other tables using the `FROM` clause. These other tables can limit the rows changed or be the source of new values.

Direct joins make it easier to write SQL to change and delete data.

View Documentation

IF [NOT] EXISTS Syntax Support

DDL object creation, modification, and deletion now support the `IF EXISTS` and `IF NOT EXISTS` syntax modifiers. This enables you to control whether an error should be raised if a given object exists or does not exist.

The `IF [NOT] EXISTS` syntax can simplify error handling in scripts and by applications.

View Documentation

New Database Role for Application Developers

The `DB_DEVELOPER_ROLE` role provides an application developer with all the necessary privileges to design, implement, debug, and deploy applications on Oracle databases.

By using this role, administrators no longer have to guess which privileges may be necessary for application development.

View Documentation

Aggregation Filters

A new `FILTER` clause filters the results of Oracle SQL aggregation functions as specified in the SQL standard.

This feature focuses on common problem areas for analytic queries by letting conditions be applied directly within aggregate functions, making queries clearer, faster, and more reliable. The `FILTER` clause helps developers write more complex analytic queries in SQL.

View Documentation

Aggregation over INTERVAL Data Types

You can pass `INTERVAL` data types to the `SUM` and `AVG` aggregate and analytic functions.

This enhancement makes it easier for developers to calculate totals and averages over `INTERVAL` values.

View Documentation

Automatic PL/SQL to SQL Transpiler

PL/SQL functions within SQL statements are automatically converted (transpiled) into SQL expressions whenever possible.

Transpiling PL/SQL functions into SQL statements can speed up overall execution time.

View Documentation

BAR_CHART Function

`BAR_CHART()` is a function that lets users easily generate bar charts from numeric values.

It can be used for visually representing the distribution of values in queries.

View Documentation

BOOLEAN Data Type

Oracle AI Database now supports the ISO SQL standard-compliant `BOOLEAN` data type. This enables you to store `TRUE` and `FALSE` values in tables and use `BOOLEAN` expressions in SQL statements.

The `BOOLEAN` data type standardizes the storage of `Yes` and `No` values and makes it easier to migrate to Oracle AI Database.

View Documentation

Calendar Functions

This series of new calendar functions allows a user to extract year, quarter, month, week, and day information for calendar, fiscal, and retail calendars.

Calendar functions simplify time-based analytics by enabling consistent queries, grouping, and navigation across calendar, fiscal, and retail time dimensions.

View Documentation

Client Describe Call Support for Tag Options

Annotations enable you to store and retrieve metadata about database objects. These are either name-value pairs or only a name. These are free-form text fields that applications can use to customize business logic or user interfaces.

Annotations help you to use database objects in the same way, across all applications. This simplifies development and improves data quality.

View Documentation

DEFAULT ON NULL for UPDATE Statements

You can define columns as `DEFAULT ON NULL` for update operations, which was previously only possible for insert operations. Columns specified as `DEFAULT ON NULL` are automatically updated to the specific default value when an update operation tries to update a value to `NULL`.

This feature simplifies application development and removes your need for complex application code or database triggers to achieve the desired behavior. Development productivity is increased and code becomes less error-prone.

View Documentation

DESCRIBE Now Supports Column Annotations

The SQL*Plus `DESCRIBE` command can now display annotation information for columns that have associated annotations available.

Annotations help you to use database objects in the same way across all applications. This simplifies development and improves data quality.

View Documentation

Data Use Case Domain Metadata Support in OCCI

Provide access to the Data Use Case Domain metadata (domain name and domain schema) for the database columns described in OCCI (Oracle C++ Call Interface) applications.

Database adds Data Use Case Domains to columns and the column metadata need to expose the same in all the data access drivers.

View Documentation

Data Use Case Domains

A data use case domain is a dictionary object that belongs to a schema and encapsulates a set of optional properties and constraints for common values, such as credit card numbers or email addresses. After you define a use case domain, you can define table columns to be associated with that domain, thereby explicitly applying the domain's optional properties and constraints to those columns.

With use case domains, you can define how you intend to use data centrally. They make it easier to ensure you handle values consistently across applications and improve data quality.

View Documentation

Dimension-Wise Arithmetic Support in PL/SQL

Addition (+), subtraction (-), and multiplication (*) can now be applied to vectors in PL/SQL. The arithmetic operation is performed at each dimensional element of the vectors.

Arithmetic operations on vectors allow AI systems to manipulate and combine abstract concepts, enhancing their ability to understand and process language or data in more sophisticated ways. PL/SQL support of vector arithmetic provides developers a means to apply these operations within PL/SQL blocks and functions without calling out to SQL.

View Documentation

Dynamic Statistics for PL/SQL Functions

Dynamic statistics support has been enhanced for PL/SQL functions used in SQL `WHERE` clauses and `TABLE` functions. Long-running PL/SQL functions can increase SQL parse times if they are used with dynamic statistics, so global-level and fine-grained controls are provided to configure which functions should be included or excluded.

Cardinality can be difficult to estimate when optimizing SQL statements containing PL/SQL functions. This can lead to poor SQL execution plans and poor SQL performance. The ability to control and use dynamic statistics with PL/SQL functions enables the optimizer to find better execution plans, which leads to improved overall database performance.

View Documentation

Error Message Improvement

The Oracle Call Interface (OCI) `OCIError()` function has been enhanced to optionally include an Oracle URL with error messages. The URL page has additional information about the Oracle error.

This feature allows users to access information about the cause of the error and the actions that can be taken.

View Documentation

Extended CASE Controls

The `CASE` statement is extended in PL/SQL to be consistent with the updated definitions of `CASE` expressions and `CASE` statements in the SQL:2003 Standard [ISO03a, ISO03b].

Dangling predicates allow tests other than equality to be performed in simple `CASE` operations. Multiple choices in `WHEN` clauses allow `CASE` operations to be written with less duplicated code.

View Documentation

GROUP BY ALL

In SQL queries with complex SELECT lists that contain aggregation functions, the new GROUP BY ALL clause eliminates the need to put all non-aggregated columns into the GROUP BY clause. Instead, the new ALL keyword indicates that the results should be automatically grouped by all non-aggregated columns.

Not having to repeat the non-aggregated columns in the GROUP BY clause, makes writing SQL queries quicker and less error prone. Users can use the GROUP BY ALL functionality to either quickly prototype their SQL query or for quick ad-hoc queries.

View Documentation

GROUP BY Column Alias or Position

You can now use column alias or `SELECT` item position in `GROUP BY`, `GROUP BY CUBE`, `GROUP BY ROLLUP`, and `GROUP BY GROUPING SETS` clauses. Additionally, the `HAVING` clause supports column aliases.

These enhancements make it easier to write `GROUP BY` and `HAVING` clauses. It can make SQL queries much more readable and maintainable while providing better SQL code portability.

View Documentation

Generate and Test UUID Using SQL

UUID() operator now supports generation of types 5 and 7 UUIDs natively in Oracle database, including predefined namespaces for DNS, URL, OID and X500. IS_UUID() has been correspondingly modified to detect types 4, 5, and 7 UUIDs.

UUIDs are widely used for generating universally unique identifiers in database workloads. Oracle now supports inbuilt UUID functions that allow generation and insertion of UUID in tables for types 4, 5, and 7 UUID. Particularly, type 7 UUID is incredibly useful for databases because it combines the uniqueness of a traditional UUID with the sequential ordering of a standard database auto-incrementing integer.

View Documentation

Improved TNS Error Messages

This feature enhances common TNS error messages by providing more information, such as cause of the error and the corresponding action to troubleshoot it.

Having a better description of errors improves diagnosability.

View Documentation

JOIN TO ONE Join Method and Syntax

`JOIN TO ONE` is a modern join syntax that simplifies `FROM` clauses while preventing common join mistakes that can silently produce wrong results. By expressing joins as non-multiplying from a clearly identified Row-Widened Tables (RWTs), it ensures each result row maps one-to-one to a subset of RWTs rows and raises an execution-time

error if any join would return more than one matching row per RWTs row - making join cardinality explicit and safer by construction.

Because all tables are reached through `JOIN TO ONE`, it rigorously avoids “chasm traps,” where a parent row unintentionally fans out across multiple sibling child tables and produces inflated or incorrect results that often go unnoticed in production. When the schema declares the relevant foreign-key to primary/unique-key relationships, `JOIN TO ONE` can also be the most concise form, inferring the join condition from those keys and without having to write an `ON` clause.

The feature makes joins easier to write and reduces the risk of flawed join combinations, either a Cartesian product, many-to-many joins, or chasm traps, that can produce wrong query results. As a result, `JOIN TO ONE` leads to higher-quality application SQL with fewer join-related SQL bugs.

View Documentation

Materialized View Query Rewrite with Cursor Sharing in the Presence of Bind Variables

Query rewrite now supports SQL with filters that include user bind variables. By enabling bind peeking during containment checks, the optimizer can reuse existing cursor metadata and apply materialized view rewrites without hard parsing. This reduces parse overhead and improves performance in workloads that rely heavily on bind variables.

This feature extends query rewrite to workloads with bind-variables, allowing more queries to benefit from materialized views. It reduces hard parses and improves query performance and response times in applications that use bind parameters extensively.

View Documentation

Mixed SELECT * and other columns/expressions

`SELECT *` can now be combined with other expressions, such as other column names, constants, or expressions.

This enables more versatile projections in SELECT clauses.

View Documentation

Multilingual Engine Support for BOOLEAN Data Type

Oracle AI Database features a native `BOOLEAN` data type. The server-side JavaScript engine fully supports the data type on all interfaces.

When using JavaScript to write stored code in Oracle, this feature allows you to take full advantage of the capabilities offered by the new `BOOLEAN` data type.

View Documentation

Non-Positional INSERT Clause

Oracle AI Database adds a new `INSERT INTO SET` clause, which is an easier, self-documenting syntax for `INSERT INTO` statements. The `SET` clause for the `INSERT INTO` statement is the same as the already existing `SET` clause for `UPDATE` statements. It also adds a `BY NAME` clause when inserting a the results of a subquery. This matches source and target columns by their name rather than their position in the `INSERT` and `SELECT` lists.

The benefit of the `SET` clause is that it is immediately clear which value in the `INSERT INTO` statement belongs to which column, which is not obvious and cumbersome to find out for current `INSERT INTO` statements with hundreds of columns. Similarly matching the order of hundreds of columns in `INSERT` and `SELECT` lists when loading a subquery is awkward. Using the `BY NAME` clause allows these lists to be in different orders, simplifying the process of writing the statement.

View Documentation

Oracle C++ Call Interface (OCCI) Support for BOOLEAN Data Type

Oracle C++ Call Interface (OCCI) now supports querying and binding of the new `BOOLEAN` data type.

Using the `BOOLEAN` data type enables applications to represent state more clearly.

View Documentation

Oracle Client Driver Support for BOOLEAN Data Type

Oracle client drivers support fetching and binding the new `BOOLEAN` database column.

Applications can use the native database `BOOLEAN` column data type with a native driver `BOOLEAN` data type. This enhancement makes working with `BOOLEAN` data types easier for developers.

View Documentation

SELECT FOR UPDATE with Row Limit

Top-N queries (`FETCH FIRST`) queries can now use the `FOR UPDATE [SKIP LOCKED]` clause.

Applications can lock and fetch a limited number of rows at a time. This allows multiple workers to process the dataset in parallel batches without waiting on rows locked by others.

View Documentation

SELECT Without FROM Clause

You can now run `SELECT` expression-only queries without a `FROM` clause.

This new feature improves SQL code portability and ease of use for developers.

View Documentation

SQL*Plus Support for SQL BOOLEAN Data Type

SQL*Plus supports the new SQL `BOOLEAN` data type in SQL statements and the `DESCRIBE` command. Enhancements to the `COLUMN` and `VARIABLE` command syntax have also been made.

SQL*Plus scripts can take advantage of the new SQL `BOOLEAN` data type for easy development.

View Documentation

Simplified Macros

Macros currently allow developers to use PL/SQL to construct SQL strings that can be returned by macro functions. While this capability is powerful, it makes simple and common use cases unnecessarily verbose and complex since we need to use PL/SQL

to define macros. Now, there is a simplified syntax for writing these simple macros, so PL/SQL interfaces can be avoided. It makes macros very user friendly.

This feature allows you to create SQL expressions that you can reuse across SQL statements. These are parameterized expressions, which can be either scalar or table valued. Furthermore, by allowing table-valued parameters, it's also possible to use these macros to define parameterized views.

View Documentation

Streaming Windowing Table Functions

The table functions enable you to split rows into time-based windows: `TUMBLE`, `HOP`, `CUMULATE`, and `SESSIONS`.

Use these functions to analyze data over time in batch mode. They split rows into consecutive, sliding, cumulative, and session-based time windows. Their design is modeled on Flink SQL's windowing table-valued functions.

View Documentation

Support for DATEADD Function

The `DATEADD` function in Oracle AI Database provides a simple API for adding or subtracting time units from years down to nanoseconds to a datetime value.

`DATEADD` gives a clear and consistent way to do datetime calculations across all time units and datetime data types.

View Documentation

Support for DATEDIFF Function

`DATEDIFF` and its synonym `TIMESTAMPDIFF` introduce a powerful utility for date and time difference calculations in Oracle AI Database by offering a more comprehensive and convenient solution to leverage existing SQL built-in functions.

`DATEDIFF` is supported in all the other databases and makes it easier for customers to migrate to Oracle AI Database.

View Documentation

Support for DECODE and CASE in SQL/JSON Path Expressions

Path expressions support functions `DECODE` and `CASE`, which are similar to SQL function `DECODE` and `CASE` expressions, respectively.

These functions allow conditional changes to JSON data, for example using SQL function `JSON_TRANSFORM` or JSON-relational duality views.

View Documentation

Support for Nested WITH Clause and Correlation

Oracle AI Database now supports nesting `WITH` clauses within other `WITH` clauses, removing the earlier restriction and allowing use anywhere a `WITH` clause is valid. However, recursive nesting, PL/SQL functions, and table value constructors inside nested `WITH` clauses remain unsupported. Existing hints (inline/materialize) can still be used but may be ignored in nested scenarios without affecting non-nested query blocks.

Supporting nested `WITH` clauses aligns Oracle SQL with other databases like PostgreSQL, easing customer migration. It also improves compatibility with MongoDB's aggregation pipelines, which rely heavily on staged query transformations. Since pipeline stages, for example, `$lookup` joins, translate into nested `WITH` subqueries, this feature enables more accurate and efficient query conversion.

View Documentation

Support for the QUALIFY Clause

The `QUALIFY` clause filters the results of analytic functions in a `SELECT` statement. The relationship between the `QUALIFY` clause and an analytic function is analogous to the relationship between the `HAVING` and `GROUP BY` clauses.

Without the `QUALIFY` clause, the only way to filter on the output of an analytic function is by nesting the query and applying a filter in the parent query, which is more verbose, adds additional maintenance overhead and may obscure the query's intent to the reader. The `QUALIFY` clause overcomes all these caveats.

View Documentation

Table Value Constructor

The database's SQL engine now supports a `VALUES` clause for many types of statements. This new clause allows for materializing rows of data on the fly by specifying them using the new syntax without relying on existing tables. Oracle supports the `VALUES` clause for the `SELECT`, `INSERT`, and `MERGE` statements.

The introduction of the new `VALUES` clause allows developers to write less code for ad-hoc SQL commands, leading to better readability with less effort.

View Documentation

Time Bucketing

Time bucketing is a common operation when processing time series or event streaming data where a series of data points within an arbitrarily defined time window need to be mapped to a specific fixed time interval (bucket) for aggregated analysis.

With the new SQL operator `TIME_BUCKET`, Oracle provides native and performant support for time bucketing of time-based data for `DATETIMES`.

Providing a native SQL operator for common fixed-time interval bucketing for time series data significantly simplifies application development and data analysis of such information. In addition to simpler and less error prone code, the native operator increases the performance of time series analysis.

View Documentation

UPDATE RETURN Clause Enhancements

The `RETURNING INTO` clause for `INSERT`, `UPDATE`, `DELETE` and `MERGE` statements are enhanced to report old and new values affected by the respective statement. This allows developers to use the same logic for each of these DML types to obtain values pre- and post-statement execution. Old and new values are valid only for `UPDATE` statements. `INSERT` statements do not report old values and `DELETE` statements do not report new values. `MERGE` can return both old and new values.

The ability to obtain old and new values affected by `INSERT`, `UPDATE`, `DELETE` and `MERGE` statements, as part of the SQL command's execution, offers developers a uniform approach to reading these values and reduces the amount of work the database must perform.

View Documentation

Unicode 15.0 Support

The National Language Support (NLS) data files for `AL32UTF8` and `AL16UTF16` character sets are updated to match version 15.0 of the Unicode Standard character database.

This enhancement enables Oracle AI Database to conform to the latest version of the Unicode Standard.

View Documentation

Graph

Native Representation of Graphs in Oracle AI Database

Oracle AI Database now has native support for property graph data structures and graph queries.

Property graphs provide an intuitive way to find direct or indirect dependencies in data elements and extract insights from these relationships. The enterprise-grade manageability, security features, and performance features of Oracle AI Database are extended to property graphs. Developers can easily build graph applications using existing tools, languages, and development frameworks. They can use graphs in conjunction with transactional data, JSON, Spatial, and other data types.

View Documentation

Support for the ISO/IEC SQL Property Graph Queries (SQL/PGQ) Standard

The ISO SQL standard has been extended to include comprehensive support for property graph queries and creating property graphs in SQL. Oracle is among the first commercial software products to support this standard.

Developers can easily build graph applications with SQL using existing SQL development tools and frameworks. Support of the ISO SQL standard allows for greater code portability and reduces the risk of application lock-in.

View Documentation

DBMS_AI_GRAPHRAG PL/SQL Package

The `DMBS_AI_GRAPHRAG` PL/SQL package extends standard vector-based RAG with graph-based traversal to enhance RAG accuracy.

GraphRAG provides key structural and reasoning advantages over traditional vector-only RAG. It supports multi-hop reasoning and provides higher accuracy for complex queries, and the answers are easier to explain and audit.

View Documentation

Property Graph: In-Database Graph Algorithms

Developers can use the Oracle Graph Algorithms package (DBMS_OGA) to execute popular algorithms like PageRank on graphs in the database.

Developers can now execute graph algorithms using SQL and PL/SQL. They can build graph applications using SQL development tools and frameworks.

View Documentation

Property Graph: Native Representation of Graphs in Oracle AI Database

Oracle AI Database now has native support for property graph data structures and graph queries.

Property graphs provide an intuitive way to find direct or indirect dependencies in data elements and extract insights from these relationships. The enterprise-grade manageability, security features, and performance features of Oracle AI Database are extended to property graphs. Developers can easily build graph applications using existing tools, languages, and development frameworks. They can use graphs in conjunction with transactional data, JSON, Spatial, and other data types.

View Documentation

Property Graph: Support for the ISO/IEC SQL Property Graph Queries (SQL/PGQ) Standard

The ISO SQL standard has been extended to include comprehensive support for property graph queries and creating property graphs in SQL. Oracle is among the first commercial software products to support this standard.

Developers can easily build graph applications with SQL using existing SQL development tools and frameworks. Support of the ISO SQL standard allows for greater code portability and reduces the risk of application lock-in.

View Documentation

Property Graph: Use JSON Collections as a Graph Data Source

SQL/PGQ queries can be executed on graphs represented as a JSON column (SQL/PGQ is the ISO standard for property graphs).

Developers can store a graph as a schema-less object in the database. Vertices and edges in a graph can have varying number and types of properties.

View Documentation

Property Graph: Use Native Representation of Graphs in Oracle AI Database with Graph Tools

Developers can visualize graphs and graph query results that use the native property graph object in Oracle AI Database.

Developers can query, analyze, and visualize graphs created by SQL DDL statements by using built-in advanced tools in Oracle AI Database. Existing applications can use this native representation of graphs without changing tools and the user interface.

View Documentation

RDF Graph: Execute Graph Analytics Algorithms with RDF Graphs

Oracle Graph algorithms in Graph Server can be used with RDF graphs.

You can now benefit from popular graph analytics algorithms, such as PageRank and Community Detection, potentially enhancing strategic decision-making and enabling deeper insights.

View Documentation

Support to Create SQL Property Graphs from Database Views

Oracle AI Database supports creation of SQL property graphs using database views.

Users can create property graphs from views on simple and multiple tables, aggregates, window and polymorphic table functions, hybrid partitioned tables, database links, object tables, table functions, and more. Graph creation can also use views with pseudocolumns, virtual columns, synonyms for views, and nested views.

With the ability to create SQL property graphs from database views, developers gain more flexibility when creating a graph. More data sources and advanced table features can now be used in a graph. For example, when creating a graph, developers can easily access data through database links, external tables in a Data Lake, data from column expressions with PL/SQL functions, and more.

View Documentation

Microservices

Kafka APIs for TxEventQ

Transactional Event Queues (TxEventQ) now support the KafkaProducer and KafkaConsumer classes from Apache Kafka.

Oracle AI Database can now be used as a source or target for applications using the Kafka APIs.

View Documentation

ODP.NET: Advanced Queuing and Transactional Event Queues

ODP.NET Core and managed ODP.NET now support Advanced Queuing (AQ) and Transactional Event Queues (TxEventQ) application programming interfaces (APIs) that can be used in modern applications, such as microservices. TxEventQ's highly optimized and partitioned implementation leverages the functions of Oracle AI Database so that producers and consumers can exchange messages at high throughput, by storing messages persistently, and propagate messages between queues on different databases. TxEventQ are a high performance partitioned implementation with multiple event streams per queue, while AQ is a disk-based implementation for simpler workflow use cases.

ODP.NET developers can leverage the same APIs no matter if they use TxEventQ or AQ. The APIs provide access to a robust and feature-rich message queuing systems integrated with Oracle AI Database. It can be used with web, mobile, IoT, and other data-driven and event-driven applications to stream events or communicate with each other as part of a workflow.

View Documentation

Prometheus/Grafana for Oracle

Prometheus/Grafana for Oracle will provide database metrics for developers running in a Kubernetes/Docker (K8S) environment. Database metrics are stored in Prometheus, a time-series database and metrics tailored for developers are displayed using Grafana dashboards. A database metrics exporter aids the metrics exports from database views into Prometheus time series database.

Developers of modern applications like microservices use observability at the app tier and often overlook the data tier. Data-driven applications don't get a full picture of the execution and performance of the application. Traditional database metrics are seen through AWR reports and Enterprise Manager, which are more targeted to the DBAs and less to the developers. For developers and architects, Prometheus and Grafana have become the tools for configuring metrics dashboards, setting alerts and taking remedial action. Developers can now tie in the app-tier metrics, Kubernetes container metrics, and Oracle AI Database metrics on behalf of the application together in a single dashboard. In addition to metrics, logs and tracing is also enabled to truly get unified observability on a single pane of glass.

View Documentation

Python and REST Drivers for Transactional Event Queues (TxEventQ)

Oracle AI Database 26ai introduces support in new languages for Transactional Event Queues (TxEventQ). TxEventQ can now be used in Python and with REST APIs (REST APIs implemented to be like Kafka's Confluent REST APIs). TxEventQ already has support for PL/SQL, C/C++, and Java using JMS or JDBC.

This feature increases developer productivity by allowing REST APIs applications to take advantage of Transactional Event Queues (TxEventQ) to handle application and data events. With increasing popularity of Python for Machine Learning applications, TxEventQ in the Oracle AI Database can now be part of the Machine Learning application data and events infrastructure.

View Documentation

Saga APIs using Oracle Saga Framework

Oracle Saga APIs are implemented in the database and provide a framework to implement transactional semantics for microservices built with Oracle AI Database.

The orchestrator Saga framework provides a way to maintain atomic data consistency across microservices.

Sagas are concurrent and execute local transactions in each participant database making it more efficient than distributed ACID transactions, thereby simplifying application code and increasing developer productivity.

View Documentation

Transactional Event Queues (TxEventQ) Propagation

Queues are used widely to send and receive events and messages between applications, increasingly being built as microservices. Transactional Event Queues (TxEventQ) are queues built into Oracle AI Database. Queue propagation allows multiple databases to act as producers and consumers of events and messages. Producer applications can send events in queues in one database, set up queue propagation to a remote database, and Consumer applications can consume events in queues in the remote database.

Queue propagation is used to consolidate critical events and data from remote locations to a central location for consolidated processing. Propagation is used to operate Transactional Event Queues (TxEventQ) as a reliable and secure Event Mesh, with multiple queues across multiple databases participating to send events and messages across the enterprise reliably and to remote subscribers with permissions. TxEventQ supports exactly-once messaging which makes it simpler to build and test applications.

View Documentation

General

.NET Metrics

.NET Metrics are application numerical measurements collected at regular time intervals for the purposes of monitoring and alerting about application health. In an ODP.NET setting, metrics can monitor connection statistics, such as number of ODP.NET hard connections to the database, number of active connections, or number of free connections.

ODP.NET Core and managed ODP.NET support .NET Metrics. ODP.NET metrics can be published to and analyzed by the rich and expansive toolsets integrated with OpenTelemetry and .NET Metrics, such as Grafana and Prometheus.

View Documentation

DBMS_DEVELOPER Package

The `DBMS_DEVELOPER` package provides an efficient way for developers to retrieve metadata for database objects, such as tables, views, and indexes.

In previous releases, object metadata had to be retrieved using the `DBMS_METADATA` package, and then queries had to be performed on various data dictionary tables and views. This method returned metadata in XML format.

The `DBMS_DEVELOPER` package returns metadata in JSON format, which can be easily integrated with applications and services, such as developer tools. You can adjust the level of detail to control the amount of metadata returned, tailoring it to your application's needs. Additionally, metadata retrieval time is significantly improved, enhancing integration with applications and services.

View Documentation

Dynamic Performance Views for Table and Partition Access Tracking

Read access to tables and partitions is tracked with a new dynamic performance view `[G]V$TABLE_ACCESS_STATS`and exposed in a user-friendly manner as data dictionary views `[DBA | ALL | USER]_TABLE ACCESS_STATS`, providing a deeper understanding of the access frequency of tables and individual partitions.

Allowing the user to see how often tables and individual partitions are read enables you to understand the importance and frequency of your data for better assessment of your lifecycle management of your data.

View Documentation

Efficient Table DDL Change Notification

Applications can now be notified when DDLs occur on tables.

Applications that need or want to be aware of table metadata can be notified of DDL changes rather than having to continuously poll for them.

View Documentation

Enhanced Inter-Session Communication with DBMS_PIPE

DBMS_PIPE, an in-database messaging framework for inter-session communication, got enhanced to support a broader set of applications and use cases. DBMS_PIPE now can share messages across multiple database sessions with concurrent reads, provides more flexibility in managing messages overall, and supports persistence and inter-instance and inter-database communication through object store buffering.

Providing a more comprehensive in-database inter-session messaging and communication enables more applications to take advantage of DBMS_PIPE, improving the reliability and scalability of applications. It also increases developer productivity by eliminating the need for more complex application architectures.

View Documentation

GB18030-2022 Support

The implementation of the Oracle client character set ZHS32GB18030 is updated to support the latest GB18030-2022 standard.

This feature enables Oracle AI Database to conform to the latest edition of the GB18030 standard, which is required for all software products sold in China.

View Documentation

GraphQL Schema Introspection API

PL/SQL function `get_graphql_schema` lets you construct the GraphQL schema for a relational schema.

A GraphQL schema helps you understand how GraphQL types (database tables) are linked with each other, and how to create GraphQL queries for the database.

View Documentation

GraphQL Table Function for SQL

SQL table function `graphql` lets you query Oracle Database using the GraphQL open-source query language.

Application developers can benefit from GraphQL's what-you-see-is-what-you-get syntax, which is especially useful for those less familiar with SQL.

View Documentation

JDBC RSI Support for Data Load Mode

In RSI stream mode, connection and prepared statement instances are created for every batch. With the new data load mode, the instances are created once and saved in the thread's local context.

This feature brings faster data ingestion into the Oracle AI Database.

View Documentation

ODP.NET: Asynchronous Programming

ODP.NET supports the .NET Task Asynchronous Programming (TAP) model with the core and managed drivers.

With support for TAP and the async and await keywords, ODP.NET data access operations are more responsive and easier to develop for asynchronicity.

View Documentation

ODP.NET: OpenTelemetry

OpenTelemetry is a popular open-source observability framework for instrumenting, generating, collecting, and exporting telemetry data. It provides a common specification and protocol so that multiple services can furnish a unified version of traces, metrics, and logs.

Numerous managed ODP.NET and ODP.NET Core APIs have been instrumented to support OpenTelemetry tracing. Developers can customize the ODP.NET OpenTelemetry trace settings and use manual, dynamic, or automatic instrumentation when needed.

With OpenTelemetry support, monitoring, tracking, and analyzing how ODP.NET operations interact in cloud computing, microservices and distributed systems becomes easier using this industry standard.

View Documentation

Oracle Call Interface (OCI) Support for String Indexed PL/SQL Associative Arrays

PL/SQL string indexed associative arrays are now supported by Oracle Call Interface (OCI). Applications can natively pass these associative arrays between the database and the client application allowing for creating, binding, and manipulating of this collection type.

This feature allows for more straightforward and less error-prone code development.

View Documentation

RESETTABLE Clause

The new `RESETTABLE` keyword in PL/SQL lets you indicate that a package can discard its state during reinstantiation without issue.

The `RESETTABLE` keyword marks packages that do not need to retain information during reinstantiation. This allows you to avoid the error `ORA-4068: existing state of packages has been discarded` during patches when you know that the package state can be discarded without issue.

View Documentation

Result Cache Integrity Mode

Oracle Result Cache allows the caching of query results in memory to improve the performance of frequently executed queries. Queries are cached optimistically based on the setting of `result_cache_mode` or explicit hinting, which considers objects that are not explicitly declared as deterministic for query caching.

Controlling the result cache integrity mode enables customers to enforce the requirement of declaring objects as deterministic before being considered for result caching.

Providing the capability to enforce the requirement of explicitly deterministic objects for query caching improves code quality and rules out the chance of accidentally caching objects that should not be cached.

View Documentation

SQL*Plus ARGUMENT Command

A new `ARGUMENT` command lets users of batch scripts control how SQL*Plus treats script argument variables for which the users have not explicitly set values. With this command, users are now able to control when to prompt for input or use a default value for each unset script argument.

This feature gives SQL script processing more resiliency and flexibility, allowing script actions to be customized by users if they want to alter parameter values.

View Documentation

SQL*Plus CONFIG Command

This command reads the default tnsnames.ora file and generates a JSON file suitable for uploading to a Centralized Configuration Provider.

This command makes it easier to migrate away from tnsnames.ora files and allows connection strings to be stored centrally.

View Documentation

SQL*Plus OERR Command and Improved HELP Syntax

A new `OERR` command in SQL*Plus allows users to see Oracle error message Cause and Action text within SQL*Plus for a user-supplied error number. The existing `HELP` command has also been enhanced to show the same text.

This feature allows developers to immediately get more information about error messages.

View Documentation

SQL*Plus PING Command and Command Line Option

A new SQL*Plus `PING` command and equivalent command line option can be used to show the round-trip time from SQL*Plus to either the network listener or to the database.

The network listener check is equivalent to the traditional `tnsping` command line utility that administrators use to check basic network connectivity. The option to check the

database round-trip time is commonly used as a liveness check to ensure that the database itself is reachable.

This feature gives users of SQL*Plus power to verify basic connectivity, which is useful in many troubleshooting or post-install scenarios.

View Documentation

SQL*Plus SET ERRORDETAILS Command

A new `SET ERRORDETAILS` command lets users decide whether additional information should be displayed when Oracle errors are generated in failure scenarios. Additional information that can be displayed is the error help URL and the message Cause and Action text.

This feature improves the developer experience by allowing faster troubleshooting.

View Documentation

SQL*Plus SHOW CONNECTION Command

This command can be used to show details about the current connection, list Oracle Net Service names present in the `tnsnames.ora` file, and resolve a given net service name to a connection string.

Knowing more about connection strings enables users to connect to Oracle AI Database more easily, and aids troubleshooting connection issues.

View Documentation

Session Exit on Invalidation

Set `SESSION_EXIT_ON_PACKAGE_STATE_ERROR` to true to force a hard session exit when a session's state has been invalidated.

Exiting sessions after state invalidation avoids errors that can occur when applications mishandle invalid state.

View Documentation

Unicode IVS (Ideographic Variation Sequence) Support

The new UCA1210_JAPANESE_IVS collation allows the processing of Unicode Ideographic Variation Sequence (IVS) in Japanese text. The SQL functions `LENGTHC()`, `SUBSTRC()`, `INSTRC()`, and `LIKEC()` are also enhanced to count IVSs as single complete characters.

This feature enables application developers to build applications supporting Unicode IVS. It is an important requirement for markets, such as Japan, where processing of data including names such as person names, place names, and historic texts often need to support ideographic characters represented in Unicode IVS.

View Documentation

Vector Data Type Support in Precompiler

Oracle has introduced a new data type, `VECTOR`. Enhancements have been made to both Pro*C and Pro*COBOL to provide application-level support for this data type. Applications can interact with the `VECTOR` data type through the following mechanisms:

1. String representation: Applications can perform `INSERT` and `FETCH` operations on `VECTOR` data using a string-based representation. In this format, vector elements are expressed as a comma-separated string.
2. Vector descriptor representation: The `VECTOR` data type can also be accessed using a descriptor, which is an opaque structure designed for handling complex data types. Through this mechanism, applications can perform `INSERT` and `FETCH` operations on `VECTOR` data in array form.

This feature enables end users to store, retrieve, and manipulate vector data more efficiently, thereby supporting modern workloads, such as similarity search, machine learning, and advanced analytics.

View Documentation

Java

Java in the Database: JDK 11 Support Including Modules

In this release, the Oracle JVM infrastructure has been re-architected to support JDK 11 capabilities including the Java module system.

This feature fosters productivity through the design or reuse and execution of code and libraries based on Java 11, inside the database.

View Documentation

Java in the Database: Web Services Callout Enhancement

This feature furnishes an enhanced implementation of the Web Services Call-Out Utility. Java, PL/SQL, and SQL can now perform a more efficient Web Services Callout.

This feature fosters extensibility and productivity by allowing Oracle AI Database to invoke external Web Services.

View Documentation

Java in the Database: HTTP and TCP Access While Disabling Other OS Calls

Oracle JVM now offers a more flexible Lockdown Profile configuration for on-premises and cloud database services (for example, Autonomous AI Database). HTTP and TCP callouts can now be enabled separately from other OS calls to allow deployments that depend on HTTP and TCP access.

This feature couples extensibility (for example, making HTTP and TCP callouts) with enhanced security for Java code running in the database.

View Documentation

JavaScript

Multilingual Engine JavaScript Modules and Environments

Multilingual Engine (MLE) Modules and Environments allow JavaScript code to persist and be managed in the database. Call specifications provide a means to call JavaScript functions from an MLE module anywhere you can call PL/SQL functions.

The introduction of JavaScript Modules and Environments as schema objects in Oracle AI Database allows developers to follow established and well-known workflows used in client-side JavaScript development. Complex projects can be broken down into smaller, more manageable pieces worked on independently by team members.

View Documentation

Multilingual Engine Module Calls

Multilingual Engine (MLE) Module Calls allow developers to invoke JavaScript functions stored in modules from SQL and PL/SQL. Call Specifications written in PL/SQL link JavaScript to PL/SQL code units.

Thanks to Module Calls, developers can use JavaScript functions anywhere PL/SQL functions are called.

View Documentation

Multilingual Engine Post-Execution Debugging

Oracle Multilingual Engine (MLE) allows developers to debug their JavaScript code by conveniently and efficiently collecting runtime state while the program is being processed, a method referred to as post-execution debugging. After the code has finished running, the collected data can be used to analyze program behavior, discover, and fix bugs.

Post-execution debugging offers a convenient way to extract runtime state information from a JavaScript code unit at runtime without having to change the observed code.

View Documentation

Multilingual Engine JavaScript SODA API

Simple Oracle Document Access (SODA) is a set of NoSQL-style APIs that let you create and store collections of documents (in particular JSON) in Oracle AI Database, retrieve them, and query them, without needing to know SQL or how the documents are stored in the database. With the introduction of MLE, JavaScript support for SODA documents exists for client-side and server-side development.

Supporting the Simple Oracle Document Access (SODA) API in JavaScript gives developers a choice between using JSON in a relational or No-SQL way, simplifying the development process and improving the portability of code.

View Documentation

Multilingual Engine JavaScript Support for JSON Data Type

Support for JavaScript Object Notation (JSON) is an integral part of Oracle AI Database. Oracle supports JSON natively with relational database features, including transactions, indexing, declarative querying, and views. A rich set of SQL functions is available to manipulate JSON in a relational model. Oracle Multilingual Engine (MLE) fully supports JSON: both dynamic MLE as well as MLE Module Calls support interactions with the JSON data type.

JSON and JavaScript objects are closely related, forming a natural match in such a way that makes working with JSON very easy with JavaScript code.

View Documentation

Compile-Time Syntax Checking for JavaScript Procedures and Functions

Syntax checks are now run at compile time on JavaScript functions executed using inline call specifications, providing valuable analysis of your code before runtime.

When used in combination with the linting tool of your choice, compile-time syntax checks can help increase confidence in the readiness of your code by providing an extra step of code analysis. This compile-time support matches the checks already available when creating MLE modules.

View Documentation

EXECUTE Privilege on JAVASCRIPT Not Required Anymore

The `EXECUTE ON JAVASCRIPT` privilege is no longer required in order to run JavaScript code from your user account.

You can now create Multilingual Engine (MLE) modules and execute inline call specifications to publish JavaScript functions without the extra step of granting the `EXECUTE ON JAVASCRIPT` privilege. This provides a more streamlined introductory experience to using MLE.

View Documentation

Foreign Function Interface for JavaScript to Call PL/SQL Code Units

The Foreign Function Interface (FFI) allows JavaScript developers to use a more familiar syntax to call code units written in PL/SQL.

Rather than using PL/SQL blocks, it is possible to use native JavaScript constructs to interact with most code written in PL/SQL. Using the FFI provides JavaScript developers with a much improved experience by streamlining the process of integrating PL/SQL code into JavaScript functions.

View Documentation

JavaScript Web API for In-Database JavaScript

Additional common web APIs have been added to MLE/JavaScript to improve compatibility with existing third-party JavaScript modules.

By supporting additional web APIs, it should now be easier to use third-party modules from community repositories, such as NPM in database applications. This enhancement should also reduce the need to create custom polyfills.

View Documentation

MLE WebAssembly Modules and JavaScript WebAssembly API

Integrating WebAssembly (Wasm) into Oracle AI Database's Multilingual Engine (MLE) enables you to run portable, language-neutral modules compiled from languages such as Rust, C, or C++ securely within the database.

WebAssembly broadens the pool of existing libraries and developer skills that can be brought directly to Oracle AI Database, reducing the need to rewrite specialized algorithms and integrating them via external process interfaces.

View Documentation

Restricted Execution Contexts for In-Database JavaScript

The `PURE` option can be specified on Multilingual Engine (MLE) environments and JavaScript inline call specifications to create restricted JavaScript contexts in which interaction with database functionality is not possible.

Creating restricted JavaScript execution contexts using the `PURE` option gives you a convenient way to limit capabilities of JavaScript code running in the database. A JavaScript program executing in a restricted context is guaranteed not to modify database tables or use PL/SQL packages, regardless of database privileges currently in effect. Common data processing use cases with user-defined functions require only computations on function inputs. In these scenarios, restricted contexts provide a

safety net, prohibiting unwanted database modifications—for example, when using third-party or open-source JavaScript libraries.

In the context of AI Vector Search, JavaScript functions can be used as user-defined vector distance metrics when creating an HNSW index. Only JavaScript functions marked with the `PURE` option are allowed on HNSW indexes.

View Documentation

SQL Objects and Collections in MLE JavaScript

Objects and collection data types are now supported with the Multilingual Engine (MLE). These types can now be used in call specifications as `IN`, `OUT`, and `IN OUT` function arguments as well as `RETURN` types.

The support of user-defined types and collections extends the available use cases for MLE in your applications.

View Documentation

Sparse Vector Support in MLE JavaScript

You can specify a vector's storage format as either `SPARSE` or `DENSE`. Sparse vectors are vectors that typically have a large number of dimensions but with very few non-zero dimension values.

Apart from potential performance improvements, sparse vectors can use considerably less space than a corresponding dense vector with very few non-zero values. Users of In-Database JavaScript benefit from sparse vector support.

View Documentation

Application Connectivity

Reset Database Session State

The reset database session state feature clears the session state set by the application when the request ends. The RESET_STATE feature cleans up dirty sessions so that the applications cannot see the state of these sessions in later usages. This feature applies to all applications that connect to the database using request boundaries.

APEX is an example of an application in the Oracle shared cloud service, where multiple users share the same connection. Application states such as Local Application Context can be used to set application information such as routing numbers, account numbers, addresses, social security numbers, medical and health care details and other private information. Leaving session state behind after usage is an application security threat.

By clearing the session state at the end of request, this has the added advantage that Transparent Application Continuity can always enable at the next user request as the session state is capturable.

When an application borrows and returns a connection to a connection pool, if the sessions state is not cleared, the next usages of that connection can see the session state set in previous requests. Leaving session state in a session after returning that session to a pool, opens a vulnerability for elicit use of session state that was set.

View Documentation

Automatic Session State Restoration for DRCP and PRCP Implicit Pooling

For DRCP and PRCP implicit pooling, Oracle AI Database automatically allows automatic session state restoration for supported implicit pooled connections by default.

Automatic session state restoration simplifies application management of supported session state across implicit pooling boundaries, while allowing callbacks for state that needs application-specific handling.

View Documentation

Implicit Connection Pooling for Database Resident Connection Pooling (DRCP)

This feature enables the automatic assignment of DRCP servers to and from an application connection at runtime when the application starts and finishes database operations, even if the application does not explicitly close the connection.

This feature can provide better scalability and efficient usage of database resources for applications that do not use application connection pooling.

View Documentation

Implicit Connection Pooling for Oracle Connection Manager in Traffic Director Mode (CMAN-TDM)

Client applications that do not use an application connection pool can take advantage of CMAN-TDM Proxy Resident Connection Pooling (PRCP) without making any application changes.

The new feature enables the automatic assignment of PRCP servers to and from an application connection at runtime when the application starts and finishes database operations even if the application does not explicitly close the connection.

This feature can reduce the size of PRCP pools required. It provides better scalability and efficient usage of resources for applications that do not use Oracle Session Pooling or Universal Connection Pooling (UCP).

View Documentation

Improved Oracle Connection Manager in Traffic Director Mode (CMAN-TDM) Pool Configuration Settings for Autonomous AI Database

Oracle Connection Manager in Traffic Director Mode (CMAN-TDM) has new Proxy Resident Connection Pooling (PRCP) configuration settings for use with Autonomous AI Database. Per-PDB PRCP pools can be enabled, allowing you to consolidate connection pools for each PDB and share these sessions across multiple services that belong to the same PDB. The maximum PRCP pool size can be dynamically configured based on the new `cman.ora` parameter `TDM_PERPDB_PRCP_CONNFACTOR` and the Oracle Compute Unit (OCPU) count allocated to each PDB.

The per-PDB PRCP mode provides efficient usage of database resources by reducing the number of pools in a CMAN-TDM gateway. Pool sizing can also now be more autonomous, reducing the need for manual re-configuration.

View Documentation

JDBC Enhancements to Transparent Application Continuity

This feature allows, when the RDBMS server supports it, templates (for example, stable restorable attributes), which are cross-session (one template might be used by multiple sessions). This feature also brings the ability to avoid the combinatorial explosion of templates by quarantining session states that are different in most sessions and therefore cannot be shared.

This feature simplifies high availability by moving most application continuity configurations to the server-side. Java applications inherit transparently (that is, no code required) the latest server-side enhancements.

View Documentation

JDBC Extensions for Apps Configuration Providers

JDBC instrumentalization for securely pulling Java Apps configuration from central stores such as Azure Config Store or OCI Object store or any JSON file accessible from generic web servers.

This feature simplifies Java application configuration in multi-Cloud environments.

View Documentation

JDBC Support for Kerberos Authentication using JAAS Configuration

By default, the Oracle JDBC Thin driver uses the default Kerberos login module, bundled with Oracle JDK (`com.sun.security.auth.module.Krb5LoginModule`). This feature enables applications that want to override the default behavior to specify a JAAS configuration file through the connection properties.

This feature provides flexibility with Kerberos Authentication configuration.

View Documentation

JDBC Support for Kerberos Authentication using User and Password Properties

This feature enables the users to configure Kerberos Principal and Password through the User and Password properties. The JDBC Thin driver takes care of initializing the `KerberosLoginModule` on behalf of the applications.

This feature simplifies Kerberos Authentication configuration.

View Documentation

JDBC Support for OAuth 2.0 Including OCI IAM and Azure AD

The Oracle JDBC driver provides support for OAuth 2.0 authentication for Oracle Cloud Infrastructure (OCI) Identity and Access Management (IAM) Cloud Service or the Azure Active Directory.

This feature simplifies Java application authentication to the Oracle Autonomous AI Database using OAuth 2.0 in multi-Cloud environments (OCI, Azure) in lieu of traditional credentials mechanisms such as username/password or strong authentication mechanisms, such as Kerberos or Radius.

View Documentation

Java Support for True Cache

The `Connection.setReadOnly` and `Connection.isReadyOnly` methods have been enhanced to transparently support True Cache. Developers simply need to set the new connection and system property `oracle.jdbc.useADCDriverConnection` to `true`.

JDBC support for True Cache furnishes mission-critical availability of Oracle AI Database to Java applications. It eliminates single points of failure and prevents data loss and downtime.

View Documentation

Multilingual Engine Supports Database-Resident Connection Pools

JavaScript in Oracle AI Database (Multilingual Engine) can now be used in database connections using a Database Resident Connection Pool (DRCP). You can now invoke JavaScript procedures and functions in DRCP sessions, and use the `DBMS_MLE` API.

Support of DRCP extends access to in-database JavaScript and allows those workloads using in-database JavaScript to leverage the benefits of DRCP, such as improved scalability.

View Documentation

Multiple Named Pools for Database Resident Connection Pooling (DRCP)

Database Resident Connection Pooling (DRCP) now supports multiple named pools. New `DBMS_CONNECTION_POOL.ADD_POOL()` and `DBMS_CONNECTION_POOL.REMOVE_POOL()` procedures are added. Oracle Net connection string syntax is enhanced so a pool name can be specified for each connection. Existing procedures can be used to start, stop, or configure the named pools. Existing `GV$` and `V$` views show the appropriate pool name(s) in use.

Having multiple pools allows finer control on the DRCP pool usage. It helps prevent situations where some applications dominate the use of a single pool.

View Documentation

ODP.NET Transparent Application Failover

Oracle Transparent Application Failover (TAF) is a high availability feature that enables client apps to automatically reconnect to a secondary database instance if the connected primary instance fails or shuts down. ODP.NET Core and managed ODP.NET now support connection and basic session state TAF.

ODP.NET TAF enables apps to recover and continue operating when database downtime occurs. It requires no changes to .NET application code to use.

View Documentation

ODP.NET: Application Continuity

ODP.NET Core and managed drivers now support Application Continuity (AC) and Transparent Application Continuity (TAC). AC and TAC mask outages from end users and applications by recovering the in-flight database sessions following recoverable outages, including transactions. The recovery is transparent such that the end user merely experiences a slightly delayed execution, but no perceptible outage nor error.

AC and TAC improve the user experience for both unplanned outages and planned maintenance. They enhance the fault tolerance of systems and .NET applications that use Oracle AI Database. Developers can use AC and TAC with existing .NET apps without making any code changes.

View Documentation

ODP.NET: Pipelining

ODP.NET core and managed drivers support pipelining for its database communication. It allows subsequent database requests to be sent and queued transparently even while ODP.NET awaits a database response.

Pipelining improves overall app performance and allows database resources to be used more effectively. ODP.NET does not need to wait for the database to respond from previous requests before submitting subsequent requests.

View Documentation

Oracle Call Interface (OCI) Pipelined Operations

Oracle Call Interface (OCI) has been enhanced to support pipelining of operations. Pipelining enables applications to submit multiple database operations without waiting for a response from the server. The application has control over when the responses to the pipelined operations are harvested. This allows applications to continue work without being blocked while the database is generating results.

This feature is used to increase the overall throughput and responsiveness of applications and languages that use OCI. Pipelining reduces the server and client idle times in comparison with the traditional request-response model.

View Documentation

Oracle Call Interface (OCI) Session Pool Statistics

The Oracle Call Interface (OCI) session pool usage statistics can be viewed.

The statistics help in tuning pool sizes for better performance, and aid in understanding the life cycle of connections.

View Documentation

Oracle Connection Manager in Traffic Director Mode (CMAN-TDM) Support for Direct Path Applications

Oracle AI Database's Set Current Schema and Direct Path API features are now supported by Oracle Connection Manager in Traffic Director Mode (CMAN-TDM).

This feature enables more client applications to leverage CMAN-TDM connection multiplexing capabilities.

View Documentation

Oracle Connection Manager in Traffic Director Mode (CMAN-TDM) Usage Statistics

A new `V$TDM_STATS` view can be used to query usage statistics for CMAN-TDM per-PDB Proxy Resident Connection Pools (PRCP), such as the number of active client connections in the connection pool, the number of busy and free server connections, the maximum number of connections reached, and more.

Providing usage statistics helps to improve the monitoring and tuning of CMAN-TDM.

View Documentation

Oracle ODBC Driver Support for Application Continuity and Transparent Application Continuity

The Oracle ODBC driver does not provide a way to use the Oracle Call Interface (OCI) session pool, so applications cannot take advantage of reduced connection times or use Application Continuity, Transparent Application Failover, or Transparent Application Continuity. The Oracle ODBC driver provides an option for applications to use the Oracle session pool and access the functionality that comes with it. Please refer to Session and Connection Pooling for details about the session pool. This option will also make DRCP available to applications to reduce resource usage in Oracle Database.

With support for Application Continuity, Transparent Application Fail-over, Transparent Application Continuity applications will be able to work without interruptions due to planned or unplanned outages.

View Documentation

Resumable Cursors

Resumable cursors, those that span transactions, will be replayable with Transparent Application Continuity. Such cursors are common in batch processing (such as loading sets of records) and require special handling to reposition those cursors during replay with Transparent Application Continuity.

Broadened support with Transparent Application Continuity for applications that rely upon resumable cursors, those that span commits. These cursors are very common in repetitive batch operations, looping through sets of records for updates and inserts with a commit for each set of records. Now, TAC will be able to replay the transactions that were interrupted (and not yet committed).

View Documentation

Shut Down Connection Draining for Database Resident Connection Pooling (DRCP)

A new, optional `DRAINTIME` argument to `DBMS_CONNECTION_POOL.STOP_POOL()` allows active DRCP pools to be closed after a specified connection drain time, or be closed immediately without waiting for connections to be idle.

This feature gives DBAs better control over DRCP usage and configuration.

View Documentation

Support for AC/TAC in Precompilers Using Session Pool

Precompilers such as ProC and ProCOBOL are embedded SQL tools that provide constructs for connecting to the database and executing statements. They create dedicated connections and destroy them when the work that uses the embedded constructs is complete.

The connection pool feature optimizes connection setup and teardown by maintaining a pool of connections and transparently assigning them. This reduces the cost of creating connections and helps ensure a seamless user experience. The runtime executes SQL and releases the connection immediately afterward. This feature provides explicit request boundaries to support preliminary HA capabilities, such as handling known outages. When supported by the session pool, precompilers can take full advantage of DRCP and OCI high availability functionality, such as TAF, AC, and TAC.

Refer to Session Pooling and Connection Pooling in OCI to learn more about the benefits of session pooling.

By supporting AC and TAC in precompilers, applications can handle planned and unplanned outages seamlessly.

View Documentation

Transparent Sharding in OCI Session Pools

This feature adds Transparent Sharding support to OCI session pools, allowing applications to receive logical sessions while OCI establishes and routes physical database connections on demand to the appropriate shard or catalog database. It also enables Single Shard execution, where a session binds to a selected shard after its first statement and remains associated with it until released. For transactions, a new

session attribute allows statements to route directly to the target shard—rather than the catalog database—so single-shard transactions stay bound to that shard until commit or rollback.

Transparent Sharding simplifies application development by eliminating the need for applications to explicitly manage sharding keys or implement shard-routing logic. For businesses that operate large-scale, highly distributed transaction processing systems, this feature significantly reduces application complexity and accelerates adoption of Oracle Sharded Databases with minimal code changes. By leveraging transparent session routing through the OCI Session Pool, applications can seamlessly scale across additional shards while maintaining high availability, improving operational efficiency, and reducing the development and maintenance costs associated with shard-aware application logic.

View Documentation

UCP Support for XA Transactions with Sharded Databases

This feature allows sharded database connections to participate in eXtended Architecture (XA) transactions managed by WebLogic Server Transaction Manager.

This feature allows reliable XA transactions coupled with the scalability of sharded databases.

View Documentation

Database Drivers API Enhancements

Deep Security Support in Java

New APIs and SPI are added to support Deep Security that enforces fine-grained access control of the data. Oracle JDBC Driver defines a service provider interface (SPI) to pull security context and APIs to push the security context.

The following are the new APIs and SPI for executing database operations within the security context of an application user:

- oracle.jdbc.EndUserSecurityContext
- oracle.jdbc.EndUserSecurityContext.Builder
- oracle.jdbc.spi.EndUserSecurityContextProvider

DeepSecurity addresses the security needs of modern applications, including enterprise software, analytics, and agentic AI systems. It enforces fine-grained access control at the row, column, and cell levels, limiting users strictly to the data they are authorized to access. By securing data at its source, DeepSec provides a unified security architecture across application and database tiers, protecting all access paths to sensitive data.

View Documentation

Easy Connect Plus Support for LDAPS/LDAP

Oracle supports LDAP based name look-up for retrieving Database connection strings from the directory servers. The directory used can be OID, OUD, or AD.

Now, LDAP-based name lookup is possible without having `ldap.ora` and `sqlnet.ora`. The values that are specified as part of `ldap.ora` and `sqlnet.ora` for ldap name lookup, are passed in the URL string. If `ldap.ora` or `sqlnet.ora` is present and the ldap URL is passed, then the preference is given to the URL string.

For example: `ldap[s]://host[:port]/name[,context]?[parameter=value{&parameter=value}]`

Easy Connect Plus extends its support beyond TCP and TCPS to make it easy to use LDAP and LDAPS protocol and parameters.

View Documentation

Enhanced UCP Connection Borrow

Connection creation using the user thread, in the context of a borrow request, can take longer than the specified connectionWaitTimeout (CWT). If a connection has been released by another thread in the meantime, the connection creation request keeps waiting for the operation to complete. It is therefore more effective to borrow the released connection rather than waiting for the one being created.

This feature brings performance enhancement to Java applications during connection borrow.

View Documentation

JDBC Connection Property sendBooleanAsNativeBoolean

A new Connection property oracle.jdbc.sendBooleanAsNativeBoolean is added to restore the old behavior of the Boolean data type, which is used to take integer (0 or 1) for a Boolean data type.
When set to false (the default is true), this property will restore the old behavior of sending integer values (0 or 1) for the boolean data type.

This feature brings compatibility to Java applications that rely on the old behavior of the boolean data type.

The feature furnishes backward compatibility with the earlier JDBC driver behavior. This feature simplifies upgrading to the latest JDBC driver without breaking the behavior of existing Java applications.

View Documentation

JDBC Support for BOOLEAN Data Type

This feature exposes the Oracle RDBMS `BOOLEAN` data type to Java through a new `BOOLEAN` data type in `oracle.jdbc.OracleType Enum`, and `DatabaseMetadata`. This feature also performs the implicit conversion of character and number data types to `BOOLEAN` data types.

Java applications can take advantage of the new JDBC support for the standard JDBC `BOOLEAN` data type. The benefits include: increased portability and the ease of development fostered by the implicit conversion of character and number to `BOOLEAN`.

View Documentation

JDBC Support for Database Annotation

Annotation is a mechanism to store application metadata centrally in the database. Annotations can be specified at creation time (CREATE) or at modification time (ALTER). An individual annotation has a name and an optional value. Both the name and the value are freeform text fields. A schema object can have multiple annotations. JDBC furnishes the `getAnnotations()` method with two signatures (as illustrated below). It returns the annotation associated with the specified table or view. It returns `null` if there is no annotation for the given object.

```
getAnnotations?(java.lang.String objectName, java.lang.String domainName,
java.lang.String domainOwner) throws java.sql.SQLException
```

```
getAnnotations?(java.lang.String objectName, java.lang.String columnName,
java.lang.String domainName, java.lang.String domainOwner) throws
java.sql.SQLException
```

This feature enables sharing metadata across applications and microservices thereby increasing metadata management and productivity.

View Documentation

JDBC Support for Pipelined Database Operations

In the previous releases, the JDBC driver would not allow another database call to start until the current call had been completed however, with asynchronous and reactive programming, Java applications could perform non-database operations, in the meantime. In this release, the database server and the Oracle JDBC-Thin both support pipelining database operations. Java applications can now asynchronously submit several SQL requests to the server without waiting for the return of the preceding calls.

The combination of Java reactive and asynchronous programming (JDBC Reactive Extension, Reactive Streams (R2DB and Virtual Threads) with database support for pipelining fosters high throughput.

View Documentation

JDBC Support for Self-Driven Diagnosability

This feature eliminates the need to switch from the non-logging JAR files (for example, `ojdbcXX.jar`) to the debug JAR files (for example, `ojdbcXX_g.jar`) for logging purposes. In addition, it enables logging in the following three ways: logging per connection, logging at the tenant level, or logging globally.

This feature furnishes increased productivity and ease of use for Java applications. It greatly simplifies the debugging of Java applications by removing the need to switch from the production jars to the debug jars.

View Documentation

OCI API Functions to Validate and Protect Inputs

The OCI-C APIs provide robust validation and formatting of SQL input parameters in Oracle AI Database applications. They mirror the established `DBMS_ASSERT` PL/SQL utilities, but are specifically tailored for use in C-based and OCI client environments.

They can perform core tasks, such as safely enclosing string literals and SQL object names, as well as ensuring conformance to qualified and simple SQL name standards.

Using these OCI-C APIs enhances the overall security posture of applications built on Oracle AI Database by proactively defending against SQL injection attacks—one of the most critical security threats cited by CISA and OWASP. Equipping developers with modern, standardized tools helps reduce critical vulnerabilities and minimizes incident response and remediation costs.

View Documentation

ODBC Support for BOOLEAN Data Type

ODBC now supports the new `BOOLEAN` data type.

Using the `BOOLEAN` data type enables applications to represent the state more clearly.

View Documentation

Oracle Call Interface (OCI) Support for BOOLEAN Data Type

Oracle Call Interface (OCI) now supports querying and binding of the new `BOOLEAN` data type.

Using the `BOOLEAN` data type enables applications to represent state more clearly.

View Documentation

Precompiler Support for BOOLEAN Data Type

The Pro*C and Pro*COBOL precompilers now support querying and binding of the new `BOOLEAN` data type.

Using the new data type makes it easier to represent boolean state in applications instead of using a character column to indicate Y or N.

View Documentation

Prefetch Implicit Results with Top-Level Procedure Execution

Prefetching is enabled for implicit result sets returned when a PL/SQL procedure is executed. Currently, the prefetch applies only at the time of fetching the rows explicitly from each of the implicit result sets.

This feature provides two ways of prefetching. The current `OCI_ATTR_PREFETCH_ROWS` ensures the total number of rows prefetched from all the implicit results put together does not exceed this setting. A new attribute, `OCI_ATTR_PREFETCH_IMPL_CURSOR`, ensures the prefetch rows for individual implicit cursors does not exceed this setting.

Applications migrating from DB2 and SQL Server may incur additional database round-trips when procedures return multiple implicit result sets, because the Oracle ODBC driver does not prefetch these results. This enhancement enables prefetching at procedure execution, helping reduce round-trips and improve performance.

View Documentation

Promotion of Sessionless Transaction to XA Transaction

This feature enables users and external transaction managers to promote an existing sessionless transaction in the database to an XA transaction. All changes made as part of the sessionless transaction then come under the purview of the XA transaction and can be committed or rolled back using the XA transaction's two-phase commit semantics. Note: This feature does not change any user interfaces or the two-phase commit semantics of XA transactions.

This feature allows customers to start transactions efficiently without tying them to a database session and seamlessly promote them to XA transactions when work expands across multiple systems. This provides stronger data consistency, better scalability, simpler application design, and reliable failure recovery for distributed business operations.

View Documentation

True Cache Support in OCI Session Pools

OCI support for True Cache provides an option in OCI session pools to request a True Cache connection when the database hosts a True Cache service. It enables applications to manage connections to both a primary database instance and a True Cache instance using a single session pool.

This feature offers consolidating the connections to the True Cache instances and to the primary instances into a single session pool, enabling efficient sharing as per the application's read-only operations and read-write operations. It eliminates the need for the application to create two separate pools. With this feature, acquiring a connection from the pool includes a mode that specifies if it requires a connection to

the True Cache instance mandatorily or optionally (if available) or if it requires a connection to the primary instance only.

View Documentation

UCP Asynchronous Extension

Universal connection pool (UCP) is extended with asynchronous (reactive) database calls.

This extension furnishes high scalability and throughput to Java applications.

View Documentation

UCP Support for Self-Driven Diagnosability

The new universal connection pool (UCP) diagnosability feature provides the following capabilities:

- When logging is enabled (it is disabled by default), log records are written into an in-memory ring buffer.
- Tracing is enabled by default. A tracing event dumps the ring buffer into either a data-source-specific buffer or a common buffer.

This feature fosters productivity (for example, real-time debugging) and ease of use for Java applications using the UCP.

View Documentation

Redis

Oracle AI Database Read API for Redis

Oracle AI Database provides Redis server-compatible read functionality that lets existing and new Redis applications retrieve database-backed objects by using familiar Redis client libraries, commands, and SQL keys. Instead of storing separate Redis values, Oracle transforms underlying relational data into Redis-compatible objects when requested, with Oracle remaining the system of record.

Oracle AI Database simplifies the application architecture by removing the need for a separate Redis tier for supported database-backed read patterns. Developers can preserve the Redis read experience while avoiding custom cache-loading code, TTL

tuning, cache invalidation, and two-write coordination. Redis-compatible access can be configured with Oracle AI Database, Oracle True Cache, or Oracle Active Data Guard, supporting requirements for current data, read scale, and availability.

View Documentation

6 Data Analytics

This section describes the new data analytics features.

General

Hybrid Partitioned Tables with Interval and Auto-List Partitioning

You can create Hybrid Partitioned Tables using single-level partitioning with interval and automatic list partitioning. This is in addition to existing support for single-level partitioning and range and list partitioning.

These extensions to Hybrid Partitioned Tables in Oracle AI Database provide a user-friendly partitioning strategy.

View Documentation

Data Quality Operators in Oracle AI Database

This release introduces the following two new string matching operators based on approximate or "fuzzy" string matching.

- `PHONIC_ENCODE` converts words or phrases into language-specific codes based on pronunciation.
- `FUZZY_MATCH`, which is language-neutral, gauges the textual similarity between two strings.

The new phonic encoding and fuzzy matching methods enable more sophisticated matching algorithms to be run directly on data in the database rather than only in external applications, providing improved matching performance and efficiency, for example in data de-duplication, linking or enhancement.

View Documentation

Automatic Data Clustering

Oracle AI Database automatically and transparently clusters storage-based data in response to the type of queries used by the application workload. This allows the workload to make more efficient use of data access optimizations, such as storage indexes, zone maps, and join zone maps.

This feature significantly improves performance for data warehousing workloads based on zone maps or storage indexes. Once data is clustered, the performance of data-scanning queries improves because larger contiguous areas (or zones) of storage are pruned or skipped when they do not contain the data being matched by a particular query.

View Documentation

Extended Support and Faster Performance for JSON Materialized Views

Materialized views of JSON tables have been enhanced with the ability to fast refresh more types of Materialized Views of JSON tables as well as Query Rewrite support for these Materialized Views.

The performance for JSON table Materialized Views is significantly improved through better fast refresh capabilities and better query rewrite capabilities for more workloads. You can use JSON table Materialized Views more broadly in your applications, with better performance and less resource utilization.

View Documentation

Oracle SQL Access to Kafka

Oracle SQL Access to Kafka (DBMS_KAFKA) provides efficient, reliable, and scalable access to data streams from Apache Kafka and OCI Streaming Service. Streaming data can be queried via SQL or loaded into Oracle AI Database tables.

Oracle AI Database provides efficient, reliable, and scalable integration with Apache Kafka using the `DBMS_KAFKA` APIs. This API enables Oracle AI Database to consume data from external data streams without the need for costly, complex direct application connections using proprietary interfaces. Oracle SQL Access to Kafka enables you to use Oracle AI Database's rich analytic capabilities across all your data.

View Documentation

SQL

Text Indexes with Automatic Maintenance

You can specify a new automatic maintenance mode for text indexes using the `MAINTENANCE AUTO` index parameter. This method automates the `CTX_DDL.SYNC_INDEX` operation. This is now the default synchronization method for new indexes.

With this method, newly created text indexes do not require you to specify a synchronization interval or manually run a `SYNC_INDEX` operation. A background process automatically performs these tasks without user intervention. This helps in synchronizing a large number of indexes in an optimal manner, and also eliminates the manual or time-based `SYNC` operations. By using a background job rather than the database scheduler, it avoids scheduling conflicts and the risk of running out of available jobs. Overall it makes for simpler, more resilient applications and better utilization of hardware resources.

View Documentation

Transportable Binary XML

Transportable binary XML (TBX) is a new self-contained XMLType storage method. TBX supports sharding, XML search index, and Exadata pushdown operations, providing better performance and scalability than other XML storage options.

You can migrate existing XMLType storage of a different format to TBX format in any of these ways:

- Insert-as select or create-as-select
- Online Redefinition
- Data Pump

Transportable binary XML (TBX) provides better performance and scalability. With the support of more database architectures, such as sharding or Exadata, and its capability to easily migrate and exchange XML data among different servers, containers, and PDBs, TBX allows your applications to take full advantage of both this new XML storage format on more platforms and architectures.

View Documentation

Concurrent Materialized View Refresh for on-commit

Materialized view refresh provides concurrent refresh, where multiple sessions can refresh the same on-commit materialized views simultaneously without the need for serialization.

Concurrent refresh broadens the applicability of materialized views for your applications and helps make application development simpler. It provides faster refresh and more up-to-date materialized views.

View Documentation

Enhanced Automatic Indexing

Indexes incur a maintenance overhead during DML operations. This can work against their improvements to data access performance. The enhancements to Automatic Indexing take a broader view than in previous releases and account for index maintenance costs when deciding which indexes will benefit the workload as a whole. Columns filtered using range predicates are considered for indexes and function-based indexes are now supported. This further increases the scope of Automatic Indexing effectiveness.

Automatic Indexing better assesses the impact of DML operations in your database when choosing automatic indexing. Your performance benefits by determining the overall advantage of an index to your workload.

View Documentation

Enhanced Automatic Materialized Views

Automatic materialized views have been enhanced to include automatic partitioning. In addition, there is a more accurate internal cost model for automatic materialized view selection, which considers both access benefits and maintenance (refresh) costs, as well as the frequency of execution.

Rewrite capabilities have been broadened, including outer join queries with filter predicates.

Enhancing Automatic Materialized Views with more accurate cost-benefit analysis and broader usability optimizes the management of your materialized view eco system and improves the overall performance of your system.

View Documentation

Enhanced Automatic SQL Plan Management

Automatic SQL plan management has been enhanced to detect and repair SQL performance regressions more quickly. SQL plan changes are detected at parse-time and, after initial execution, SQL performance is compared with the performance of previous SQL execution plans. If a performance degradation is detected, the plan is repaired accordingly.

With automatic SQL Plan Management, your application service levels improves, and impacts caused by SQL performance (plan) regressions are minimized and addressed transparently and proactively.

View Documentation

Enhanced LOB Support for Distributed and Sharded Environments

Distributed LOBs are LOBs that are fetched from one server to another and may optionally be returned to the client. Shared LOBS are an extension of distributed LOBs where LOBs are transported between shards or between a shard and the shard coordinator. In previous versions, support for sharded and distributed LOBs were limited to persistent LOBs, and temporary LOBs only where they originate from JSON operations. Now all temporary LOBs (including Value LOBs) and new increased-length inline LOBs are usable as distributed and sharded LOBs.

You can now work with inline LOBs, value LOBs, and all temporary LOBs in distributed and sharded environments.

You experience improved performance, scalability, and garbage collection when you work with temporary LOBs, thus improving your developer productivity and application resilience.

View Documentation

Enhanced Parallel Processing Resources Management

Parallel processes are released pro-actively before individual statements using parallelism are finished. For example, an uncommitted parallel DML operation or a partially fetched parallel `SELECT` statement with 2 Parallel Server Sets (Producer-Consumer) will release one of the Parallel Server Sets as soon as it has finished working, freeing half of the parallel process for use of other statements.

Releasing parallel processes as early as possible and making them usable for other statements optimizes the utilization of your available resources, improving the overall performance of your systems and applications.

View Documentation

Increased Maximum Size of Inline LOBs of 8000 Bytes

LOB values are stored either in the table row (inline) or outside of the table row (out-of-line). The maximum size of the inline LOB is increased to 8000 bytes, allowing larger LOB values being stored inside a row. Earlier, the maximum size was 4000.

This provides better input-output performance while processing LOB columns. You can experience the improved performance while running operations, such as full table scans, range scans, and DML.

View Documentation

Materialized View Support for ANSI Joins

Materialized Views in Oracle AI Database support full rewrite capabilities for SQL statements using ANSI join syntax and for Materialized View definitions using ANSI join syntax.

Full support of ANSI joins with materialized view rewrite provides a significant performance improvement. Many queries, particularly ones generated by SQL Tools and Reports, often use ANSI join syntax. This enhancement allows such tools to benefit from materialized views for query rewrite regardless of the syntax used by joins.

View Documentation

Read-Only Value LOBs

Value LOBs, a read-only subset of Temporary LOBs, are valid for a SQL fetch duration and optimize the reading of LOB values in the context of a SQL query. Many applications use LOBs to store medium-sized objects, about a few megabytes in size, and you want to read the LOB value in the context of a SQL query.

Value LOBs provide faster read performance and get automatically freed when the next fetch for a cursor is performed, preventing the accumulation of temporary LOBs and simplifying the LOB management within your application.

Value LOBs provide faster read performance than classical reference LOBs for your workload and don't need specific LOB management in your application. Using Value LOBs improves your application performance and makes implementing applications with LOBs simpler and more manageable.

View Documentation

Semi-Join Materialized Views

Semi-Join Materialized View Rewrite is a unique form of query rewrite. A single, large unified dimension table in the query is replaced with one or more join-specific materialized views. In a unified dimension data model, where multiple dimension tables are merged into a single large dimension table, semi-join Materialized Views materialize one or more of the joins of such a single, large unified dimension table with the fact table.

This new type of Materialized View significantly improves the runtime and resource consumption for complex analytical operations. Semi-join Materialized Views are especially beneficial when the number of applicable dimension keys derived from the large unified dimension table (through semi-join) is small.

View Documentation

Ubiquitous Search With DBMS_SEARCH Packages

The new `DBMS_SEARCH PL/SQL` package allows the indexing of multiple schema objects in a single index. You can add a set of tables, external tables, or views as data sources into this index. All the columns in the specified sources are indexed and available for a full-text search.

With a simplified set of `DBMS_SEARCH` APIs, you can create indexes across multiple objects, add or remove data sources, and perform a full-text search within a single data source or across multiple sources using the same index.

This simplifies indexing tasks that were previously performed using the `USER_DATASTORE` procedures, thus enhancing developer productivity.

View Documentation

In-Memory

Automatic In-Memory Enhancements for Improving Column Store Performance

Automatic In-Memory (AIM) has been enhanced to automatically enable creation and removal of Database In-Memory performance features based on an enhanced workload analysis algorithm. These features include Join Groups, caching of hashed dictionary values for join key columns and In-Memory Optimized Arithmetic.

Automatic In-Memory (AIM) has been enhanced to identify and enable or disable Database In-Memory features that can improve performance. It enables features either selectively or globally, depending on which adds the most benefit. This improves application performance and also conserves space in the In-Memory column store without requiring manual intervention.

View Documentation

Automatic In-Memory Sizing for Autonomous AI Databases

The In-Memory column store will now automatically grow and shrink dynamically based on workload. This allows the In-Memory column store to be available on Autonomous AI Database. Exadata scan performance is further improved for objects that are partially populated.

With Automatic In-Memory sizing, there is no longer a need to manually resize the In-Memory column store to accommodate different database workloads. This reduces the administrative effort of enabling Database In-Memory. Automatic In-Memory sizing also allows the In-Memory column store to be enabled on Autonomous AI Database (ADB), enabling applications running on ADB to also take advantage of faster analytic query performance.

View Documentation

Heat Map Retention Time

Heat Map Retention Time allows users to set the number of days that Heat Map data, which includes data access and data modification tracking information, is retained by the database before it is deleted. Using the new PL/SQL package `DBMS_HEAT_MAP_ADMIN`, users can specify Heat Map Retention Time, ranging from 1 day to 4294967294 days.

Retention times allow customers to optimize their storage by purging Heat Map data that is no longer required.

View Documentation

In-Memory Optimized Dates

To enhance the performance of DATE-based queries DATE components (i.e. DAY, MONTH, YEAR) can be extracted and populated in the IM column store leveraging the In-Memory Expressions framework.

This enhancement enables faster query processing on DATE columns which can significantly improve the performance of date based analytic queries.

View Documentation

In-Memory RAC-Level Global Dictionary

Database In-Memory Join Groups now support Global Dictionaries across RAC nodes. With Join Groups a common dictionary is shared by columns that are joined together. In-Memory RAC-level global dictionaries now synchronize these common dictionaries across nodes within a RAC database.

In a RAC environment, this feature further improves database In-Memory performance for distributed hash joins.

View Documentation

Selective In-Memory Columns

With Selective In-Memory columns, it is now easier to add or exclude columns for in-memory. An `ALL` sub-clause has been added so that all columns can be either enabled or disabled from in-memory. This reduces the need to have very long strings of included or excluded columns.

With Selective In-Memory columns, the ability to specify `ALL` columns to be enabled or disabled from in-memory reduces the need for very long strings of columns, which reduces the chance for errors and makes configuring in-memory enabled tables easier.

View Documentation

Vectorized Query Processing: Multi-Level Joins and Aggregations

This feature enhances the In-Memory Deep Vectorization framework by fully exploiting SIMD capabilities to further improve hash join and group by aggregation performance. New optimizations include incorporating multi-level hash join support, full In-Memory group by aggregation support, and support for multi-join key and additional join methods.

This feature adds improvements in the performance of joins and aggregations, which are the foundations for analytic queries. This enables faster real-time analytic

performance and requires no application SQL changes. This feature is automatically used when enabled, which is the default.

View Documentation

Machine Learning - Enhancements

Automated Time Series Model Search

This feature enables the Exponential Smoothing algorithm to select the forecasting model type automatically - as well as related hyperparameters - when you do not specify the `EXSM_MODEL` setting. This can lead to more accurate forecasting models.

This feature automates the Exponential Smoothing algorithm hyperparameter search to produce better forecasting models without manual or exhaustive search. It enables non-expert users to perform time series forecasting without detailed understanding of algorithm hyperparameters while also increasing data scientist productivity.

View Documentation

Explicit Semantic Analysis Support for Dense Projection with Embeddings in OML4SQL

The unstructured text analytics algorithm Explicit Semantic Analysis (ESA) is able to output dense projections with embeddings, which are functionally equivalent to the popular doc2vec (document to vector) representation.

Producing a doc2vec representation is useful as input to other machine learning techniques, for example, classification and regression, to improve their accuracy when used solely with text or in combination with other structured data. Use cases include processing unstructured text from call center representative notes on customers or physician notes on patients along with other customer or patient structured data to improve prediction outcomes.

View Documentation

GLM Link Functions

The in-database Generalized Linear Model (GLM) algorithm now supports additional link functions for logistic regression: probit, cloglog, and cauchit.

These additional link functions expand the set available to match standard Generalized Linear Model (GLM) implementations. They enable increasing model quality, for example, accuracy, by handling a broader range of target column data distributions and expand the class of data sets handled. Specifically, the probit link function supports binary (for example, yes/no) target variables, such as when predicting win/lose, churn/no-churn, buy/no-buy. The asymmetric link function complementary-log-log (cloglog) supports binary target variables where one outcome is relatively rare, such as when predicting time-to-relapse of medical conditions. The cauchit link function supports handling data with, for example, data recording errors, more robustly.

View Documentation

Improved Data Prep for High Cardinality Categorical Features

This feature introduces the setting `ODMS_EXPLOSION_MIN_SUPP` to allow more efficient, data-driven encoding for high cardinality categorical columns. You can adjust the threshold (define the minimum support required) for the categorical values in explosion mapping or disable the feature, as needed.

This feature introduces a more efficient, data-driven encoding of high cardinality categorical columns, allowing users to build models without manual data preparation of such columns.

It efficiently addresses large datasets with millions of categorical values by recoding categorical values to include only those with sufficient support, enabling you to overcome memory limitations.

View Documentation

Lineage: Data Query Persisted with Model

This feature enables users to identify the data query that was used to provide the training data for producing a model. The `BUILD_SOURCE` column in the `ALL/DBA/USER_MINING_MODELS` view enables users to access the data query used to produce the model.

This feature records the query string that is run to specify the build data, within the model's metadata to better support the machine learning lifecycle and MLOps.

View Documentation

Multiple Time Series

The Multiple Time Series feature of the Exponential Smoothing algorithm enables conveniently constructing Time Series Regression models, which can include multivariate time series inputs and indicator data like holidays and promotion flags. It enables constructing Time Series Regression models to include multivariate time series inputs and indicator data like holidays and promotion flags.

This feature automates much of what a data scientist would perform manually by generating backcasts and forecasts on one or more input time series, where the target time series also receives confidence bounds. The result is used as input to other ML algorithms, for example, to support time series regression using XGBoost, with multivariate categorical, numeric, and time series variables.

View Documentation

OML4Py and OML4R Algorithm and Data Type Enhancements

The Oracle Machine Learning for Python (OML4Py) API exposes additional in-database machine learning algorithms, specifically Non-negative Matrix Factorization (NMF) for feature extraction, Exponential Smoothing Method (ESM) for time series forecasting, and Extreme Gradient Boosting (XGBoost) for classification and regression. OML4Py introduces support for date, time, and Integer datatypes.

The Oracle Machine Learning for R (OML4R) API exposes additional in-database machine algorithms, specifically Exponential Smoothing Method (ESM) for time series forecasting, Extreme Gradient Boosting (XGBoost) for classification and regression, Random Forest for classification, and Neural Network for classification and regression.

The enhancements to OML4R and OML4Py further enable Oracle AI Database as a platform for data science and machine learning, providing some of the most popular in-database algorithms from Python and R.

The additional in-database algorithms enable use cases such as demand forecasting using ESM, churn prediction and response modeling using Random Forest, and generating themes from document collections using NMF. As a feature extraction algorithm, NMF supports dimensionality reduction and as a data preparation step prior to modeling using other algorithms. XGBoost is a popular classification and regression algorithm due to its high predictive accuracy and also supports the machine learning technique survival analysis. Random Forest is a popular classification algorithm due to its high predictive accuracy. Neural Network is a

classification and regression algorithm that is well-suited to data with noisy and complex patterns, such as found in sensor data, and provides fast scoring.

The OML4Py support for date, time, and integer data types enables operating on database tables and views that contain those data types, for example, to transform and prepare data at scale in the database.

View Documentation

Outlier Detection using Expectation Maximization (EM) Clustering

The Expectation Maximization algorithm is expanded to support distribution-based anomaly detection. The probability of anomaly is used to classify an object as normal or anomalous. The EM algorithm estimates the probability density of a data record, which is mapped to a probability of an anomaly.

Using Expectation Maximization (EM) for anomaly detection expands the set of algorithms available to support anomaly detection use cases, like fraud detection. Since different algorithms are capable of identifying patterns in data differently, having multiple algorithms available is beneficial when addressing machine learning use cases.

View Documentation

Partitioned Model Performance Improvement

This feature improves the performance for a high number of partitions (up to 32K component models) in a partitioned model and speeds up the dropping of individual models within a partitioned model.

Machine learning use cases often require building one model per subset of data, e.g., a model per state, region, customer, or piece of equipment. The partitioned models capability already automated the building of such models - providing a single model abstraction for simplified scoring - and this enhancement improves overall performance when using larger number of partitions.

View Documentation

XGBoost Support for Constraints and for Survival Analysis in OML4SQL

The in-database XGBoost algorithm is enahnced to support the machine learning technique survival analysis, as well as feature interaction constraints and monotonic constraints. The constrainsts allow you to choose how variables are allowed to interact.

Survival analysis is an important machine learning technique for multiple industries. This enhancement enables increased model accuracy when predicting, for example, equipment failures and healthcare outcomes. Specifically, this supports data scientists with the Accelerated Failure Time (AFT) model - one of the most used models in survival analysis – to complement the Cox proportional hazards regression model.

Interaction and monotonic constraints provide for greater control over the features used to achieve better predictive accuracy by leveraging user domain knowledge when specifying interaction terms.

View Documentation

Machine Learning - Enhancements for R

Exponential Smoothing Method (ESM) for Time Series Forecasting

Exponential Smoothing is a moving average method with a single parameter which models an exponentially decreasing effect of past levels on future values. This in-database algorithm is exposed through the R API of Oracle Machine Learning for R.

Exponential Smoothing Methods have been widely used in forecasting for over half a century. It has applications at the strategic, tactical, and operation level. Being exposed as part of the R API, you have native R access to this in-database algorithm.

View Documentation

In-Database Neural Network Algorithm in OML4R

The in-database Neural Network algorithm allows you to address classification and regression use cases.

Neural networks are well-suited to data with noisy and complex patterns, such as found in sensor data, and provide fast scoring. Being exposed as part of the R API, you have native R access to this in-database algorithm.

View Documentation

In-Database Random Forest for Classification in OML4R

The Random Forest algorithm provides an ensemble learning technique for classification.

Random Forest is a popular classification algorithm due to its high predictive accuracy. You can now use this in-database algorithm through the R API of Oracle Machine Learning for R.

View Documentation

XGBoost Support for Classification and Regression in OML4R

XGBoost is a scalable gradient tree boosting algorithm that supports both classification and regression. The in-database implementation makes available the XGBoost Gradient Boosting open source package.

XGBoost is a popular classification and regression algorithm due to its high predictive accuracy and its support for the machine learning technique survival analysis. Being exposed as part of the R API, you have native R access to this in-database algorithm.

View Documentation

Machine Learning - Enhancements for Python

Exponential Smoothing Method (ESM) for Time Series Forecasting in OML4Py

Exponential Smoothing is a moving average method with a single parameter which models an exponentially decreasing effect of past values. This in-database algorithm is exposed through the Python API of Oracle Machine Learning for Python.

Exponential Smoothing Methods have been widely used in forecasting for over half a century. It has applications at the strategic, tactical, and operational levels. Being exposed as part of the Python API, you have native Python access to this in-database algorithm.

View Documentation

Non-Negative Matrix Factorization Support for Dimensionality Reduction in OML4Py

Non-Negative Matrix Factorization (NMF) is a state-of-the-art feature extraction algorithm. You can now use this in-database algorithm through the Python API of Oracle Machine Learning for Python.

NMF is useful when there are many attributes, and those attributes are ambiguous or have weak predictability. By combining attributes through linear combinations, NMF can produce meaningful patterns, topics, or themes.

View Documentation

Support for Date, Time, and Integer Data Types in OML4Py

OML4Py introduces support for date, time, and Integer data types.

The OML4Py support for date, time, and integer data types enables you to create pandas DataFrame proxy objects and operate on database tables and views that contain those data types. This enables you to explore and prepare data at scale in the database.

View Documentation

XGBoost for Classification and Regression in OML4Py

XGBoost is a scalable gradient tree boosting algorithm that supports both classification and regression. The in-database implementation makes available the XGBoost Gradient Boosting open source package.

XGBoost is a popular classification and regression algorithm due to its high predictive accuracy. Being exposed as part of the Python API, you have native Python access to this in-database algorithm.

View Documentation

Spatial

Spatial: 3D Models and Analytics

The point cloud feature of Oracle AI Database supports change detection through SQL and PL/SQL APIs.

This feature automates discovery of relevant changes between two point clouds, enabling easy inclusion in applications, such as modeling changes in forest canopies, assessing damages to landscape due to fire, flood, landslides, or earthquakes, and measuring progress over time in infrastructure projects.

View Documentation

Spatial: REST APIs for GeoRaster

Oracle AI Database includes a comprehensive set of REST APIs for working with GeoRaster data such as satellite imagery.

In addition to existing PL/SQL and Java APIs, developers can use REST APIs to perform GeoRaster query and data manipulation operations. This feature simplifies the development of cloud applications which frequently depend on REST APIs.

View Documentation

PL/SQL API to Generate Spatial Vector Tiles for Map Visualization

Developers can use SQL to generate vector tiles, a dynamic mapping technology, to convert geometries stored in the Oracle AI Database into vector tiles for efficient map rendering capabilities in web applications.

The utilization of vector tiles in map visualization enables businesses to deliver customizable, efficient, and engaging map experiences.

View Documentation

Workspace Manager: Improved Security when using Oracle Workspace Manager

Database users can have workspace manager objects in their own schema instead of in the WMSYS schema.

Oracle Workspace Manager enables collaborative development, what-if analysis from data updates, and maintains a history of changes to the data. Developers can create multiple workspaces and group different versions of table row values in different workspaces. With this enhancement, developers have improved security when using this feature. All the workspace manager objects can be stored and invoked in their own user schema in Oracle Autonomous AI Database and user-managed databases.

View Documentation

7 Data Warehousing/Big Data

This section describes the new data warehousing/big data features.

Data Type Override for Iceberg Tables

This enhancement lets you define the Oracle data type mapped to each column in an Iceberg table, enabling inserts using a wider range of Oracle data types. For example, you can map an Iceberg `LIST` or `ARRAY` type to Oracle `VECTOR`, or an unlimited-length Iceberg `STRING` type to a fixed-length or extended `VARCHAR`, or to `CLOB`.

Data Type Override gives users control over how Iceberg column types are mapped to Oracle data types during inserts. This enables loading a wider range of Oracle data types into Iceberg tables using types that better fit the data and its intended use. This flexibility can reduce the need to reshape data before loading and make Iceberg data easier to use with Oracle applications and vector workloads.

View Documentation

Enhanced Partitioning Metadata

Data dictionary views that contain partitioning-related metadata, for example, `ALL_TAB_PARTITIONS`, have two additional columns representing the high value (boundary) information of partitions and subpartitions in JSON and CLOB format.

Providing the high value (boundary) partitioning information in JSON and as CLOB allows you to use this information programmatically. This enables simple and automated processing of this information for schema retrieval or lifecycle management operations.

View Documentation

Extended Language Support in Oracle Text

Language support is extended in Oracle Text, now supporting up to 48 languages. Additionally, there is extended support for all languages. To avoid the extended language support increasing your install footprint on disk, a new mechanism is introduced to control the downloaded languages on demand.

Customers benefit from the improved support for languages and can download the required files for only the languages they support. They can avoid using disk space unnecessarily for unneeded languages.

View Documentation

External Table Partition Values in the File Path

External tables pointing to data in the object store can consist of a large number of files. These files can be organized across multiple directories, and even directory trees. You can use external table partitioning with folder names as part of the file paths. External table columns can also now return the file name of the source file for each row. The partition values can be derived from the directory name or file name.

External tables and SQL*Loader can load large numbers of data files in object stores and meet the requirements for Hive-generated tables organized across multiple directories, and even directory trees. This feature enables external table partitioning based upon the directory and file name; for example, you can have files for different months or for different states in separate directories.

View Documentation

Iceberg Table Properties

Iceberg table properties for Oracle Database are stored in the metadata of an Iceberg table and enable users to configure and manage table-specific settings that influence Iceberg behavior and interoperability. Table properties can be specified during table creation or updated later using SQL, providing a convenient way to customize an Iceberg table without modifying its schema.

Iceberg table properties provide a standardized and flexible way to configure table-specific behavior without modifying table definitions or application code. By supporting Iceberg table properties, Oracle Database improves interoperability with the Apache Iceberg ecosystem, enables workload-specific tuning, and simplifies Data Lake management. Organizations can consistently manage storage, performance, and maintenance settings across Iceberg tables while maintaining compatibility with other Iceberg-enabled compute engines.

View Documentation

Iceberg Update and Delete

Iceberg tables accessed through Oracle Database now support `DELETE` and `UPDATE` statements. The feature applies to both non-partitioned and partitioned Iceberg tables and provides familiar SQL DML capabilities for maintaining Lakehouse data.

This feature permits mutation and removal of data in Iceberg tables directly from an Oracle AI Database. Applications and administrators can use standard SQL to correct, maintain, and remove Lakehouse data without first copying it into an Oracle-managed table or relying on a separate processing engine.

View Documentation

Logical Partition Change Tracking for Materialized View Refresh and Staleness Tracking

Logical Partition Change Tracking (LPCT) tracks the staleness of materialized views. LPCT operates at a fine level of logical granularity and gives you the flexibility to align the boundaries of logical partitions with the business rules and with changes applied to tables. It evaluates the staleness of the base tables for individual logical partitions without using a materialized view log or requiring any of the tables used in the materialized view to be partitioned.

With LPCT, materialized view staleness is tracked at the granularity of the logical partitions. This functionality significantly broadens the applicability of query rewrite for your application due to the fine-grained query rewrite. With LPCT, you perform refresh operations targeted at stale logical partitions only, improving the refresh time and avoiding complete re-loading data.

View Documentation

Staging Tables

Staging tables are heap tables optimized for fast data ingestion and for handling volatile data. Key table attributes are set to defaults for these use cases without any additional user interaction.

Creating staging tables rather than 'normal' tables saves you time and effort so that you do not need to tune your table attributes for fast data ingestion with volatile data content. A staging table is configured by default with optimal configuration settings in

order to guarantee the best possible performance and to avoid unnecessary performance debugging and tuning.

View Documentation

8 Cloud Migration

Oracle Cloud makes migration to Oracle AI Database 26ai more reliable. It provides a tool to transparently move your Advanced Queue events to the new Transactional Event Queues. The Cloud Premigration Advisor tool proactively identifies, reports, and recommends solutions to repair incompatibilities between your source database and an Oracle Cloud database or on-premises target of interest.

Classic Queue to Transactional Event Queues (TxEventQ) Online Migration Tool

Advanced Queuing (AQ) is a key feature of the Oracle AI Database used to build application workflows using events. Queues in the Oracle AI Database come in two flavors - Classic Queues and the new and improved Transactional Event Queues (TxEventQ). TxEventQ are Kafka-like and highly performant due to their persistent in-memory caching implementation.

Oracle AI Database 26ai introduces an online migration tool to migrate from AQ Classic Queues to the new TxEventQ. The database creates a new TxEventQ with aliased name and routes enqueues to it, while it dequeues drain messages from the existing AQ. After the messages are drained, the migration is complete and the TxEventQ takes the name of the AQ queue. The TxEventQ migration interface checks for compatibility and adaptability for migration of an AQ deployment, and can commit or roll back a migration after running some tests on TxEventQ.

Existing AQ customers interested in higher throughput queues and with Kafka compatibility using a Kafka Java Client and Confluent like REST APIs, can migrate easily from AQ to Transactional Event Queues (TxEventQ).

TxEventQ is Oracle's next generation messaging system that offers many benefits and features over Advanced Queuing (AQ), like scalability, performance, key-based partitioning, and Kafka compatibility with a Java client, TxEventQ natively supports JSON payload, which makes event-driven microservices/application writing easier in multiple languages - Java, JavaScript, PL/SQL, Python, etc.

View Documentation

Cloud Premigration Advisor Tool for Source Database Migrations to Other Oracle AI Databases

The Cloud Premigration Advisor Tool (CPAT) assists you with database migration, both on-premises and in Oracle Cloud. It assesses the characteristics of the source database for migration and determines whether the source database can be migrated successfully to another Oracle AI Database. The JSON output from CPAT can be read by tools and applications for further processing and reporting.

CPAT helps you to avoid incompatibility issues in migrations. It helps you save time and effort in planning your migration of on-premises and Oracle Cloud databases to Oracle Autonomous AI Database.

View Documentation

9 Cloud Operations

Cloud operations is the process of managing and optimizing cloud-based infrastructure, databases, and services. Features in this category enhance the availability, monitoring, maintenance, diagnosability, and security of cloud-based database deployments and reliably guarantees mission-critical high availability and resiliency.

Manageability

Hybrid Read-Only Mode for Pluggable Databases

Administrators can configure pluggable databases (PDBs) to operate in a new mode called hybrid read-only. Hybrid read-only mode enables a PDB to operate as either read-write or read-only, depending on the user who is connected to the PDB. For common users, the PDB will be in read-write mode. For local users, the PDB will be restricted to read-only mode.

Hybrid read-only mode enables you to patch and maintain an application in a safe mode for open PDBs without the risk of local users, including higher privileged ones, interfering with the ongoing maintenance operation of the PDB.

View Documentation

Real-Time SQL Monitoring Enhancements

Real-time SQL Monitoring works independently and concurrently across multiple PDB containers in an efficient manner. SQL statements, PL/SQL procedures and functions, and DBOPs (Database Operations) are monitored at PDB and CDB levels. You can efficiently query SQL Monitor reports across ad-hoc time ranges, DBIDs (internal database identifiers), and CON_DBIDs (CDB identifiers). This data is also accessible through SQL History Reporting.

Additionally, SQL Monitoring data can be exported along with the Automatic Workload Repository (AWR) and imported into another database or container for longer term storage and analysis.

Real-time SQL Monitoring is now supported per-PDB and CDB levels efficiently by default. As a PDBA persona, you can get a more accurate view of the monitored SQL for your application.

SQL Monitoring data can be transported through the AWR framework to a different container or database for longer term storage and offline analysis.

View Documentation

Control PDB Open Order

Administrators can define a startup order or priority for each pluggable database (PDB) where the most important PDBs are started first. The priority is applied to PDB opening order and upgrade order as follows:

- Restoring PDB states when opening the CDB
- Setting PDB states when using the `PDB OPEN ALL` statement
- Setting the order for PDB database upgrade operations
- Starting PDBs in an Active Data Guard (ADG) switchover or failover

This feature allows critical PDBs to start and open before less important PDBs, reducing the time for the critical applications to become usable.

View Documentation

Inter-Instance Resource Management

Inter-Instance Resource Management is a preemptive task scheduling and resource management capability that enables fine-grained control over CPU resources across multiple Container Databases, Pluggable Databases, and non-Database processes residing within servers, hosts, or Virtual Machines, in clustered and non-clustered environments. Inter-Instance Resource Management includes upper limits on CPU resource consumption, as well as lower-bound guarantees to enable bursting for multiple Pluggable Databases within a Container and multiple Container Databases on a server, host, or Virtual Machine.

This feature enables effective use of system resources with high-density database consolidation.

View Documentation

Optimized Performance for Parallel File System Operations

In environments that contain many PDBs and require multiple `DBMS_FS` requests to be processed in parallel, you can update the number `OFS_THREADS` to increase the number of `DBMS_FS` requests executed in parallel. This increases the number of worker threads

executing the make, mount, unmount, and destroy operations on Oracle file systems in Oracle AI Database. This will reduce the time needed to execute parallel file system requests in environments with multiple PDBs.

This feature significantly reduces the time required to perform parallel file system requests in consolidation environments containing multiple PDBs.

View Documentation

Read-Only Users and Sessions

You can control whether a user or session is enabled for read-write operations, irrespective of the privileges of the user that is connected to the database. The `READ_ONLY` session applies to any type of user for any type of container. The `READ_ONLY` user only applies to local users.

Providing the capability to disable and re-enable the read-write capabilities of any user or session without revoking and re-granting privileges provides you with more flexibility to temporarily control the privileges of users or sessions for testing, administration, or application development purposes. It also gives you a simple way to control the read-write behavior within different parts of an application that are used by the same user or session.

View Documentation

Continuous Availability

Application Continuity Session State Restore with Database Templates

Application Continuity uses database templates to checkpoint the session state, restore the session state at the start of replay, and support session migration during planned maintenance. Database templates restore server-side and client-visible session states at the beginning of the Application Continuity replay, thus increasing Application Continuity protection.

Application Continuity with Database Templates broadens and simplifies the use of Application Continuity to reduce planned maintenance-related downtime. It also enables the migration of more sessions faster during unplanned outages, ensuring higher levels of protection.

View Documentation

Enhanced Upgrade of Time Zone Data

The process of upgrading timezone data to reflect up-to-date Governmental Daylight Saving Time rules is optimized, taking the actual data content of tables into account. Only tables impacted by a Daylight Saving Time rule change will undergo a data change.

Optimizing the necessary data changes for a Daylight Savings Time rule change significantly improves the overall upgrade of timezone data to the absolute bare minimum to bring a database up to the latest global timezone rules. The implicit analysis and reduction of the data required to change significantly reduces the overall timezone upgrade process and the resources needed.

View Documentation

Optimized Read-Write Operations for Database Processes

To optimize the read and write operations performed by database processes when you access files managed through OFS or DBFS, specify the new `db_access` mount option for the `dbms_fs.mount_oracle_fs` procedure while mounting the file system.

When you enable `db_access`, both memory consumption and CPU usage reduces. The throughput increases while performing read and write operations by database processes on the files managed by OFS.

View Documentation

Smart Connection Rebalance

Smart Connection Rebalance transparently reshuffles service-based connections based on real-time performance monitoring across Oracle Real Application Clusters (Oracle RAC) instances.

Smart Connection Rebalance improves database performance by automatically moving sessions across Oracle RAC database instances without needing manual intervention.

View Documentation

Smooth Reconfiguration of Oracle RAC Instances

The smooth reconfiguration feature reduces the brownout time caused by certain Oracle Real Application Clusters (Oracle RAC) operations, such as nodes joining or leaving an Oracle RAC cluster.

Smooth Reconfiguration of Oracle RAC Instances ensures continuous availability of Oracle RAC services and reduces brownout time for database instances running in an Oracle RAC database.

View Documentation

Support for the Coexistence of DGPDB and GoldenGate Capture

This project introduces perPDB DataGuard. When DGPDB is configured on a source/Primary database, there are validations that insure there is no GoldenGate Capture pre-existing on the source. GoldenGate capture sessions will be broken if a DGPDB is allowed and executes a role transition.

This project adds support for coexistence of DGPDB and GoldenGate Capture. Changes/support will be required in the LogMiner, redo transport, and Broker layers.

View Documentation

General

Adaptive Result Cache Object Exclusion

With adaptive result cache object exclusion, the database decides to blocklist certain objects if using the result cache is not beneficial for these objects, based on statistical evidence such as the number of invalidations, the cost savings of using result caching, and others. You have full control over the objects considered for exclusion to ensure you can continue using result cache for all your objects of interest.

Adaptive exclusion of objects that don't benefit or even have a detriment impact on result caching reduces the overall development and management workload for you. It can improve the database performance out of the box.

View Documentation

Cloud Developer Packages

The Cloud Developer packages provide built-in tools to connect, process, and integrate with cloud services seamlessly. These packages, namely `DBMS_CLOUD`, `DBMS_CLOUD_PIPELINE`, `DBMS_CLOUD_REPO`, `DBMS_CLOUD_NOTIFICATION`, and `DBMS_CLOUD_AI` collectively provide the following advantages:

- Enable easy data access to cloud storage
- Automate workflows through pipelines
- Trigger event notifications
- Integrate AI capabilities for use directly from the database

Since Oracle AI Database 26ai, Release Update 23.7, `DBMS_CLOUD_AI` supports the Select AI feature, which provides a SQL and PL/SQL interface to leverage Large Language Models (LLMs) for natural language to SQL generation (NL2SQL). The complete Select AI functionality is available on Oracle Autonomous AI Database, including NL2SQL and retrieval augmented generation (RAG), among other AI-based features.

The Cloud Developer packages enable businesses to work with data stored in the cloud, process real-time data, and use advanced tools like AI, benefiting from cloud capabilities. This approach helps simplify operations and supports specifically a mix of on-premises and cloud systems.

View Documentation

Diagnose and Repair SQL Exceptions Automatically at Compile-Time

Automatic error mitigation automatically detects and repairs many severe compile-time SQL exceptions that would otherwise cause SQL statements to fail. If a severe internal error occurs during SQL compilation, the exception is trapped, and the foreground processes will re-parse the statement to try and work around the problem. If re-parse is successful, a SQL patch is used to persist the workaround for future hard parses, and the session parsing the SQL statement will proceed without interruption and without receiving an exception.

This feature improves the robustness of your applications and its service levels.

View Documentation

Disable Statistics Gathering During Autonomous Database Import Operations

The `DATA_OPTIONS=DISABLE_STATS_GATHERING` parameter option disables statistics gathering during an import job.

Statistics gathering can have a performance impact on a large import operation. This parameter disables statistics gathering during an import job. The parameter is also supported for on-premises and user-managed cloud services import operations. Statistics gathering is not automatic in non-autonomous database environments, but this parameter can be useful. It controls whether Oracle Autonomous Database gathers environment statistics automatically for DML operations, such as data loading during an Oracle Data Pump Import job.

View Documentation

Oracle Database Cloud Backup Module for Azure Blob Storage

The Oracle Database Cloud Backup Module for Azure Blob Storage enables Oracle AI Database to send backups to and recover from Microsoft Azure Blob Storage. This backup module is compatible with Oracle AI Databases deployed on-premises or on Azure cloud. Support for writing backups to Azure Blob Storage for Oracle-managed database services is not available. Database administrators can use RMAN commands, RMAN scripts, and Oracle Enterprise Manager to perform backup and recovery operations with Azure Blob Storage.

You can run Oracle AI Databases in many different locations, and the backup module for Azure Blob Storage provides on-premises and Azure virtual machine users with more flexibility for backup storage locations.

View Documentation

Read-Only Tablespace on Object Storage

Read-only tablespaces can be moved to and from Oracle Object Storage transparently, storing portions of a database on lower-cost storage in the Cloud.

Allowing to move tablespaces to Oracle Object Storage enables a data lifecycle management strategy, storing the data on the most cost-effective storage tier based on its business value or access frequency.

View Documentation

Unified Memory

Unified Memory is a flexible and simple memory configuration for Oracle AI Databases that uses a single parameter to control database memory allocations, reducing or eliminating the need for system restart to change memory configurations. Unified Memory is especially useful in multiple workload high density database consolidation environments.

Unified Memory simplifies memory management to run multiple workloads in a highly consolidated environment with minimum disruption. It is easier to set the single parameter `MEMORY_SIZE` for configuring the database instance memory instead of using separate parameters like `SGA_TARGET` and `PGA_AGGREGATE_LIMIT`.

View Documentation

10 High Availability

This section describes the new high availability features.

Data Guard

Oracle Data Guard Redo Decryption for Hybrid Disaster Recovery Configurations

Oracle Data Guard now provides the capability to decrypt redo operations in hybrid cloud disaster recovery configurations where the cloud database is encrypted with Transparent Data Encryption (TDE) and the on-premises database is not.

Hybrid disaster recovery (DR) with Data Guard is now more flexible and easy to configure. Hybrid disaster recovery for the Oracle AI Database allows you to expand outage and data protection to take advantage of the automation and resources of Oracle Cloud Infrastructure (OCI). By enabling the ability to quickly configure disaster recovery in OCI, even in cases where on-premises databases might not already be encrypted with Transparent Data Encryption (TDE), the steps required to configure hybrid disaster recovery environments and prepare on-premises databases for a DR configuration with cloud databases in OCI have been greatly reduced.

View Documentation

Automatic Block Repair in Active Data Guard per Pluggable Databases

Active Data Guard per Pluggable Database automatically repairs corrupt data blocks at the pluggable database (PDB) level by using the good copy from the standby pluggable database, in the same way Active Data Guard does for the whole Container Database (CDB). When corruption is detected in a PDB, the database replaces bad blocks in real time, without user intervention or raising errors to the application.

This feature keeps the applications and users working, even when block corruption strikes. By fixing data block corruption instantly and transparently, this feature cuts downtime and minimizes data loss risk.

View Documentation

Automatic Standby Pluggable Database Instantiation and Standby Redo Log Creation

When adding a pluggable database in Data Guard per Pluggable Database environments, the broker automatically copies datafiles to the standby container database and creates standby redo logs with the proper number and size.

These enhancements streamline Data Guard per Pluggable Database environments and reduce the number of manual operations, minimizing risk of configuration errors.

View Documentation

Configuration and Member Tagging for Oracle Data Guard

Oracle Data Guard Broker lets you assign user-defined key-value tags to configuration members. Tags make it easy to logically group members using details like location, life cycle status, organization, or compliance needs. Choose any naming scheme to match your business processes.

Tagging enables teams to organize configuration members according to business needs. It improves monitoring and management, supports compliance tracking, and helps classify members while reducing operational errors. As a result, teams gain better control over the Data Guard environment.

View Documentation

DML Redirection for Data Guard per Pluggable Database

Data manipulation statement (DML) redirection now also operates at the pluggable database level in Active Data Guard per Pluggable Database environments. This enhancement directs write operations transparently from the standby to the primary pluggable database.

DML redirection transforms the standby pluggable database from a passive replica into an active, high-value asset. Applications can run mixed read and write workloads directly from the standby PDB. As a result, the standby database delivers greater return on investment by offloading more application activity and increasing the flexibility of disaster recovery environments.

View Documentation

JSON Output for the Data Guard Command-Line

The Data Guard command-line (DGMGRL) now presents output in JSON format. Users can choose this structured format to make data straightforward to process and reuse, moving beyond plain-text output.

JSON-formatted output enables easy integration with automation tools and scripting environments. Customers can parse results programmatically, connect workflows, and boost productivity by eliminating manual handling of plain-text results.

View Documentation

Per-PDB Data Guard Integration Enhancements

The new Oracle Data Guard per Pluggable Database architecture provides more granular control over pluggable databases, which can now switch and fail over independently. The enhancements to the Oracle Data Guard per Pluggable Database include simplified setup and validation, automatic addition of temporary files, improved management and housekeeping, and target pluggable databases open for query off-loading (Real-Time Query feature at the pluggable database level).

You can rely on Data Guard protection while keeping the flexibility and high consolidation rates that the multitenant architecture offers, reducing operational costs.

View Documentation

Restrict Switchover and Failover to a List of Databases

The Oracle Data Guard broker property `PrimaryDatabaseCandidates` lists which databases can become primary during switchover or failover. This property enforces finer control over role transitions, helping customers prevent the activation of undesired databases. An optional override mode lets you perform an emergency failover without changing the policy.

This feature increases control and predictability in managed database environments. Organizations can enforce business, regulatory, or operational rules about which databases may act as the primary, reducing risk from operational errors. The override command ensures critical failover remains possible during emergencies, while surfacing appropriate warnings.

View Documentation

TLS Authentication for Redo Transport

Oracle Data Guard enables use of TLS authentication for redo transport connections. Strong external authentication can be used instead of the password file method, removing the need for password file transfers and centralizing management across multiple databases.

Centralized and strong authentication for the redo transport reduces operational risk and strengthens security controls. Organizations gain efficient key management and increased flexibility, especially in complex data replication scenarios, such as GoldenGate downstream capture, by eliminating shared password dependencies. This enhancement fosters stronger security governance and easier compliance.

View Documentation

Transport-Lag-Based Auto Redo Prioritization

Oracle Data Guard prioritizes redo generation for ASYNC redo transport to a physical standby database when transport lag exceeds the configured threshold. Redo-intensive batch or maintenance sessions are assigned a lower redo-generation priority, allowing OLTP sessions to maintain normal responsiveness and transactional throughput while redo transport catches up. This helps keep the standby database closer to the primary without changing the configured data protection mode.

Physical standby databases stay within the configured transport lag target more consistently, helping lower the recovery point objective and strengthen fast-start failover protection. At the same time, OLTP responsiveness and transactional throughput remain preserved.

View Documentation

General

Application Continuity Support for DBMS_ROLLING

Application Continuity and the draining of database sessions are now supported when performing a rolling upgrade or applying non-rolling patches using `DBMS_ROLLING`.

Using Application Continuity with `DBMS_ROLLING`, applications are continuously available during the database upgrade process.

View Documentation

Database Native Transaction Guard

Transaction Guard is an application-independent infrastructure that enables recovery of work from an application perspective. With Transaction Guard, each logical transaction may map to single or multiple server-side transactions. Persisting each logical transactions, as part of a commit, introduces overheads in normal transaction operation. Database Native Transaction Guard enhances existing Transaction Guard and does not require persistence in a separate table.

Database Native Transaction Guard does not incur extra redo generation or performance overhead (extra writes are eliminated) and no client-side changes are required.

View Documentation

Flashback Time Travel Enhancements

Flashback Time Travel can automatically track and archive transactional changes to tables. Flashback Time Travel creates archives of the changes made to the rows of a table and stores the changes in history tables. It also maintains a history of the evolution of the table's schema. By maintaining the history of the transactional changes to a table and its schema, Flashback Time Travel enables you to perform operations, such as Flashback Query (`AS OF` and `VERSIONS`), on the table to view the history of the changes made during transaction time.

Flashback Time Travel helps to meet compliance requirements based on record-stage policies and audit reports by tracking and storing transactional changes to a table, which has also been made more efficient and performant in this release.

View Documentation

General Improvements for Oracle FPP Job Scheduler

The Oracle Fleet Patching and Provisioning (Oracle FPP) job scheduler now supports pausing and resuming jobs, forcing jobs to start in a paused state, and pausing jobs between batches in a chain execution and allows to tag jobs for easy querying.

Additional job scheduler options provide you with more control over the patching execution. You can pause jobs to verify executed jobs and resolve dependencies before resuming jobs.

View Documentation

Intelligent Drain Timing

Intelligent drain timing uses client-side request duration distributions and in-flight request tracking to predict when work will complete, recommend optimal drain windows, and dynamically guide drain timing without relying on fixed timeouts.

This feature removes the guesswork from maintenance planning, minimizing downtime that impacts the user experience.

View Documentation

MAA Client Readiness

MAA Client Readiness provides real-time client behavior and historical request patterns to determine whether sessions are drainable or failover-ready. This provides actionable guidance to ensure that applications can achieve safe, interruption-free maintenance.

This feature minimizes application disruption by identifying which database sessions can safely drain or fail over during maintenance or load balancing operations.

View Documentation

Optimized Fast-Start Failover Delay Detection in Maximum Performance Mode

Oracle Data Guard Fast-Start Failover has two additional properties for improved lag detection and status changes. `FastStartFailoverLagType` sets the lag type that Fast-Start Failover must consider when in Maximum Performance mode (`APPLY` or `TRANSPORT`). `FastStartFailoverLagGraceTime` lets the configuration transition to a pre-emptive `LAGGING` state that the observer can acknowledge before reaching the actual lag limit, so the status can transition immediately to `TARGET OVER LAG LIMIT` without waiting for the observer quorum.

The new properties for the Maximum Performance protection mode further enhance Fast-Start Failover capabilities and reduce the impact on application transactions for status changes requiring the observer quorum.

View Documentation

Oracle DBCA Support for Standard Edition High Availability

Using the Oracle Database Configuration Assistant (Oracle DBCA) and facilitating Oracle's Automatic Storage Management or Oracle's Advanced Cluster File System, you can now quickly create a Standard Edition High Availability Oracle AI Database fully configured for automatic failover.

A Standard Edition High Availability Oracle AI Database can now be created very easily with more automation, eliminating manual steps and the associated complexity.

View Documentation

Oracle RAC Two-Stage Rolling Updates

The Oracle Real Application Clusters (Oracle RAC) rolling update feature enables you to apply certain non-rolling fixes in a rolling fashion. Fixes that you implement using this framework are disabled by default. You can choose to enable these fixes after all the nodes are patched successfully.

Oracle RAC two-stage rolling updates reduce the need for downtime to apply non-rolling patches. This feature significantly reduces the number of Oracle RAC non-rolling patches.

View Documentation

Transaction Guard Support during DBMS_ROLLING

Transaction Guard support for `DBMS_ROLLING` ensures continuous application operation during the switchover issued by `DBMS_ROLLING` to Transient Logical Standby. The procedure uses the last commit outcome of transactions part of in-flight sessions during a switchover-related outage (or caused by an error/timeout) to protect the applications from duplicate submissions of the transactions on replay.

Application Continuity supported by Transaction Guard during database upgrades using `DBMS_ROLLING` ensures that commit outcomes are guaranteed across the entire upgrade process.

View Documentation

Real Application Clusters

General Purpose Cluster Configuration

The General Purpose Cluster is a new cluster deployment that requires minimum network and storage configuration and supports applications that benefit from managed server restarts and failover capabilities. Oracle Grid Infrastructure installer provides a streamlined option to configure a general-purpose cluster in either interactive or silent mode.

This feature increases productivity and reduces the required deployment time by simplifying deployment requirements.

View Documentation

Local Rolling Database Maintenance

Local Rolling Database Maintenance creates a new local database home and starts a second instance of the same database from the new home on the same server, allowing you to perform rolling patching and maintenance operations locally on one node of a multi-node Oracle Real Application Clusters (Oracle RAC) or Oracle RAC One Node cluster.

Local Rolling Database Maintenance provides uninterrupted database availability during maintenance activities (such as patching) for Oracle RAC and Oracle RAC One Node databases. This feature significantly improves the availability of your databases while limiting the impact on other nodes in the cluster.

View Documentation

Single-Server Rolling Database Maintenance

Single-Server Rolling Database Maintenance creates a new local database home and starts a second instance of the same database from the new home on the same server, allowing you to perform rolling patching and maintenance operations on a single server hosting an Oracle RAC One Node or Real Application Clusters (Oracle RAC) database.

Single-Server Rolling Database Maintenance provides database availability during maintenance activities (such as patching) on a single server hosting an Oracle RAC or Oracle RAC One Node database. This feature significantly improves the availability of

your single-node databases without expanding them to a multi-node cluster and adding support for shared storage.

View Documentation

11 Security

Oracle AI Database 26ai is packed with new features that help you reduce risk, better secure your data, and achieve regulatory compliance objectives. From marquee new features like the new SQL Firewall through standards updates like TLS 1.3 support to small (but important) changes like increasing the maximum length of a password from 30 bytes to 1024 bytes, you'll find this newest release a step up from your older database versions.

SQL Firewall

Oracle SQL Firewall Included in Oracle AI Database

A new feature of Oracle Database Vault, SQL Firewall is built into Oracle AI Database. SQL Firewall inspects all incoming SQL statements and ensures that only explicitly authorized SQL is run. When licensed, you can use SQL Firewall to control which SQL statements are allowed to be processed by the database. You can restrict connection paths associated with database connections and SQL statements. Unauthorized SQL can be logged and blocked. Because SQL Firewall is embedded in Oracle AI Database, it cannot be bypassed. All SQL statements are inspected, whether local or network, or encrypted or clear text. It examines top-level SQL, stored procedures and the related database objects. Consult Table 1-11 of the *Oracle Audit Vault and Database Firewall Licensing Information* for more information on licensing requirements for SQL Firewall.

SQL Firewall provides real-time protection against common database attacks by restricting database access to only authorized SQL statements or connections. It mitigates risks from SQL injection attacks, anomalous access, and credential theft or abuse.

SQL Firewall uses session context data such as IP address, operating system user name, and operating system program name to restrict how a database account can connect to the database. This helps mitigate the risk of stolen or misused application service account credentials. A typical use case for SQL Firewall is for application workloads.

You can use SQL Firewall in both the root and a pluggable database (PDB).

View Documentation

Encryption

Transport Layer Security (TLS) 1.3 Now Supported in Oracle AI Database

Transport Layer Security (TLS) version 1.3 is supported in Oracle AI Database 26ai. TLS 1.3 is the latest and most secure TLS protocol to protect network connections to and from Oracle AI Database.

Because TLS 1.3 handles initial session setup more efficiently than prior TLS versions, users moving to TLS 1.3 should see improvements in TLS performance, particularly for applications that frequently connect and reconnect to the database. TLS 1.3 also implements newer, more secure cipher suites that improve confidentiality of data in transit.

View Documentation

Strict DN Matching with Both Listener and Server Certificates

The behavior of the `SSL_SERVER_DN_MATCH` parameter has changed. Previously, Oracle AI Database performed the DN check only with the database server certificate, and both the `HOSTNAME` and the `SERVICE_NAME` setting in the connect string could be used for a partial DN match.

Oracle AI Database 26ai checks both the listener and server certificates. In addition, the `SERVICE_NAME` setting in the connect string is not used to check during a partial DN match. The `HOSTNAME` setting can still be used for partial DN matching with the certificate DN and subject alternative name (SAN), on both the listener and server certificates.

When set to `TRUE`, the `SSL_ALLOW_WEAK_DN_MATCH` parameter reverts `SSL_SERVER_DN_MATCH` to the behavior earlier than Oracle AI Database 26ai and enables DN matching to only check the database server certificate (but not the listener) and enable the service name to be used for partial DN matching.

DN matching with both the listener and server certificates provides better security to ensure that the client is connecting to the correct database server. The service name setting is also removed from `SSL_SERVER_DN_MATCH` for better security and partial DN matching can still be performed with the `HOSTNAME` connect string parameter with the he certificate DN and subject alternative name (SAN) matching.

The `SSL_ALLOW_WEAK_DN_MATCH`, though new to this release, is marked as deprecated because it is considered a temporary solution to enable the behavior of `SSL_SERVER_DN_MATCH` prior to Oracle AI Database 26ai.

View Documentation

Simplified Transport Layer Security Configuration

The Transport Layer Security (TLS) configuration between the database client and server has been simplified with streamlined parameters, performance improvements, and an additional parameter to find a wallet. Older TLS protocols have also been removed.

These changes improve security and make it easier to implement TLS.

View Documentation

Ability to Configure Transport Layer Security Connections Without Client Wallets

An Oracle AI Database client is no longer required to provide a wallet to hold well-known CA root certificates if they are available in the local system. The Oracle AI Database wallet search order determines the location (Windows (Microsoft Certificate Store) or Linux) of these certificates in the local system.

Transport Layer Security (TLS) requires either one-way authentication or two-way authentication. In one-way TLS authentication, which is commonly used for HTTPS connections, you will no longer need to install and configure a client wallet to hold the server's CA certificate as long as it is already available in the local system. If the server's CA certificate is not installed in the local systems, client wallet is still required. Starting in this release, you no longer need to install and configure a wallet to hold a well-known root certificate if it is already available in the local system.

This feature greatly simplifies the Oracle AI Database client installation and the use of TLS protocol to encrypt Oracle AI Database client-server communications.

View Documentation

New sqlnet.ora Parameter to Prevent the Use of Deprecated Cipher Suites

You can block the use of deprecated cipher suites by setting the `SSL_ENABLE_WEAK_CIPHERS sqlnet.ora` parameter to `FALSE`.

Removing the ability to use older, less secure cipher suites improves protection for data in-motion between the database.

View Documentation

AES-XTS Encryption Mode Support for TDE Tablespace Encryption

Transparent Database Encryption (TDE) tablespace encryption now supports Advanced Encryption Standard (AES) XTS (XEX-based mode with ciphertext stealing mode) in `CREATE TABLESPACE` statements. In previous releases, TDE used AES-CFB cipher mode.

AES-XTS provides improved security and better performance, especially on platforms where TDE can take advantage of parallel processing and specialized instructions built into processor hardware.

View Documentation

Changes for TDE Encryption Algorithms and Modes

The default encryption algorithm for both TDE column encryption and TDE tablespace encryption is now AES256. The previous default for TDE column encryption was AES192. For TDE tablespace encryption, the default was AES128.

The decryption libraries for the GOST and SEED algorithms are deprecated. New keys cannot use these algorithms. The encryption libraries for both of these libraries are desupported.

The column encryption mode is now Galois/Counter mode (GCM) instead of cipher block chaining (CBC), and the tablespace keys are now used in tweakable block ciphertext stealing (XTS) operating mode instead of cipher feedback (CFB).

The Oracle Recovery Manager (RMAN) integrity check for column encryption keys now uses SHA512 instead of SHA1.

The keys for Oracle RMAN and column keys are now derived from SHA512/AES for key generation. In previous releases, they used SHA-1/3DES as a pseudo-random function.

These enhancements enable your Oracle AI Database environment to use the latest, most secure algorithms and encryption modes.

View Documentation

Improved and More Secure Local Auto-Login Wallets

A local auto-login wallet is now more tightly bound to the host where it was created or modified (both bare metal and virtual). The local auto-login process is also more secure, does not require additional deployment requirements, and does not require root access.

Local auto-login wallets are more secure now and support both bare metal and virtual environments.

This enhancement also applies to Transparent Data Encryption (TDE) local auto-login keystores.

View Documentation

Changes to DBMS_CRYPTO

The following updates have been made to the DBMS_CRYPTO package:

- Added XTS mode to AES algorithms and set it as the default mode
- Added SHA-3
- Added SM2/3/4

Customers can use the latest cryptographic features with Oracle AI Database.

View Documentation

New Parameter to Control the TDE Rekey Operations for Oracle Data Guard

You now can use the `DB_RECOVERY_AUTO_REKEY` initialization parameter for Oracle Data Guard environments. `DB_RECOVERY_AUTO_REKEY` controls whether an Oracle Data Guard standby database recovery operation automatically performs the corresponding tablespace rekey when it encounters a redo that says the primary database has performed a tablespace rekey operation.

This feature is useful for standby deployments with large tablespaces whose users must perform an online TDE conversion.

View Documentation

Post-Quantum Cryptography Support

Post-Quantum Cryptography (PQC), also known as quantum-resistant or quantum-safe cryptography, is a new class of cryptographic algorithms designed to be secure against attacks by future, large-scale quantum computers.

Oracle AI Database supports the use of two PQC algorithms standardized by the National Institute of Standards and Technology (NIST): ML-KEM and ML-DSA.

PQC algorithms are based on different mathematical problems—such as those related to lattices, hash functions, or codes—that are believed to be difficult for both classical and quantum computers to solve. The transition to PQC is a critical, proactive measure to protect long-term data, such as medical records and intellectual property, from being harvested today and decrypted in the future once a quantum computer becomes available.

Oracle AI Database 26ai supports ML-KEM and a ML-KEM and ECDHE based hybrid for key-exchange secrets for TLS sessions. A hybrid KEM combines the key-exchange secrets from both ML-KEM and ECDHE to produce a single shared secret for the TLS session. A hybrid KEM can provide a critical security fallback in case either individual algorithm fails.

In addition, Oracle AI Database 26ai supports creating self-signed ML-DSA certificates, reading ML-DSA certifications, and adding ML-DSA certificates to Oracle wallet with JDK 24 or later.

View Documentation

ezTLS

ezTLS is a simple way to encrypt communication between an Oracle Database client and server using TLS-based protection without requiring customers to configure certificates or wallets.

ezTLS is designed as a convenient alternative to Native Network Encryption (NNE). It preserves the simple deployment model of NNE while using the TLS protocol for network encryption.

With ezTLS, SQL*Net data is encrypted while it is in transit. You do not need to create or deploy wallets for ezTLS, nor do you need to configure certificates or certificate authorities for ezTLS. You can enable encryption with a small number of configuration

changes. Existing database authentication, such as username and password or Kerberos, continues to be used separately.

View Documentation

Audit

Audit Object Actions at the Column Level for Tables and Views

You can create unified audit policies to audit individual columns in tables and views.

This feature enables you to configure more granular and focused audit policies, and ensures that auditing is selective enough to reduce the creation of unnecessary audit records, and effective enough to let you meet your compliance requirements.

View Documentation

Control Authorizations for Unified Auditing and Traditional Auditing

You can control how privileged users can grant and revoke the Oracle AI Database `AUDIT_ADMIN` and `AUDIT_VIEWER` roles by using Oracle Database Vault APIs. Database Vault blocks direct modification of the database audit tables except through the `DBMS_AUDIT_MGMT` PL/SQL package by authorized users. A new mandatory default realm (Oracle Audit Realm) protects the `AUDSYS` schema and audit-related objects in the `SYS` schema.

This new Database Vault realms simplifies auditing database vault, consolidating the privileges required for auditing into one authorization mechanism. In addition to facilitating the granting of audit-related privileges to the user, this enhancement provides greater separation of duties for managing auditing in an Oracle Database Vault environment.

View Documentation

Authentication

Microsoft Azure Active Directory Integration

You can log into Oracle AI Databases using your Microsoft Azure Active Directory (Azure AD) single sign-on `OAuth2` access token. This feature has been backported to Oracle Database 19c release 19.16 and later, but not for Oracle Database 21c.

New features for Oracle AI Database 26ai include support for Azure AD v2 tokens and retrieving the tokens directly with the Oracle AI Database clients. Use of scripts to retrieve tokens for end-users will not be necessary when using the OAuth2 interactive flow.

This multi-cloud feature integrates authentication and authorization between Azure AD and Oracle AI Databases.

View Documentation

ODP.NET: Azure Active Directory Single Sign-On

ODP.NET can log into Oracle AI Databases using a Microsoft Azure Active Directory (Azure AD) OAuth 2.0 access token. Users can sign-on once with Azure AD, acquire the token, and access their on-premises and cloud-based Oracle AI Databases. This feature is available in ODP.NET Core and managed ODP.NET.

This multicloud capability eases authentication and authorization between Azure AD and Oracle AI Databases by simplifying user access and management.

View Documentation

Increased Oracle AI Database Password Length

Oracle AI Database now supports passwords up to 1024 bytes in length. In previous releases, the password length and the secure role password length could be up to 30 bytes.

Increasing the password length supports an industry-wide trend for stronger authentication. In cases where passwords must be used, the increased length permits passwords that are more difficult to guess.

View Documentation

JDBC-Thin Support for Longer Passwords

Passwords for database user authentication can now be as long as 1024 characters.

This feature fosters increased authentication security for Java applications in Cloud and On-premises environments.

View Documentation

Oracle Data Pump Export and Import Support for Longer Encryption Passwords

Oracle Data Pump can protect export files with encryption passwords of up to 1024 bytes long.

Oracle Data Pump enhances security by supporting encryption passwords of up to 1024 bytes long.

View Documentation

Oracle Call Interface (OCI) and Oracle C++ Call Interface (OCCI) Password Length Increase

Oracle Call Interface (OCI) and Oracle C++ Call Interface (OCCI) now support passwords for database user authentication up to 1024 bytes long.

This feature allows longer passwords to be used to improve security. It also aids database use with tools that generate long passwords.

View Documentation

Updated Kerberos Library and Other Improvements

Oracle AI Database supports MIT Kerberos library version 1.21.2, and provides cross-domain support for accessing resources in other domains.

This Kerberos enhancement improves security and allows Kerberos to be used in more Oracle AI Database environments.

View Documentation

Enhancements to RADIUS Configuration

RADIUS is frequently used to provide multi-factor authentication (MFA) for Oracle AI Database. Oracle AI Database 26ai now supports the RFC 6613 and 6614 guidelines for RADIUS and implements TCP over Transport Layer Security (TLS) by default. This enhancement introduces new RADIUS-related `sqlnet.ora` parameters to support the new standards. The enhancement also deprecates several RADIUS-related `sqlnet.ora` parameters that are no longer needed to support the new standards.

This update to RADIUS standards support improves security for customers using RADIUS-based authentication.

View Documentation

UTL_HTTP Support for SHA-256 and Other Digest Authentication Standards

UTL_HTTP is extended to support both SHA-256 and SHA-512/256 for digest authentication, to ensure forward compatibility.

UTL_HTTP can be seen as an API for client-side HTTP access, much like a standard browser. Support for both SHA-256 and SHA-512/256 for digest authentication enables UTL_HTTP to be at par with other standard browsers.

View Documentation

XDB HTTP SHA256 and SHA512 Digest Authentication

Oracle XDB HTTP protocol server now supports digest authentication SHA256 and SHA512 authentication, which are more secure digest algorithms than MD5.

This feature improves security when using Oracle XDB from the web.

View Documentation

Ability of OCI and Instant Client to Directly Retrieve Microsoft Entra ID (Azure AD) OAuth2 Tokens

Oracle Call Interface (OCI) and Oracle Database Instant Client now can retrieve a Microsoft Entra ID (formerly Azure AD) `OAuth2` token directly from Entra ID instead of relying on a separate script or process to retrieve the token first.

This design improves the interactive flow between the database server and the client when users connect to the database (for example, with SQL*Plus).

This enhancement simplifies the configuration that an end-user must perform in order to retrieve tokens. In previous releases, the end-user had to run a script to get the token from Entra ID before starting SQL*Plus or any other OCI utilities. Now, the token retrieval is part of OCI. This enhancement is similar to recent enhancements with the JDBC-thin and ODP.NET core and managed clients.

View Documentation

Application Identity Logon

Application Identity Logon enables a trusted application or MCP server to connect to Oracle AI Database using its own application identity, while establishing the end-user security context needed for Oracle Deep Data Security enforcement. This allows the database to validate the application, recognize the user or agent operating through it, and apply the appropriate data roles and data grants at runtime.

You can securely support MCP-based and multi-tier application access without relying on broad service-account privileges or creating separate database schemas for every end user. Oracle Deep Data Security enforces access in the database based on the application identity and end-user context, helping ensure each user or agent can access only authorized rows, columns, and values.

View Documentation

Custom Certificates for Oracle FPP Server Authentication

Starting with Oracle Grid Infrastructure 26ai, you can specify custom security certificates for authentication between Oracle Fleet Patching and Provisioning (Oracle FPP) server and clients. By default, Oracle FPP uses SSL/TLS certificates. You can use any security certificate assigned by a Certificate Authority of your choice.

Choosing custom security certificates enables you to meet stringent security policies and regulations set by your organization.

View Documentation

Kerberos Constrained Delegation for DB Links

Release updates 23.26.1 and 19.30 introduce a new `sqlnet.ora` parameter, `SQLNET.KERBEROS5_DELEGATION_MODE`.

Valid values for the new parameter are `CONSTRAINED` and `UNCONSTRAINED`. The default value is `UNCONSTRAINED`.

This new parameter removes the need to set the Windows registry key `AllowTgtSessionKey` to a value that Microsoft considers insecure.

View Documentation

Microsoft Entra ID (Azure AD) Integration Now Supported on AIX, Solaris, and HPUX

The Microsoft Entra ID (previously Azure AD) integration is now available to all Oracle AI Database users regardless of the server operating system platform.

In addition to the newly supported AIX, Solaris, and HPUX platforms, Linux and Windows are still supported. This feature is supported with the Oracle Cloud Infrastructure (OCI) full client and instant clients on Windows and Linux only.

View Documentation

Multi-Factor Authentication for Database Authentication

You can enhance Oracle AI Database security by enabling multi-factor authentication for users. You can use push notifications through the Cisco Duo or Oracle Mobile Authenticator app or combine a username and password with PKI (Public Key Infrastructure) certificate-based authentication.

Multi-factor authentication is a required security requirement for many databases, especially if the database holds sensitive data.

View Documentation

New Parameters to Specify Wallet Certificate and Keys

The `orapki` command line utility now enables you to store alias names and thumbprint signatures in an Oracle wallet.

These enhancements enable users to do the following:

- Specify these private keys using their thumbprint or alias in a connect string.
- Use the thumbprint to specify a private key in the Microsoft Certificate Store (MCS).
- Store certificates with their serial numbers to simplify specifying certificates or removing certificates.

This enhancement affects the `orapki wallet add`, `orapki wallet remove`, and `orapki wallet display` commands. The benefit of this feature is the simplification of managing wallets and selecting certificates through new the thumbprint, alias, and serial number parameters.

The benefit of this feature is the simplification of managing wallets and selecting certificates through new the thumbprint, alias, and serial number parameters.

View Documentation

Setting Multi-Factor Authentication for Local Deep Sec End User

Multi-factor authentication for local Deep Sec end users adds a second authentication factor, such as mobile push approval through DUO or OMA, when locally-managed Deep Sec users sign in. Local end users are managed in the database and can log in directly using password authentication. This enhancement strengthens that login path before Deep Sec establishes the end-user security context and activates the user's data roles.

You can better protect local Deep Sec end-user accounts without moving those users to an external IAM system. MFA helps reduce the risk of password-only access to sensitive data, while Deep Sec continues to enforce least-privilege data access through local data roles and data grants after authentication.

View Documentation

Using OCI IAM DB Token to Authenticate with Deep Data Security

OCI IAM DB token authentication enables OCI users, applications, and services to access Oracle Cloud databases using OCI IAM-issued database tokens. Oracle AI Database validates the token, establishes the user or service identity, and makes that identity available to Deep Sec for data role and data grant enforcement.

You can use OCI IAM as the identity source for Deep Sec without creating separate database users or duplicating authorization logic in each service. This helps OCI services, MCP tools, and cloud-native applications enforce least-privilege data access while preserving identity for authorization and audit.

View Documentation

mkstore Features Included in orapki

`mkstore` features have been incorporated into the `orapki` command line utility to simplify the management of Oracle AI Database wallets, certificates, and secrets.

The new commands in `orapki` support the following capabilities of `mkstore`:

- The ability to create, modify and delete secret store credentials and entries
- The ability to list specific secret store credentials and entries
- The ability to delete a wallet

The capabilities are supported with the `orapki secretstore` command.

The `mkstore` utility has been deprecated. Oracle recommends that you use `orapki` instead.

View Documentation

Authorization

Schema Privileges to Simplify Access Control

Oracle AI Database supports granting privileges on schemas (in addition to the existing object, system, and administrative privileges).

This feature improves security by simplifying authorization for database objects, especially for schemas that frequently add new objects. Instead of granting broad system level (* ANY) privileges that apply to the entire database, privileges can now be granted at the individual schema level.

View Documentation

Oracle Label Security Triggers Are Now Part of the New LBAC_TRIGGER Schema

A new schema, `LBAC_TRIGGER`, is introduced to own the internal triggers that were previously owned by the `LBACSYS` schema. You can migrate existing `LBACSYS` triggers to this new schema.

Both the `LBACSYS` and `LBAC_TRIGGER` schemas are Oracle-maintained and dictionary-protected.

This feature improves security when using the Oracle Label Security option.

View Documentation

Oracle Data Dictionary Protection Extended to Non-SYS Oracle Schemas with Separation of Duties Protection

Oracle AI Database schemas can have data dictionary protection with additional separation of duties protection for `SYSOPER`, `SYSASM`, `SYSBACKUP`, `SYSKM`, `SYSRAC`, and `SYSDG`.

Oracle schemas provide critical functionality for Oracle AI Database features. By enabling these schemas to have data dictionary protection with additional separation of duties, you can prevent inadvertent and malicious changes within these schemas that could endanger Oracle AI Database functionality.

View Documentation

GoldenGate Capture and Apply User Roles

New roles `OGG_CAPTURE`, `OGG_APPLY`, `OGG_APPLY_PROCREP`, `XSTREAM_CAPTURE`, `XSTREAM_APPLY` have been created for granting appropriate capture and apply privileges to the GoldenGate and XStream administrators. These new roles replace the functionality in the procedures of the `DBMS_GOLDENGATE_AUTH` and `DBMS_XSTREAM_AUTH` packages, which are now de-supported.

This feature simplifies administrative tasks.

View Documentation

New Utility Functions for Finding Client Host and IP Information

You can use two new Oracle Database Vault utility functions to find information about client hosts and IPs. These new utility functions are as follows:

- `DBMS_MACUTL.CONTAINS_HOST`
- `DBMS_MACUTL.IS_CLIENT_IP_CONTAINED`

These utility functions enable you to conveniently check if an IP address (or a host) is contained in a domain (or subnet range). They are useful for configuring rules and rule sets.

View Documentation

Ability to Set Tracing Using Oracle Database Vault APIs

You now can use two Oracle Database Vault APIs to control system level tracing, which applies to all database sessions. These new APIs are as follows:

- `DBMS_MACADM.SET_TRACE_LEVEL`
- `DBMS_MACUTL.GET_TRACE_LEVEL`

This enhancement enables users who have been granted the `DV_ADMIN` role to enable or disable tracing for all database sessions. In previous releases, this user needed the `ALTER SYSTEM` and the `ALTER SESSION` system privileges to perform this task, in addition to the `DV_ADMIN` role. The `ALTER SYSTEM` system procedure for tracing is still supported. The enhancement also provides the `DBMS_MACUTL.GET_DV_TRACE_LEVEL` function, which returns the trace level that has been set for the current database session. This trace level can have been set by `ALTER SYSTEM`, `ALTER SESSION`, or `DBMS_MACADM.SET_DV_TRACE_LEVEL`.

View Documentation

Cross-Table Data Grant Support

Cross-Table Data Grant Support enables Deep Sec policies to inherit access across related tables. For example, a user's access to child-table rows such as order items, compensation details, timecards, or fact-table rows can be derived from the user's authorized access to the related parent record, such as an order, employee, or dimension row.

You can secure related business data as a unit without duplicating complex predicates across every child table. This simplifies policy administration, improves consistency across related data, and enables targeted access so users who can view a parent record do not automatically receive access to sensitive child data unless the appropriate privilege is granted.

View Documentation

Fewer Parameters to Specify When Creating or Updating Controls

When configuring Oracle Database Vault, you may now omit parameters in the following cases:

- If you are creating a new control, omitting the parameter specifies its default value.

- If you are updating an existing control, omitting the parameter retains the current setting.

The procedures that are affected are as follows:

- `DBMS_MACADM.CREATE_COMMAND_RULE`
- `DBMS_MACADM.CREATE_CONNECT_COMMAND_RULE`
- `DBMS_MACADM.CREATE_FACTOR`
- `DBMS_MACADM.CREATE_POLICY`
- `DBMS_MACADM.CREATE_REALM`
- `DBMS_MACADM.CREATE_RULE`
- `DBMS_MACADM.CREATE_RULE_SET`
- `DBMS_MACADM.CREATE_SESSION_EVENT_CMD_RULE`
- `DBMS_MACADM.CREATE_SYSTEM_EVENT_CMD_RULE`
- `DBMS_MACADM.UPDATE_COMMAND_RULED`
- `DBMS_MACADM.UPDATE_CONNECT_COMMAND_RULE`
- `DBMS_MACADM.UPDATE_FACTOR`
- `DBMS_MACADM.UPDATE_POLICY_STATE`
- `DBMS_MACADM.UPDATE_REALM`
- `DBMS_MACADM.UPDATE_RULE`
- `DBMS_MACADM.UPDATE_RULE_SET`
- `DBMS_MACADM.UPDATE_SESSION_EVENT_CMD_RULE`
- `DBMS_MACADM.UPDATE_SYSTEM_EVENT_CMD_RULE`

Omitting parameters for default behaviors while creating or updating realms, rules, command rules, factors, and policies streamlines the process, allowing administrators to complete tasks more efficiently and reducing the opportunity for errors.

View Documentation

Oracle Deep Data Security (Deep Sec)

Oracle Deep Data Security (Deep Sec) is a database-enforced data authorization framework in Oracle AI Database. It enforces fine-grained access control at the row, column, and cell levels using declarative SQL, securing all access paths to sensitive data across enterprise applications, analytics tools, and agentic AI systems. Deep Sec integrates natively with external identity and access management systems (like Microsoft Entra ID and OCI IAM) to establish end-user security contexts, which ensures that every SQL operation is authorized against the actual requesting user's identity, roles, and attributes.

As organizations integrate agentic AI into their operations, traditional security models, such as application-level controls, external authorization systems, and conventional database security cannot adequately address the dynamic and autonomous nature of these workloads. Deep Sec addresses this gap by securing data at its source rather

than relying on application code, prompts, or other potentially vulnerable guardrails. By enforcing least-privilege access directly at the database layer, Deep Sec helps organizations adopt AI and modern applications with confidence, while making it easier to address regulatory compliance and data privacy requirements. Its declarative SQL-based policy model simplifies administration, integrates with CI/CD pipelines, and reduces the learning curve compared to procedural or API-driven alternatives.

View Documentation

Autonomous Database

Identity and Access Management Integration with Oracle Autonomous Cloud Databases

You can now log in to additional Oracle Cloud Infrastructure (OCI) DBaaS databases by using an Identity and Access Management (IAM) password or a token-based authentication. It's possible to log in to these databases by using these IAM credentials from tools, such as SQL*Plus or SQLcl.

This feature improves security through centralized management of credentials for OCI DBaaS database instances.

View Documentation

ODP.NET: Oracle Identity and Access Management

ODP.NET supports Oracle Identity and Access Management (IAM) cloud service for unified identity across Oracle cloud services, including Oracle Cloud Database Services. ODP.NET can use the same Oracle IAM credentials for authentication and authorization to the Oracle Cloud and Oracle cloud databases, now with IAM SSO tokens. This feature is available in ODP.NET Core and managed ODP.NET.

This capability allows single sign-on and for identity to be propagated to all services Oracle IAM supports including federated users via Azure Active Directory and Microsoft Active Directory (on-premises). A unified identity makes user management and account management easier for administrators and end users.

View Documentation

Oracle Client Increased Database Password Length

Oracle AI Database and client drivers now support passwords up to 1024 bytes in length.

The Oracle AI Database and client password length has been increased to 1024 bytes, up from 30 bytes, to allow users to set longer passwords if needed. The maximum number of characters is based on the character set used since some characters are larger than one byte.

View Documentation

Other

Secure Distributed Transaction Recovery Background Process (RECO)

Oracle AI Database enables queries and DMLs on objects hosted on a different database. When objects are updated on a remote database, the transaction on the source database ends up becoming a distributed transaction. If a distributed transaction fails, a database background process (RECO) periodically tries to re-establish contact, with the yet-to-be-notified subordinates and pushes the final outcome to those remote databases.

Secure Distributed Transaction Recovery Background Process (RECO) provides additional security for the RECO process.

View Documentation

IP Rate Limit in CMAN

You can use Oracle Connection Manager (CMAN) to limit the number of new connections allowed from an IP address in the specified unit of time. This IP rate limit feature enables you to protect your database against potential denial-of-service (DoS) attacks.

Malicious clients can send excessive connection requests to the server node. This can saturate the capacity of CMAN to handle new connections per second, and thus cause DoS attacks on your database. Using this security feature, you can prevent these types of attacks by detecting such clients early and rejecting those connections.

View Documentation

Enhancements to Oracle Data Redaction

This release includes many enhancements to Oracle Data Redaction, such as the optimization of existing capabilities and the removal of previous limitations.

These enhancements to Oracle Data Redaction improve the overall experience.

View Documentation

OCI Attributes for Microsoft Azure Active Directory Integration with Additional Oracle AI Database Environments

You can log into additional Oracle AI Database environments using your Microsoft Azure Active Directory (Azure AD) single sign-on `OAuth2` access token. The previous release supported Azure AD integration for Oracle Cloud Infrastructure (OCI) Autonomous Database (Shared Infrastructure). This release has expanded Azure AD integration to support on-premises Oracle Database 19c release 19.16 and later. The project adds the OCI attributes needed to supply the bearer token for connection creation.

This multi-cloud feature integrates authentication and authorization between Azure AD and Oracle AI Databases in Oracle Cloud Infrastructure and on-premises.

View Documentation

Verifying Digital Signature and Integrity of Installation Archive Files

In Oracle AI Database 26ai, Oracle digitally signs the installation archive files with Oracle certificates.

Oracle digitally signs the installation archive files to allow customers to ensure the integrity of the packages before deploying them in their environments.

View Documentation

12 OLTP and Core Database

This section describes the new OLTP and core database features.

Availability

True Cache

True Cache is an in-memory, consistent, and automatically managed cache for Oracle AI Database. It operates similarly to an Active Data Guard readers farm, except that True Cache is mostly diskless and is designed for performance and scalability, as opposed to disaster recovery. An application can connect to True Cache directly for read-only workloads. A general read-write Java application can also simply mark some sections of code as read-only, and the 26ai JDBC Thin driver can automatically send read-only workloads to configured True Caches.

Today, many Oracle users place a cache in front of Oracle AI Database to speed up query response time and improve overall scalability. True Cache is a new way to have a cache in front of Oracle AI Database. True Cache has many advantages including ease of use, consistent data, more recent data, and automatically managed cache.

View Documentation

Directory-Based Sharding Method

Directory-based sharding is a type of user-defined sharding in Oracle Globally Distributed Database where the location of data records associated with a sharding key is specified dynamically at the time of insert based on user preferences. The key location information is stored in a directory which can hold a large set of key values in the hundreds of thousands. With directory-based sharding, you have the freedom to move individual key values from one location to another, or make bulk movements to scale up or down, or for data and load balancing.

Directory-based sharding method improves the user-defined sharding model and provides linear scalability, complete fault isolation, and global data distribution for the most demanding applications.

View Documentation

Oracle Globally Distributed Database Raft Replication

Raft replication provides built-in replication for Oracle Globally Distributed Database without requiring configuration of Oracle GoldenGate or Oracle Data Guard. Raft replication is logical replication with consensus-based (RAFT) commit protocol, which enables declarative replication configuration and sub-second failover.

RAFT Replication helps simplify management, improves availability and SLA delivery, as well as optimizes hardware utilization for sharded database environments.

View Documentation

Asynchronous Cross-Region Replication for Globally Distributed Databases Deployed with Raft Replication

This feature enables creation of read-only Raft clusters that are replicas of the primary Read-Write Raft cluster and can be used for disaster recovery and bringing read-only data closer to consumers.

Improves business continuity through cross-region disaster recovery and enables faster access to read-only data by placing replicas closer to users and applications.

View Documentation

Automatic Data Move on Sharding Key Update

When you update the sharding key value on a particular row of a sharded table, the data with that key value might be mapped to a different partition or shard than where it currently resides. Oracle Globally Distributed Database now handles moving the data to the new location, whether it is in a different partition on the same shard or on a different shard.

This features makes data movement between partition or shards seamless when sharding key value update occurs due to various reasons, for example, a move to another country or change in roles.

View Documentation

Automatic Transaction Quarantine

System MONitor (SMON) is a background process responsible for transaction recovery. Transaction Quarantine can now automatically quarantine the recovery of

problematic transactions while keeping the database open, allowing SMON to proceed with the recovery of the other transactions. Alerts and diagnostic information are provided to the DBA or operator so that they can review and resolve the quarantine while other database operations continue unaffected.

The benefit of transaction quarantining is increased fault tolerance and high availability of the database. The database stays up and running and continues processing transactions while the quarantine is being resolved.

View Documentation

Creating Immutable Backups Using RMAN

RMAN is now compatible with immutable OCI Object Storage using locked retention rules, which prevents deletion or modification of backups.

To help organizations meet ransomware protection or strict regulatory requirements for record management and retention, RMAN now prevents anyone, even an administrator, from deleting or modifying backups in OCI Object Storage.

View Documentation

Fine-Grained Refresh Rate Control For Duplicated Tables

Oracle Globally Distributed Database enables refresh rate control for individual duplicated tables. Each duplicated table can have a separate refresh rate which is defined either at its creation or by the `ALTER TABLE` statement.

This feature helps optimize the use of resources by customization of refresh rates for individual duplicated tables.

View Documentation

Global Partitioned Index Support on Subpartitions

Globally Distributed Database allows a global partitioned index on the sharding key when the sharded table is sub-partitioned. You can create a primary key/unique indexes on sharded tables that are composite partitioned without having to include sub-partition keys.

The benefit of this feature is that it removes the restriction on the primary key columns when the sharded table is sub-partitioned, as in the composite sharding method.

View Documentation

JDBC Support for Split Partition Set

This feature enables the Java connection pool (UCP) to receive ONS events about data in a chunk being split and moved across partition sets, and then update the sharding topology appropriately.

This feature furnishes high availability to Java applications using sharded databases.

View Documentation

Managing Flashback Database Logs Outside the Fast Recovery Area

In previous releases, you could store flashback database logs only in the fast recovery area. Now you can optionally designate a separate location for flashback logging. For example, if you have write-intensive database workloads, then flashback database logging can slow down the database if the fast recovery area is not fast enough. In this scenario, you can now choose to write the flashback logs to faster disks. Using a separate destination also eliminates the manual administration to manage the free space in the fast recovery area.

Managing flashback database logs outside the fast recovery area lowers the operational costs related to space management and guarantees the best performance for workloads that are typically impacted by flashback logging on traditional storage.

View Documentation

Move Data Chunks Between Shardspaces

With the Oracle Globally Distributed Database composite sharding method, data is organized into different shardspaces, allowing you to differentiate subsets of data; however, in the past, any automatic chunk moves for load balancing occurred within a shardspace. Now, Oracle Globally Distributed Database lets you move data chunks between existing shardspaces, or move some data to a newly added shardspace.

You can arrange sharded data for your new business needs, such as providing new levels of services or resources for certain customers, or move customers from one class of service to another while maintaining regional data sovereignty.

View Documentation

New Duplicated Table Type - Synchronous Duplicated Table

Oracle Globally Distributed Database introduces a new kind of duplicated table that is synchronized on the shards 'on-commit' on the shard catalog. The rows in a duplicated table on the shards are synchronized with the rows in the duplicated table on the shard catalog when the active transaction performing DMLs on the duplicated tables in the shard catalog is committed.

This features enables efficient and absolute data consistency and synchronization for duplicated tables, across all shards at all times.

View Documentation

New Partition Set Operations for Composite Sharding

For Oracle Globally Distributed Database sharded databases using the composite sharding method, two new `ALTER TABLE` operations enhance partition set maintenance. Previously, partition set operations did not support specifying tablespace sets for child and reference-partitioned tables that are affected due to add and split partition set operations. `MOVE PARTITIONSET` lets you move a whole partition set from one tablespace set to another, within the same shardspace. `MODIFY PARTITIONSET` lets you add values to the list of values of a given partition set.

These new operations enhance resharding capability. `MOVE PARTITIONSET` gives you the control to move all subpartitions of a given table to another tablespace set, within a given shardspace. You can also specify separate tablespace sets for LOBs and subpartitions. `MODIFY PARTITIONSET` extends the add list values feature of partitions to partition sets.

View Documentation

Oracle Data Pump Adds Support for Sharding Metadata

Oracle Data Pump adds support for sharding DDL in the API `dbms_metadata.get_ddl()`. A new transform parameter, `INCLUDE_SHARDING_CLAUSES`, facilitates this support. If this parameter is set to `true`, and the underlying object contains it, then the `get_ddl()` API

returns sharding DDL for create table, sequence, tablespace and tablespace set. To prevent sharding attributes from being set on import, the default value for `INCLUDE_SHARDING_CLAUSES` is set to `false`.

Oracle Data Pump supports sharding migration with support for sharding DDL. You can migrate sharding objects to a target database based on source database shard objects.

View Documentation

Oracle Globally Distributed Database Coordinated Backup and Restore Enhancements

Coordinated backup and restore functionality in Oracle Globally Distributed Database has been extended to include the following:

- Enhanced error handling and diagnosis for backup jobs
- Improved automation of sharded database restore
- Support for running RMAN commands from GDSCTL
- Support for using different RMAN recovery catalogs for different shards
- Encryption of backup sets
- Support for additional backup destinations: Amazon S3, Oracle Object Storage, and ZDLRA

The benefits of this functionality are:

- Easily diagnose problems in backup jobs
- Backups sets can be encrypted so that the data is secured
- Support for additional destinations other than on-disk storage
- Support for different RMAN catalogs and destinations to abide by data residency requirements

View Documentation

PL/SQL Function Cross-Shard Query Support

PL/SQL functions are enhanced with the keyword `SHARD_ENABLE` to allow these functions to be referenced in Oracle Globally Distributed Database cross-shard queries. With the new keyword, the query optimizer takes the initiative to push the execution of the PL/SQL function to the shards.

This feature significantly improves performance for PL/SQL functions in sharded database environments.

View Documentation

Parallel Cross-Shard DML Support

The Oracle Globally Distributed Database query coordinator runs cross-shard updates and inserts in parallel on multiple shards.

This feature improves cross-shard DML performance by running updates and inserts in parallel rather than serially.

View Documentation

Pre-Deployment Diagnostic for Oracle Globally Distributed Database

While processing `GSDSCTL ADD SHARD`, `ADD GSM`, and `DEPLOY` commands, Oracle Globally Distributed Database runs a series of checks to make sure that there is no potential environmental issue.

This feature proactively avoids common pitfalls to reduce time taken to complete a sharded database deployment.

View Documentation

Priority Transactions

If a transaction does not commit or roll back for a long time while holding row locks, it can potentially block other high-priority transactions. This feature allows applications to assign priorities to transactions and for administrators to set timeouts for each priority. The database will automatically roll back a lower priority transaction and release the row locks held if it blocks a higher priority transaction beyond the set timeout, allowing the higher priority transaction to proceed.

Priority Transactions reduces the administrative burden while also helping to maintain transaction latencies and SLAs on higher priority transactions.

View Documentation

RMAN Backup Encryption Algorithm Now Defaults to AES256

RMAN encrypted backups now default to `AES256` encryption algorithm. RMAN will continue to support restore using existing backups created with `AES128` or `AES192` encryption algorithms. You may also choose to create new backups using `AES128` by changing the default `AES256` setting. This default change applies to `BACKUP BACKUPSET` command and the `ALLOCATE CHANNEL` command.

To strengthen the security of encrypted backups from being decrypted by malicious users, RMAN encrypted backups now default to the `AES256` encryption standard.

View Documentation

RMAN Operational, Diagnostics, and Upgrade Enhancements

RMAN now includes easier standby database registration for Oracle Data Guard, better fault tolerance and optimization for Oracle Real Application Cluster (Oracle RAC), enhanced diagnosability which automatically gathers information to help identify issues, and updates to mitigate bottlenecks and pause sessions during recovery catalog upgrades.

RMAN operations are now easier and more resilient for highly available Oracle environments with less complex backup registration, automatic diagnostic gathering, and fewer failures when performing maintenance activities.

View Documentation

Simplified Database Migration Across Platforms Using RMAN

Using RMAN to migrate databases across different operating system platforms has been streamlined and includes support for databases encrypted with Transparent Data Encryption (TDE) and multi-section backups. New command options allow existing RMAN backups to be used to transport tablespaces or pluggable databases to a new destination database with minimal downtime.

Migrations using RMAN are now easier, faster, and require fewer steps to execute. The new capabilities enable a simple and straightforward migration process, minimizing downtime for your applications, reducing risk, and increasing productivity.

View Documentation

Support for Oracle AI Database Version Specific RMAN SBT Library

The Oracle Home directory now includes the database version compatible libraries (`SBT_LIBRARY`) for Zero Data Loss Recovery Appliance, OCI Object Storage and Amazon S3. You can now configure RMAN to directly access libraries from the Oracle Home directory using an alias. For example, if the backup destination is OCI Object Storage, you only have to specify the alias name `oracle.oci` for the `SBT_LIBRARY` parameter. When RMAN attempts to backup to Object Storage, it uses the specified alias to access the SBT library used for backup cloud service from the Oracle home directory.

The RMAN storage libraries are now included with the database, eliminating the need to download and install additional software and ensuring that you have all the necessary components to immediately start backing up and restoring from Zero Data Loss Recovery Appliance, OCI Object Storage, or Amazon S3.

View Documentation

Witness Replica

In Raft-based distributed systems, the witness replica serves as a lightweight, non-voting node that does not maintain a full database copy. It receives and applies Raft logs to track consensus state changes while participating in quorum voting for leader elections and transaction commits.

This enables efficient fault tolerance and scalability without the storage overhead of data replication, making it ideal for high-volume environments. It can enable up to 33% storage savings for deployments.

View Documentation

Blockchain

Blockchain Table User Chains

Earlier versions of blockchain tables supported only system chains. A system chain (one of the 32 chains per instance) is randomly chosen by Oracle for every new row inserted into a blockchain table.

A user chain is a chain of rows based on a set of up to three user-defined columns of type `NUMBER`, `CHAR`, `VARCHAR2`, and `RAW`. For example, consider a blockchain table created for tracking banking transactions (withdrawals, deposits, transfers) associated with various accounts. Assume there is a column called `ACCOUNTNO` in the blockchain table

for account numbers. Each transaction inserts a new entry into this blockchain table for some account number. A user chain can be associated with every unique value in `ACCOUNTNO`. If there are a total of 100 different account numbers, there can be at most 100 user chains. You can then run a verification procedure only on a chain for a specific `ACCOUNTNO`, providing greater data isolation. This feature allows you to create user chains for rows in blockchain tables based on version columns even if they are split across system chains.

Multiple user chains increase the flexibility of applying blockchain tables and their verification procedures to make it easier to leverage tamper-resistant tables in your applications.

View Documentation

Blockchain Table Row Versions

The blockchain table row version feature allows you to have multiple historical versions of a row that is maintained within a blockchain table corresponding to a set of user-defined columns. A view `<bctable>_last$` on top of the blockchain table allows you to see just the latest version of a row.

This feature allows you to guarantee row versioning when using tamper-resistant blockchain tables in your application.

View Documentation

Blockchain Table Log History

Flashback Data Archive History tables are now blockchain tables. This feature allows changes to one or more regular user tables to be tracked in a blockchain table maintained by Oracle AI Database as part of the Flashback Data Archive. Each change in a regular table will be added to the blockchain log history table as a separate row within a cryptographic hash chain maintained by the blockchain table. You can verify the data and chain integrity in a Flashback Data Archive Blockchain Log History table using the built-in verification procedures (`DBMS_BLOCKCHAIN_TABLE.verify_rows`) or through an external verification, including a continuous verification process illustrated by a sample provided in `https://github.com/oracle/blockchain-table-samples`.

This feature allows you to record changes to regular user tables in a cryptographically secure and verifiable fashion.

View Documentation

Add and Drop User Columns in Blockchain and Immutable Tables

This feature allows evolution of Blockchain and Immutable Tables, namely it allows columns to be added and dropped while maintaining the current data, including that in dropped columns for continuity of crypto-hash chains.

As applications evolve you may need to modify existing tables by adding or dropping columns. In this release, you can easily add or drop columns in previously created Blockchain or Immutable tables. Any rows prior to a column deletion will maintain the data in these columns in order to preserve the integrity of the crypto-hash chains and allow the verification procedures to work across the entire table.

View Documentation

Blockchain Table Countersignature

You can request a database countersignature at the time of signing a row. In addition to recording the countersignature and its metadata in the row, the countersignature and the `signed_bytes` are returned to the caller. The caller can then save the countersignature and `signed_bytes` in another data store, such as Oracle Blockchain Platform, for non-repudiation purposes.

A countersignature can provide user additional guarantees that data has been securely stored in the blockchain table.

View Documentation

Blockchain Table Delegate Signer

A delegate is an alternate user who's allowed to sign rows inserted by the primary user. This feature allows a delegate to sign rows in an immutable or blockchain table on behalf of another user. A delegate's signature is accepted only if the signature can be verified using the public key in the delegate's certificate, which has been added to the dictionary table.

A delegate signer can be used when users are not able to sign the rows they created and they trust their delegate.

View Documentation

In-Place Conversion to Immutable Tables and Parallel CTAS Support

The following enhancements have been introduced for blockchain and immutable tables:

- **In-place conversion to immutable tables:** Adds DDL support to convert an existing table into an immutable table.
- **Parallel CTAS support:** Enables parallel execution for creating blockchain tables using the `CREATE TABLE AS SELECT` (CTAS) operation.

These enhancements to blockchain and immutable tables improve both usability and performance during their creation.

View Documentation

New Special Privilege Required to Set Long Idle Retention Times for Blockchain and Immutable Tables

Blockchain or immutable tables with idle retention set to a sufficiently large value cannot be dropped until the newest row of the table becomes very old. This limits the ability to drop the blockchain/immutable table if necessary to prevent a disk space exhaustion attack. Hence, the operation of setting a table's idle retention to a large value is restricted to privileged users via a grant of a new `TABLE RETENTION` system privilege. The idle retention threshold, which specifies when to require the new privilege `BLOCKCHAIN_TABLE_RETENTION_THRESHOLD`, is configurable.

Ability to create blockchain or immutable tables with long retention times and inserting large amounts of data that can not be deleted could potentially be a vector for a denial of service attack via disk space exhaustion. To reduce this risk, the special privilege has been introduced. Only users granted this privilege can set idle retention above the configurable threshold level.

View Documentation

Parallel Verification of Blockchain Tables

There is now a new PL/SQL stored procedure that performs parallel verification of blockchain tables.

This feature significantly speeds up verification of data integrity (detecting deletion or tampering) by parallelizing the process through a new PL/SQL stored procedure, allowing more frequent verifications with less impact on other database workloads.

View Documentation

Database Architecture

Lock-Free Reservation

The Lock-Free Reservation feature enables concurrent transactions to proceed without being blocked on updates of heavily updated rows. A Lock-Free Reservation is held on the row, instead of locking the row. The Lock-Free Reservation verifies if the updates can succeed and defers the updates until the transaction commit time.

Lock-Free Reservation improves the end user experience, and concurrency, in transactions.

View Documentation

Wide Tables

The maximum number of columns allowed in a database table or view has been increased to 4096. This feature allows you to build applications that can store attributes in a single table with more than the previous 1000-column limit. Some applications, such as Machine Learning and streaming IoT application workloads, may require the use of de-normalized tables with more than 1000 columns.

You now have the ability to store a larger number of attributes in a single row which for some applications may simplify application design and implementation.

View Documentation

Assertions, a Powerful New Constraint Type to Simplify Data Integrity Enforcement

An assertion allows the user to define business rules on one or more tables so that DMLs on the table(s) will always honor those rules. Current solutions, such as check constraints and triggers, have limits. Check constraints only work at the column or row level. Triggers, while table-level, need complex PL/SQL and careful management for multi-table cases. With assertions, developers use a clear, declarative syntax to embed business logic directly in the database.

This feature reduces the risk of inconsistent data and eliminates redundant validation code across applications. Assertions also resolve concurrency and serialization issues

by default. When business rules change, updating a single assertion is faster and less error-prone than editing many codebases.

View Documentation

Consolidated Service Backgrounds for Oracle Instance

We are introducing a new set of service processes which execute database service actions.

Service actions are responsible for maintenance tasks, parallel tasks and brokered tasks, consolidated tasks and many more. These were performed by dedicated processes in the database before. The new background scheduler group processes can execute any of these service actions, thus providing consolidation of background service actions.

View Documentation

Improve Performance and Disk Utilization for Hybrid Columnar Compression

Enhancements to the compression algorithms for Hybrid Columnar Compression (HCC) include improvements for faster compression and decompression speeds, as well as better compression ratios for newly created HCC compressed tables or for existing HCC compressed tables that are rebuilt. The exact benefits can vary based on the data and the chosen compression level.

This feature improves an application's workload performance while reducing database storage utilization.

View Documentation

Migrating LOBs Made Easier

The number of steps required to migrate and compress BasicFile LOB segments to SecureFiles LOB segments, using SecureFiles Migration Utility, have been reduced.

This enhancement simplifies the process to migrate LOB segments.

View Documentation

Optimized Oracle Native Access to NVMe Devices Over Fabric

Starting with Oracle AI Database 26ai, you can use TCP/IP network connections to access the remote NVMe storage devices using NVMe over Fabrics (NVMe-oF). The Oracle Grid Infrastructure server works as an initiator that connects to an NVMe-oF storage target created using Linux Kernel nvmet_tcp module, providing optimized user mode access to remote NVMe devices.

NVMe-oF provides a low-latency and secure way to access remote NVMe devices exported using NVMe Over Fabrics target. Oracle provides an optimized way to access these NVMe-oF devices directly from an Oracle process. This Oracle-native method of accessing NVMe-oF devices reduces latency while Oracle ASM makes storage manageability easier.

View Documentation

Oracle DBCA Support for PMEM Storage

Oracle Database Configuration Assistant (Oracle DBCA) enables you to select persistent memory database (PMEM) as your storage option when creating a single-instance database.

This feature automates the process of assigning a PMEM device for your database storage enabling you to place database files in a PMEM storage device.

View Documentation

Partition by Expression

Partition by expression allows you to define a partition key using an expression directly in the partitioning clause. This eliminates the need for creating a separate virtual column solely to serve as the partition key, streamlining schema design.

This feature simplifies DDL and maintenance by removing the overhead of managing virtual columns, while making partitioning definitions more intuitive and expressive. It gives developers more flexibility to partition on derived values (dates, substrings, JSON fields, etc.) with less schema clutter.

View Documentation

System Timezone Autonomy for Pluggable Databases

Oracle Multitenant enables Oracle AI Database to consolidate multiple pluggable databases as self-contained databases, improving resource utilization and database management. In addition to providing a fully centrally managed database environment with identical, global time zone settings for all pluggable databases (impacting `SYSDATE` and `SYSTIMESTAMP`), pluggable databases can now control their time zone settings independently. You can control the time zone setting, including internal processes and operations, or only on a user-visible level.

The ability to control the time zone behavior for `SYSDATE` and `SYSTIMESTAMP` on a pluggable database level increases to self-containment of individual databases in a multitenant environment and enhances your consolidation capabilities of independent databases.

View Documentation

Unrestricted Direct Loads

Prior to this feature, after a direct load and prior to a commit, queries and additional DMLs were not allowed on the same table for the same session or for other database sessions. This enhancement allows the loading session to query and perform DML on the same table that was loaded. Rollback to a savepoint is also supported.

This feature removes the restrictions that you may have encountered when loading and querying data. Potentially improving the performance of your applications in areas such as Data Warehousing and complex batch processing.

View Documentation

General

Unrestricted Bulk Transactions

Oracle AI Database allows DML statements (`INSERT`, `UPDATE`, `DELETE`, and `MERGE`) to be executed in parallel by breaking the DML statements into mutually exclusive smaller tasks. Executing DML statements in parallel can make DSS queries, batched OLTP jobs, or any larger DML operations faster. However, parallel DML operations had a few transactional limitations.

This includes a limitation that restricted transactions with multiple per-table parallel DMLs. This means that once an object is modified by a parallel DML statement,

that object cannot be read or modified by later statements of the same transaction. This enhancement removes this limitation, enabling users to run parallel DMLs, and any combination of statements like queries, serial DML, and parallel DML on the same object, within the same transaction.

For users, this simplifies and speeds up data loading and analytical processing by making full use of Oracle AI Database's parallel execution and parallel query capabilities.

View Documentation

ACFS Auto Resize Variable Threshold

ACFS auto resize now allows you to configure the threshold percentage for your file system automatic resize.

A more flexible threshold is now available for your file systems auto resize. Previously, the threshold was fixed to 10%. Now, you can customize to your specific use case needs.

View Documentation

ACFS Cross Version Replication

ACFS replication now allows for primary clusters to replicate to standby cluster on a previous or older release.

This feature will provide flexibility in replication configurations, providing ample time for upgrading and lifecycle maintenance.

View Documentation

ACFS Encryption Migration from OCR to OKV

ACFS Encryption now allows you to migrate from OCR to OKV.

This feature allows for a centralized point for key management using Oracle Key Vault.

View Documentation

ACFS Replication Password-less SSH Setup Tool

A new tool provides users the ability to configure SSH keys management for ACFS Replication.

Users can now avoid the repetitive, error-prone process of SSH keys management, setup, and configuration with this new tool. The tool makes the ACFS replication setup process more efficient and easier.

View Documentation

ACFS Replication Switchover

A new command, `acfsutil repl switchover`, provides a coordinated failover. However, if ACFS cannot establish contact the replication primary site, the command will fail.

Enhanced flexibility in ACFS replication management is now available with the addition of this new command.

View Documentation

ACFS SSH-less Replication

This feature provides an alternative transport choice for ACFS Replication which eliminates the need to maintain ssh-related host and user keys.

Users now have an alternative to ssh, including network data transfer, authentication between replication storage locations, encryption of the data stream, and a facility for executing remote commands.

View Documentation

ACFS Snapshots RMAN Sparse Backup and Restore

You can now back up and restore PDB snapshot copies on ACFS.

Backing up and restoring PDB snapshot copies on ACFS, provides the space-efficient storage that is inherent of ACFS Snapshots.

View Documentation

ACFS Sparse Backup and Restore of Snapshots

The `acfsutil snap duplicate` command can now generate a backup of an entire ACFS file systems and its snapshots, while preserving its sparseness.

You can now apply a full backup to another location while retaining the original sparseness. You can now replicate an entire ACFS file system and its snapshot tree with this new functionality.

View Documentation

ACFSutil plogconfig Log Files Wrapping Info

ACFSutil plogconfig offers you a way to manage persistent logging configuration settings. `acfsutil plogconfig -q` will now offer you additional information on whether the logs have wrapped or not. You can also get this information with `acfsutil plogconfig -w`, which will offer only this information and not all the comprehensive information offered by `acfsutil plogconfig -q`.

Further information regarding persistent logging is now available, hence enhancing the experience in the realm of diagnosability.

View Documentation

Automatic Parallel Direct Path Load Using SQL*Loader

The SQL*Loader client can automatically start a parallel direct path load for data without dividing the data into separate files and starting multiple SQL*Loader clients. This feature prevents fragmentation into many small data extents. The data doesn't need to be resident on the database server. Cloud users can employ this feature to load data in parallel without having to move data on to the cloud system if there is sufficient network bandwidth.

SQL*Loader can load data faster and easier into Oracle AI Database with automatic parallelism and more efficient data storage.

View Documentation

BIGFILE Default for SYSAUX, SYSTEM, and USER Tablespaces

Starting with Oracle AI Database 26ai, BIGFILE functionality is the default for `SYSAUX`, `SYSTEM`, and `USER` tablespaces.

A bigfile tablespace is a tablespace with a single, but large datafile. Traditional small file tablespaces, in contrast, typically contain multiple datafiles, but the files cannot be as large. Making `SYSUAX`, `SYSTEM` and `USER` tablespaces bigfile by default will benefit large databases by reducing the number of datafiles, thereby simplifying datafile, tablespace and overall global database management for users.

View Documentation

Bigfile Tablespace Shrink

This feature supplies the capability to reliably shrink a bigfile tablespace.

In earlier releases, organizations may have found that the datafile of a bigfile tablespace grew larger despite the actual used space being much smaller. This could happen after a user dropped segments/objects in the tablespace, but was not able to use datafile resize to recover the freed space due to the location of the data in the datafile.

By using Bigfile Tablespace Shrink, organizations can now reliably shrink a bigfile tablespace to close to the sum of the size of all objects in that tablespace, optimizing storage and reducing costs.

View Documentation

CEIL and FLOOR for DATE, TIMESTAMP, and INTERVAL Data Types

You can now pass `DATE`, `TIMESTAMP`, and `INTERVAL` values to the `CEIL` and `FLOOR` functions. These functions include an optional second argument to specify a rounding unit. You can also pass `INTERVAL` values to `ROUND` and `TRUNC` functions.

These functions make it easy to find the upper and lower bounds for date and time values for a specified unit.

View Documentation

CONSTRAINT_NOVALIDATE, a Data Pump Import TRANSFORM Parameter

The Oracle Data Pump `TRANSFORM` parameter option `CONSTRAINT_NOVALIDATE` enables you to set validation preferences. It has two options: `Y` or `N`. When set to `Y`, constraints are not validated during import.

Validating constraints during import that were valid on the source can be unnecessary and slow the migration process. Validation can be done after import.

View Documentation

Centralized Configuration Providers

Database clients can securely pull application configuration data from Azure or OCI Cloud. The store can contain data such as application connection descriptors and tuning parameters.

Central configuration makes application management and scaling easier. It fits well with architectures such as microservices and serverless deployments.

View Documentation

EBCDIC Binary Collation Support

The new EBCDIC binary collations can be used to emulate the binary ordering of IBM EBCDIC code pages natively within the Oracle Database. The collation key sizes are optimized to be comparable to the character data being compared or sorted.

The new EBCDIC binary collations help customers migrating mainframe systems to Oracle Database in maintaining the dependency on EBCDIC collation sequences with minimal performance impact.

View Documentation

IBM-Compatible EBCDIC Client Character Set Support

The IBM-compatible client character sets can be used to achieve character mapping behavior between EBCDIC and ASCII based on IBM definitions when migrating from mainframe systems to Oracle Database.

The new IBM-compatible character sets help preserve data integrity when migrating from mainframe systems to Oracle Database. They can also be used with the EBCDIC binary collation feature to maintain the EBCDIC binary collation behavior in Oracle Database during mainframe migration.

View Documentation

LONG_TO_LOB, a Data Pump import TRANSFORM parameter

The Oracle Data Pump `TRANSFORM` parameter now includes the `LONG_TO_LOB` option to migrate all `LONG` and `LONG RAW` data types.

Using this transform enables you to migrate the deprecated `LONG` and `LONG RAW` data types by transparently and automatically converting them to `CLOB` and `BLOB` data types. This parameter changes all `LONG` data types to `CLOB` and all `LONG RAW` data types to `BLOB`.

The syntax is `TRANSFORM = LONG_TO_LOB:[Y|N]`. The default is `N`.

View Documentation

Materialized Expression Columns

Expression columns, also known as virtual columns, are additional columns derived (computed) from existing columns. They can be persisted (materialized) on disk, complementing the existing default functionality of computing the results at runtime only.

Choosing between computation at runtime and computation at DML time for expression columns provides more flexibility in choosing the right approach for an application.

Materializing expression columns trades disk storage for the need to compute the same expression over and over again.

View Documentation

Oracle Data Pump Filters GoldenGate ACDR Columns from Tables

The ACDR feature of Oracle GoldenGate adds hidden columns to tables to resolve conflicts when the same row is updated by different databases using active replication. GoldenGate can also create a "tombstone table," which records interesting column values for deleted rows. Oracle Data Pump can exclude the hidden columns and the tombstone tables by setting a new import transform parameter, `OMIT_ACDR_METADATA`.

Oracle Data Pump enhances migration flexibility. It can migrate data from an Oracle GoldenGate ACDR (automatic conflict detection and resolution) environment to a non-ACDR environment by excluding the GoldenGate ACDR metadata during import.

View Documentation

PDB Snapshot Carousel ACFS Support

Oracle ACFS now supports PDB Snapshot Carousel, which allows you to maintain a library of PDB Snapshots.

Oracle AI Database files stored on Oracle ACFS file systems can now leverage PDB Snapshot Carousel in conjunction with ACFS snapshot technology.

View Documentation

SQL*Loader Supports SODA (Simple Oracle Document Access)

SQL*Loader now supports Simple Oracle Document Access (SODA). You can insert, append, and replace external documents into SODA collections in Oracle AI Database applications by using the SQL*Loader utility in both control file and express modes.

SQL*Loader support for Simple Oracle Document Access (SODA) makes it easier and faster to load schema-less JSON or XML-based application data into Oracle AI Database.

View Documentation

Sessionless Transactions

Managing a transaction requires the connection and session resources to be tied to the transaction throughout its lifecycle. Therefore, the session or connection can be released only after the transaction has ended. This often results in underutilization of sessions/connections. In Sessionless Transactions, after you start a transaction, you have the flexibility to suspend and resume the transaction during its lifecycle. The session or connection can be released back to the pool and can be reused by other transactions, therefore effectively being able to multiplex transactions and sessions/connections.

Sessionless Transactions provide ability for applications to suspend and resume transactions across sessions/connections (single instance or RAC) without the need for an external transaction manager, and without the application having to coordinate the commit and recovery protocols. The database manages transaction lifecycle, including commit and recovery. Application performance, and throughput, benefit from reduced commit latency since fewer client-server roundtrips are needed. Since external coordination is not required, using Sessionless Transactions results in vastly

simplified mid-tier or app-tier infrastructure, and significantly decreases downtimes when compared with externally coordinating transactions (such as with XA).

View Documentation

Unicode 17.0 Support

The National Language Support (NLS) data files for `AL32UTF8` and `AL16UTF16` character sets are updated to match version 17.0 of the Unicode Standard character database.

This enhancement enables Oracle AI Database to conform to the latest version of the Unicode Standard.

View Documentation

XMLTYPE_STORAGE_CLAUSE, a Data Pump Import TRANSFORM Parameter

Transportable Binary XML simplifies the XML data storage and makes it easier to transport. It does not store the metadata used to encode or decode XML data in a central table.

`XMLTYPE_STORAGE_CLAUSE` now takes the options `TRANSPORTABLE BINARY XML` or `BINARY XML`. Oracle recommends that you use the `TRANSPORTABLE BINARY XML` XMLType, the new and recommended storage type for Release 26ai to store data in a self-contained binary format.

Use the `BINARY XML` (Non-Transportable) storage XMLType to store data in a post-parse, binary format designed specifically for XML data. Binary XML is compact, post-parse, XML schema-aware XML data. Binary XML is non-transportable and stores the metadata used to encode or decode XML data efficiently in a central table.

Data Pump can export and import data of type XMLType regardless of the source database XMLType storage format (object-relational, binary XML or CLOB). Oracle Data Pump exports and imports XML data as Transportable Binary XML so the the source and target databases can use different XMLType storage models for the data.

View Documentation

Manageability and Performance

Advanced LOW IOT Compression

An index-organized table (IOT) is a table stored in a variation of a B-tree index structure where rows are ordered by primary key. IOTs are useful because they provide fast random access by primary key without duplicating primary key columns in two structures – a heap table and an index. In earlier releases, IOTs only supported Oracle's prefix compression (formerly called key compression), which required additional analysis and had the possibility of negative compression (where the overhead of compression outweighed the compression benefits).

Advanced LOW IOT Compression allows you to reduce the overall storage for Oracle AI Databases.

View Documentation

Automatic SecureFiles Shrink

Automatic SecureFiles Shrink selects SecureFiles LOB segments based on a set of criteria and executes the free space shrink operation in the background for the selected segments. With Automatic SecureFiles Shrink, the shrink operation happens transparently in small and gradual steps over time while allowing DDL and DML statements to execute concurrently. In the manual method, you must decide on which LOB segments to shrink using tools like Segment Advisor and use a DDL statement to execute the shrink operation. The manual method may not be feasible for very large LOB segments because it is time-consuming.

Automatic SecureFiles Shrink simplifies administrator duties and saves time due to the automation of this process.

View Documentation

Automatic Storage Compression

Organizations use Hybrid Columnar Compression for space saving and fast analytics performance. However, the compression and decompression overhead of Hybrid Columnar Compression can affect direct load performance. To improve direct load performance, Automatic Storage Compression enables Oracle AI Database to direct load data into an uncompressed format initially, and then gradually move rows into Hybrid Columnar Compression format in the background.

Automatic Storage Compression improves direct load performance, while keeping the advantages of Hybrid Columnar Compression, including space savings and fast analytics performance.

View Documentation

DBCA Silent Options Changes

DBCA silent mode options changes in various functionality

DBCA silent command line options integrate smoothly with custom scripts and provide user-friendly errors.

View Documentation

Enhanced Query History Tracking and Reporting

Enhanced Query History Tracking and Reporting lets you track and report on a more complete history of user-issued queries than is available in previous releases. This feature provides you with greater capability to track user-initiated queries within a session. It includes non-parallel queries with less than five seconds of execution time, which are not tracked with Real-time SQL Monitoring unless tracking is forced by a hint. Each user can access and report on their own current session history. SYS users and DBAs can view and get query history reports for all current user sessions and can also turn this functionality on or off. Reporting is configurable, with options for selecting the reporting scope and detail level.

Enhanced Query History Tracking and Reporting allows application developers and development operations (DevOps) personas to get detailed insight into the queries that execute on your databases. This insight allows you to better manage and optimize your applications.

View Documentation

Fast Ingest (Memoptimize for Write) Enhancements

This feature adds enhancements to Memoptimize Rowstore Fast Ingest with support for partitioning, compressed tables, fast flush using direct writes, and direct In-Memory column store population support. These enhancements make the Fast Ingest feature easier to incorporate in more situations where fast data ingest is required.

This feature helps Oracle AI Database provide better support for applications requiring fast data ingest capabilities. Data can be ingested and then processed all in the same database. This reduces the need for special loading environments, and thus reduces complexity and data redundancy.

View Documentation

Improved Performance of LOB Writes

You can experience improved read and write performance for LOBs due to the following enhancements:

- Multiple LOBs in a single transaction are buffered simultaneously. This improves performance when you use switch between LOBs while writing within a single transaction.
- Various enhancements, such as acceleration of compressed LOB append and compression unit caching, improve the performance of reads and writes to compressed LOBs.
- The input-output buffer is resized based on the input data for large writes to LOBs with the NOCACHE option. This improves the performance for large direct writes, such as writes to file systems on DBFS and OFS.

This feature adds a host of improvements to accelerate SecureFiles writes for JSON document-based applications, for write calls issued by a database file system, and also for LOB workloads where the underlying data is compressed for storage savings.

View Documentation

Improved System Monitor (SMON) Process Scalability

Queries can require large amounts of temporary space and some temporary space operations run in critical background processes, like the System Monitor (SMON) process. SMON is responsible for cleaning up temporary segments that are no longer in use. SMON checks regularly to see whether it is needed, and other processes can call SMON. Temporary space management can affect SMON's scalability for other critical actions. This new enhancement instead uses the Space Management Coordinator (SMCO) process so that the responsibility of managing temporary space is offloaded from SMON, thereby improving its scalability.

This feature improves the overall scalability of the SMON process, particularly in a multitenant Oracle RAC cluster.

View Documentation

Log-Buffer-Space-Based Auto Redo Prioritization

Oracle AI Database automatically prioritizes redo generation for sessions when log buffer space becomes constrained on a primary database. Redo-intensive batch or maintenance sessions are assigned a lower redo-generation priority, while OLTP sessions continue to receive normal responsiveness and transactional throughput. The prioritization is based on session characteristics, such as redo volume and call duration, enabling the database to favor workloads that are more latency-sensitive.

Log buffer space remains available for ongoing transaction processing, helping prevent redo pressure from disrupting database operations. By giving batch-style redo producers a lower priority and preserving OLTP responsiveness, the database maintains higher availability, protects transactional throughput, and reduces the risk of buffer space exhaustion during periods of heavy redo generation.

View Documentation

Memory Speed Columnar

The Memory Speed Columnar enhancement is the next generation of Hybrid Columnar Compression. The existing Hybrid Columnar Compression (HCC) uses an on-disk data format that is best suited for analytical workloads but does not support in-memory columnar formats. Memory Speed Columnar introduces a new Compression Unit (CU) data format that enables Memory Speed Columnar scans in all storage tiers, and eliminates the columnar format mismatch between HCC, Exadata flash columnar formats and the Database In-Memory format. The Memory Speed Columnar enhancement is an optional extension to the existing Hybrid Columnar Compression syntax and can coexist with existing HCC objects.

The Memory Speed Columnar enhancement makes on-disk scans more efficient. It does this by leveraging in-memory formats and enabling the creation of larger on-disk compression units (CUs), which improves processing efficiency, such as SIMD (Single Instruction/Multiple Data) operations. The new format eliminates the mismatch between HCC, Exadata flash columnar, and Database In-Memory columnar formats, minimizing transformations. As a result, analytic workloads typically benefit from reduced CPU usage, improved query performance, and storage efficiency comparable to existing Hybrid Columnar Compression capabilities.

View Documentation

Migrate BasicFile LOBs Using the SecureFiles Migration Utility

You can use the SecureFiles Migration Utility to simplify the migration and compression of BasicFile LOB segments to SecureFiles LOB segments.

Earlier it was challenging to decide which BasicFile LOBs to migrate to SecureFile LOBs, and whether or not to compress the LOBs, especially considering that organizations often have many databases, with a large numbers of schemas, tables, and segments. SecureFiles Migration Utility automates several steps that were earlier performed manually. It also generates several reports that help you decide which BasicFile LOBs you want to migrate and compress.

View Documentation

Ordered Sequence Optimizations in Oracle RAC

The processing of ordered sequences in Oracle Real Application Clusters (Oracle RAC) has been optimized to provide better performance without requiring manual changes, ensuring a strict sequence order.

Applications using ordered sequences in Oracle RAC environments will benefit from improved performance and scalability.

View Documentation

PDB Target Redo Generation Management

The `REDO_GENERATION_KBPS_MAX` initialization parameter specifies the target maximum amount of redo, in kilobytes, that a PDB can generate for each second.

A large amount of redo generation can cause poor performance. If one PDB is generating redo at a high rate, then it can affect the performance of other PDBs in the same CDB. Use the `REDO_GENERATION_KBPS_MAX` initialization parameter to limit the redo generated by a specific PDB.

Set the value based on the workload and operational requirements for the PDB. When the redo generation rate for a PDB exceeds the target, Oracle Database can delay sessions in the PDB to keep the redo generation rate close to the target. The actual redo generation rate can be lower than the target.

View Documentation

Parallel Index Creation Parameters for Data Pump Import

The Oracle Data Pump Import command-line mode `ONESTEP_INDEX` parameter optimizes index creation concurrency and balances it with overall job parallelism. It does this in conjunction with the `INDEX_THRESHOLD` parameter.

Index creation is an essential step in the migration process. Enabling parallelism for large index creation and balancing the needs of large index creation with overrall import job parallelism makes more efficient use of parallel resources assigned to a Data Pump import job. It makes the logical migration process more effective by substantially reducing the time to import. Finally, it removes the effort associated with the extra step of extracting the large index definitions from the source dumpfiles and creating these large indexes outside the import job.

View Documentation

Pluggable Database Support in Oracle Data Guard Environments

There is now a pluggable database configuration in a Data Guard environment using Database Configuration Assistant (DBCA).

A command line based silent mode option is available for configuring pluggable databases (PDBs) in Data Guard environments.

View Documentation

Refreshable PDBs in DBCA

Database Configuration Assistant (DBCA) allows you to clone a remote Pluggable database (PDB) as a refreshable PDB. When a PDB is created as refreshable, the changes of the source PDB will periodically propagate to the refreshable PDB. The refreshable PDB can be configured to refresh manually or automatically during creation.

A DBCA-based graphical user interface or scripted silent mode for cloning a remote refreshable PDB reduces many commands needed to create a remote refreshable PDB clone ensuring a faster and more reliable cloning of PDBs.

View Documentation

Runtime Staleness Detection and Fast Refresh for Materialized Views Containing External Tables

Fast refresh and query rewrite is now supported for materialized views defined on external tables, such as those sourcing data from cloud object stores.

This feature enables fast refresh for materialized views that reference external tables, such as those sourcing data from object stores. By avoiding full refreshes and instead refreshing only changed data, it significantly reduces resource usage and refresh time. This makes it easier and more efficient to work with large or frequent updated external datasets in cloud-based data lake architectures.

View Documentation

Smallfile Tablespace Shrink

This feature supplies the capability to reliably shrink a smallfile tablespace.

In earlier releases, organizations may find the datafiles of a smallfile tablespace grow larger despite the actual used space being much smaller. This can happen after a user drops segments or objects in the tablespace, but depending on where data was located in the datafiles, users were not always able to use datafile resize to recover the freed space.

By using Smallfile Tablespace Shrink, you can now expect your smallfile tablespace size to be close to the sum of the size of all segments,and objects in that tablespace. You can now reliably shrink a smallfile tablespace, which means improved storage optimization and reduced storage costs.

View Documentation

Smart Connection Rebalance Recommendation

The SMART Recommendation Engine analyzes the activity of each database service to assess whether enabling the Smart Connection Rebalance feature at the service level will improve database performance. For each service, the engine provides actionable recommendations, which are accessible in both the Automatic Workload Repository (AWR) report and the `(G)V$ACTIVE_SERVICES` view.

By following these recommendations, you can enhance overall performance, including improvements in throughput and response times for your database services.

View Documentation

WAIT and NOWAIT Support for All DML Statements

Rows modified by DMLs are locked and if an attempt is made to modify the same row by another session, then that session must wait for the row to become available. The `SELECT FOR UPDATE` DML statement supports the `WAIT` clause, allowing users to specify how much time they would like to wait for a row to become available, before rolling back the statement. This new feature adds support for `WAIT`, and `NOWAIT`, for all DML statements: `INSERT`, `UPDATE`, `DELETE` and `MERGE`.

The new `WAIT` and `NOWAIT` features enable users tell the database how to proceed if a transaction attempts to lock a row that is locked by another transaction. Specifying `NOWAIT` causes the database to rollback a DML statement immediately if a lock exists. Like the existing `SELECT FOR UPDATE` usage, the `WAIT` clause enables users to tell the database how long to wait for the row to become available, before rolling back. A new `WAIT FOREVER` clause instructs the database to retain its default behavior of waiting for row locks indefinitely.

View Documentation

13 Diagnosability

This section describes the new diagnosability features.

General

Cluster Health Monitor Improved Diagnosability

Cluster Health Monitor introduces a new diagnostic ability to listen for critical component events that could indicate pending or actual failure and report these with recommended corrective actions. In some cases, these actions may be executed autonomously. Such events and actions could then be captured and admins notified through components such as Trace File Analyzer.

Improving the robustness and reliability of Oracle AI Database hosting infrastructure is a critical business requirement for enterprises. This improved ability to detect and correct at first failure and self-heal autonomously delivers value by improving business continuity.

View Documentation

Diagnosability 26ai Improvements

These are existing features but changes and enhancements to functionality and default values.

Decreases time to identify and address critical events in the database. Changes tracing limits to be more reasonable than being unlimited and facilitates content identification so that customers are aware what trace data is provided to Oracle for further diagnosis.

View Documentation

Enhanced Cluster Health Advisor Support for Oracle Pluggable Databases

Cluster Health Advisor's (CHA) diagnostics capability is extended to support 4K pluggable databases (PDBs) from 256. This is critical for Oracle Autonomous Database deployments. CHA's problem detection and root cause analysis improves accuracy by considering database events such as reconfiguration. This improves detection, analysis, and targeted preventative actions for problems, such as instance evictions.

By adding this support to Cluster Health Advisor, performance and availability are kept in line with the deployment size. The business continuity of critical applications is preserved with improved prognostics and targeted preventive actions.

View Documentation

Oracle Update Advisor

Oracle Update Advisor is a software update recommendation framework that provides information to keep software at recommended versions.

Using Oracle Update Advisor simplifies software maintenance. In combination with tools such as the Database Creation Assistant (DBCA) and Fleet Patching and Provisioning (FPP), Oracle Update Advisor provides you with your software status and provides update and maintenance recommendations.

View Documentation

Oracle Update Advisor Support with Oracle FPP

You can leverage Oracle Update Advisor with Oracle Fleet Patching and Provisioning (Oracle FPP). Oracle Update Advisor is a software recommendation framework that analyzes the health of Oracle AI Database and Oracle Grid Infrastructure, provides patch recommendations, and creates fully functional gold images based on your patch maintenance policy. These gold images are downloaded automatically into the Oracle FPP Gold Image repository.

Using Oracle Fleet Patching and Provisioning in conjunction with Oracle Update Advisor greatly simplifies and streamlines the update experience. You can use the Oracle Update Advisor for both Oracle FPP Local Mode and Oracle FPP Central Server Mode.

View Documentation

SQL Diagnostic Report

The SQL diagnostic package (`DBMS_SQLDIAG`) includes a new function called `REPORT_SQL` which generates a zip archive containing a deep-level HTML diagnostic report for a specified SQL statement. The report includes important diagnostic information such as a plan history, non-default database parameters, optimizer statistics history, statistics status, index details, and more. The archive will also include, where available, captured SQL Monitor reports for the SQL.

This feature makes it easier to diagnose SQL performance issues.

View Documentation

Telemetry Streaming

Telemetry Streaming offers a comprehensive, turnkey solution for metrics streaming and observability with Oracle AI Database. It supports ultra-fast metrics ingestion using REST and PL/SQL. It enables users to query metrics data for monitoring and alerting purposes through SQL, PL/SQL, or PromQL (Prometheus' Query Language). Additionally, Telemetry Streaming automates data lifecycle management, including compression and downsampling for older metrics data, optimizing storage and performance over time.

Metrics streaming is essential for modern enterprise use cases, such as DevOps monitoring, asset tracking, and anomaly detection. While Oracle AI Database offers best-of-breed features for building a metrics streaming solution, assembling these components can be complex and time-consuming. Telemetry Streaming simplifies this process by packaging Oracle's powerful technologies into a streamlined, ready-to-use solution, enabling faster deployment, easier management, and reduced operational overhead for metrics streaming applications.

View Documentation

Reduce Time to Resolve

ADDM Extension

The Extensible ADDM feature provides a way for a DBA to register their own logic (written in PLSQL) as an extension that will be executed by ADDM. The PLSQL code for this logic can be in a user-created PLSQL package, which needs to be registered with ADDM via the register_rule API in package DBMS_ADDM.

Every registered rule has a publisher associated with it along with an owner who is the database user that registered the rule. The rule can be enabled to be run for manual ADDM tasks run by the owner, or for all tasks including automatic tasks run in the background.

Extensions are always executed as the user who registered the extension rule regardless of who ran the ADDM task. Due to this, even for automatic ADDM tasks run by `SYS`, the extensions are run only with the privileges of the owner of the extensions.

When an extension is registered, a parent rule can be provided under which the extension will be run, or it can be a top-level rule that will always be executed by ADDM.

Extensible ADDM provides many utility functions in the `DBMS_ADDM_UTIL` package that you can use in an extension for common tasks, such as fetching the value of a particular statistic or wait event from AWR, constructing a string that looks like an ADDM finding or recommendation, and performing other tasks.

The output of an extension will appear in a regular ADDM report and will be annotated with the publisher name to distinguish them from the built-in rules.

Many DBAs have carefully developed home-grown scripts to analyze performance issues they encounter. DBAs have to manage versions of these scripts and run them on every candidate database they manage. This makes the scripts difficult to manage, requires manual effort, and can lead to errors.

This feature allows DBAs to register such scripts with ADDM once and store them in the database, where they remain available even when the database is patched or upgraded.

View Documentation

Add Verified SQL Plan Baseline

The SQL plan management API (`DBMS_SPM`) includes a new procedure called `ADD_VERIFIED_SQL_PLAN_BASELINE`. It searches the cursor cache, AWR, and automatic SQL tuning set to establish which execution plan is best for a specified SQL statement. It creates an accepted SQL plan baseline for the best plan.

This feature provides improved performance management.

View Documentation

CMAN Diagnostics and Logging Enhancements

Using the command line interface, you can now monitor statistics for all database service registration operations (such as register, update, or unregister) that the Oracle Connection Manager (CMAN) listener performs. You can also view additional diagnostic details about service registration events in the CMAN and listener log files.

This feature enables you to evaluate statistics about service registration operations at both global and instance levels, analyze their traffic, and diagnose registration issues.

View Documentation

DBMS_DICTIONARY_CHECK PL/SQL Package

`DBMS_DICTIONARY_CHECK` is a read-only and light weight PL/SQL package procedure that helps you identify database dictionary inconsistencies that are manifested in unexpected entries in the RDBMS dictionary tables or invalid references between dictionary tables. Database dictionary inconsistencies can cause process failures and, in some cases, instances crash. `DBMS_DICTIONARY_CHECK` assists you in identifying such inconsistencies and provides a guided remediation to resolve the problem and avoid such database failures.

This feature improves database availability thus reducing the management and maintenance time for environments utilizing this package.

View Documentation

DBMS_SECUREFILES Package for SecureFile LOBs

The `DBMS_SECUREFILES` package provides expanded support for advanced SecureFiles subprograms, focused on diagnosability, advisory, and repair.

This package makes it easier to manage SecureFile LOBs.

View Documentation

Database Observability

Oracle's Kernel Service Distributed TRaCe (KSTRC) infrastructure provides a distributed tracing service built on top of the Unified Tracing Service (UTS or DBGT). KSTRC provides a powerful observability framework for understanding complex operations in a distributed system, allowing clients to gain insights into the execution of their code and identify potential issues. Oracle Database natively supports a single OpenTelemetry trace/span that helps in debugging end-to-end issues. The Oracle DB span will be correlated with the clients that support OpenTelemetry. By leveraging KSTRC, developers can improve the reliability, performance, and maintainability of

their applications. Metrics enable the collection and management of metrics within the VOS infrastructure.

KSTRC provides a comprehensive observability framework for Oracle databases. KSTRC enables distributed tracing, allowing customers to track complex operations end-to-end and identify performance bottlenecks. With these components, customers can:

1. Faster issue resolution: By providing detailed tracing and metrics, these components enable customers to quickly identify and diagnose issues, reducing mean time to detect (MTTD) and mean time to resolve (MTTR).
2. Proactive performance optimization: Customers can use the metrics and tracing data to optimize database performance, reducing the likelihood of issues and improving overall system efficiency.

KSMET and KSTRC provide a wealth of metrics and tracing data for enhanced performance monitoring and analysis, which can be used to analyze database performance. Customers can:

1. Monitor key performance indicators (KPIs): KSMET provides metrics on database performance, such as operation counts, latency, and throughput. Customers can use these metrics to monitor KPIs and identify areas for improvement.
2. Analyze tracing data: KSTRC's distributed tracing capabilities enable customers to analyze complex operations and identify performance bottlenecks.

View Documentation

Estimate the Space Saved with Deduplication

Before you enable deduplication, you can estimate the space that you can save by enabling advanced LOB deduplication for existing LOBS.

This enables you to take an informed decision to enable deduplication. Advanced LOB Deduplication enables Oracle AI Database to automatically detect duplicate LOB data within a LOB column or partition, and conserve space by storing only one copy of the data.

View Documentation

Extent-Based Scrubbing

Automatic Storage Management (ASM) extent-based scrubbing changes the granularity level on which ASM scrubs data from a file and disk group level to the extent level.

Compared to scrubbing the whole file, scrubbing specific extent sets significantly reduces the scrubbing turn-around time, improves the data availability, and minimizes the performance impact.

View Documentation

High Availability Diagnosability Using the DBMS_SCHEDULER Package

The Scheduler In-Memory Tracing feature is aimed at designing and implementing tools for the collection and temporary in-memory storage of scheduler trace messages generated during process execution.

It is critical to successfully restart jobs when they are interrupted by forced shut downs, like a forced patching cycle. With the addition of High Availability (HA) diagnostics in the `DBMS_SCHEDULER` package, you will be able to add real-time in-memory diagnostics during forced shut downs, and address any issues that result from these diagnostics.

This feature provides benefits like easier collection of trace messages generated since the initial failure, reduction in user interaction to collect traces, and significant reduction in multiple requests of problem reproduction.

View Documentation

In-Memory Advisor

The In-Memory Advisor is now part of Oracle AI Database and has two components: (1) an eligibility test that identifies databases that are not good candidates for Database In-Memory and (2) an advisor with enhanced analysis capability to better identify workloads that will benefit from Database In-Memory.

The In-Memory Advisor makes it easier and faster to identify databases that can take advantage of Database In-Memory. The In-Memory Advisor is now built into the database in place of having to install a separate standalone utility. An eligibility test provides the ability to quickly eliminate workloads that will not benefit from Database In-Memory, saving time and effort. An enhanced analysis capability that makes

identification of analytic workloads that will benefit from Database In-Memory simpler and more accurate. Together, these two components make it much simpler to decide where and when to use Database In-Memory.

View Documentation

Network Monitor

Network Monitor (NMON) is a low-overhead network monitoring service that leverages advanced kernel features to deliver always-on, comprehensive monitoring. It continuously captures network statistics and events at multiple levels, including system-wide, per interface, per IP, and per flow, providing detailed metrics for higher-level services. Built on NMON, the Health Score and Network Diagnostic Agent services simplify network health assessment with a standardized 0–100 score and proactively address issues by collecting diagnostics and rerouting traffic as needed.

NMON provides efficient and actionable network statistics, enabling timely detection of anomalies and effective correlation with database activities.

View Documentation

Oracle Call Interface (OCI) APIs to Enable Client-Side Tracing

New Oracle Call Interface (OCI) APIs allow applications to enable and disable client-side OCI diagnostic tracing dynamically without the need to update configuration files or set environment variables.

This feature allows developers to improve OCI application problem troubleshooting and reduce issue resolution time.

View Documentation

Rename LOB Segment

To rename an existing LOB segment users perform an operation such as `ALTER TABLE ... MOVE`, which could perform slowly since the operation physically moves the LOB data as part of the renaming.

This enhancement improves the performance of renaming a LOB segment, at the table, partition and subpartition level by eliminating the physical movement of the LOB data.

View Documentation

14 Installation, Upgrade, and Patching

In Oracle AI Database 26ai, you get significant improvements with the installation process, especially with upgrades using the AutoUpgrade utility.

Installation

Archiving and Unarchiving of Gold Images

You can archive and unarchive Oracle FPP gold images that are not currently used but cannot be deleted for future access. Archiving and unarchiving gold images allow you to store those images on external storage devices of your choice in a space-efficient, compressed fashion, thereby freeing up storage space on high-end storage devices hosting the central FPP gold image repository.

Archiving gold images that are not currently used, but need to be retained, saves space and costs by allowing you to flexibly and efficiently store them on external storage devices of your choice.

View Documentation

Backup, Restore, and Relocation for FPP Server

You can create a backup of the Oracle Fleet Patching and Provisioning (Oracle FPP) server, restore data from the backup, and relocate the server from backup to new hardware. Relocate the Oracle FPP Server to new hardware and re-point Oracle FPP targets whenever needed.

Ensure data safety by backing up Oracle FPP Server and restoring data in case of failure.

View Documentation

Oracle AI Database Installer Command-Line Support

Oracle AI Database Installer now supports specifying commands and input parameters for those commands using the command-line interface.

Easier and simpler Oracle AI Database deployments are supported using the command-line interface in addition to the graphical user interface.

View Documentation

Oracle FPP Local Mode Without Java Container

Oracle Fleet Patching and Provisioning Local Mode does not require the Grid Infrastructure Management Repository (GIMR) and the Java container.

Oracle Grid Infrastructure and Oracle AI Database administrators can use Oracle FPP Local Mode without setting up any additional components on a cluster, making the patching process simpler and faster.

View Documentation

Oracle FPP Metadata on External Databases

Starting with Oracle Grid Infrastructure 26ai, Oracle Fleet Patching and Provisioning (FPP) stores metadata in a local Single Instance Enterprise Edition Database or an external Oracle metadata repository. New installations will store the metadata by default in a Single Instance Enterprise Edition Database.

Allowing to choose an external metadata repository database or a local Single Instance Enterprise Edition Database database simplifies the deployment of the Fleet Patching and Provisioning server.

View Documentation

Oracle Grid Infrastructure Installer Command-Line Support

Oracle Grid Infrastructure Installer now supports specifying life-cycle management operations and input parameters for those operations on the command line.

Easier and simpler Oracle Grid Infrastructure deployments are supported using the command line, in addition to the graphical user interface.

View Documentation

Oracle Grid Infrastructure Installer Improvements

The Oracle Grid Infrastructure installer has been upgraded with options to create and manage gold images and perform out-of-place patching while reducing the inventory metadata to effectively manage installation and patching.

Out-of-place patching using the Oracle Grid Infrastructure Installer directly makes patching more manageable and reliable.

View Documentation

Store Images and Transfer Working Copies as ZIP Files

Store Oracle Fleet Patching and Provisioning (Oracle FPP) gold images as ZIP files, create ZIP files from existing Oracle homes, and transfer these ZIP files.

Save significant storage, bandwidth, and transfer time by storing and transferring gold images as ZIP files.

View Documentation

Upgrade

AutoUpgrade Release Update (RU) Upgrades

AutoUpgrade supports the option of using AutoUpgrade to perform out-of-place Oracle home Release Update patching.

For an out-of-place patch of Oracle AI Database using AutoUpgrade, AutoUpgrade moves the source database that you want to patch to a new Oracle AI Database Oracle home, and then patches the database binaries in that target Oracle home with the Release Update that you select. With this option, you can use AutoUpgrade at any time that you want to move the database to a new Oracle home, either as part of a planned upgrade or as part of a patch plan. In a patch operation, AutoUpgrade performs the patch using the following workflow:

1. AutoUpgrade recognizes that the source database and the target Oracle AI Database are the same base release.
2. AutoUpgrade skips the upgrade steps.
3. AutoUpgrade patches the target database using the Release Update.

View Documentation

AutoUpgrade Sets Parallelism Based on System Resources

AutoUpgrade automatically evaluates system resources and makes an intelligent decision as to how many upgrade jobs can run simultaneously.

AutoUpgrade uses the `CPU_COUNT` value and system process parameters to determine available system resources, and calibrates both the number of upgrades that can run at a time and the number of parallel threads for each upgrade. Upgrades that exceed a

safe threshold are put in a queue so that they can be run as system resources become available.

View Documentation

AutoUpgrade Supports Upgrades with Keystore Access to Databases Using TDE

AutoUpgrade enhances support for databases that use transparent data encryption (TDE) by enabling keystore generation.

AutoUpgrade now enables you to provide passwords to an external key manager generated and maintained by AutoUpgrade. With this configuration, AutoUpgrade supports unmanned or automated operations of TDE-enabled databases. AutoUpgrade can open the source database keystore without prompting for the keystore password, and enroll the target database into the TDE external keystore for key management, so that the target database can start automatically.

View Documentation

AutoUpgrade Unplug-Plugin Upgrades to Different Systems

You can now use the AutoUpgrade Unplug/Plug method in Oracle AI Database to unplug a PDB from one system and plug into a different system and upgrade.

In earlier releases, AutoUpgrade supported unplug/plug/upgrades on the same server, but it was not possible to unplug a PDB from one server, plug it into a different system, and then upgrade the PDB. With this feature, you can now migrate and upgrade the PDB in a single operation, including migrations to the cloud.

View Documentation

REST APIs for AutoUpgrade

To facilitate safe and secure remote use of the AutoUpgrade for Oracle AI Database upgrades, AutoUpgrade now provides REST APIs (ORDS and OCI).

The Oracle REST Data Services (ORDS) database API is a database management and monitoring REST API embedded into Oracle REST Data Services. The Oracle Cloud Infrastructure (OCI) REST API is enabled by configuring the REST Adapter connection to use the OCI Signature Version 1 security policy. You can now use these features to run AutoUpgrade upgrades remotely over SSH.

View Documentation

Patching

Gold Image Based Out of Place Patching

Oracle DBCA can apply Release Updates (RUs) out-of-place using Gold Images for both Oracle Grid Infrastructure homes and Oracle AI Database homes.

DBCA greatly simplifies and automates the process of applying new Gold Image-based quarterly Release Update patches to Oracle Grid Infrastructure and Oracle AI Database homes using command line parameters.

View Documentation

15 Appendix A: Oracle AI Database 26ai Release Updates

This section lists the new features for Oracle AI Database 26ai by Release Update.

Release Update 23.26.4

- ADDM Extension
- AI Vector Search: Scalar Quantized IVF Indexes
- Asynchronous Cross-Region Replication for Globally Distributed Databases Deployed with Raft Replication
- Automatic Session State Restoration for DRCP and PRCP Implicit Pooling
- BAR_CHART Function
- DBMS_AI_GRAPHRAG PL/SQL Package
- Data Type Override for Iceberg Tables
- Extend Select AI Translate for Google, AWS, and Azure Translation Services
- Iceberg Table Properties
- Iceberg Update and Delete
- Included Columns Support for Distributed HNSW Indexes
- Incremental Schema Inference for the JSON-to-Duality Migrator
- Incremental Snapshots Support for Distributed HNSW Indexes
- MLE WebAssembly Modules and JavaScript WebAssembly API
- Mixed SELECT * and other columns/expressions
- Multi-Vector Search with HNSW Indexes
- Notification Directives
- Oracle ODBC Driver Support for Application Continuity and Transparent Application Continuity
- Prefetch Implicit Results with Top-Level Procedure Execution
- Product Quantized HNSW Indexes
- Promotion of Sessionless Transaction to XA Transaction
- Runtime Staleness Detection and Fast Refresh for Materialized Views Containing External Tables
- SELECT FOR UPDATE with Row Limit
- Select AI Support for NL to Graph Queries
- Select AI Supports Specifying Custom LLM via Private Endpoint in Customer-Hosted Environment
- Simplified Macros
- Streaming Windowing Table Functions
- Support for AC/TAC in Precompilers Using Session Pool
- Transparent Sharding in OCI Session Pools

- Witness Replica
- ezTLS

Release Update 23.26.3

- Application Identity Logon
- Array Limits for JSON-Relational Duality Views
- Cross-Table Data Grant Support
- Deletable Shared Tables for JSON-Relational Duality Views
- Intelligent Drain Timing
- Log-Buffer-Space-Based Auto Redo Prioritization
- MAA Client Readiness
- OCI API Functions to Validate and Protect Inputs
- Oracle AI Database Read API for Redis
- PDB Target Redo Generation Management
- Private AI Support for Large Language Models, Reranking, and Text Classification
- Scalar Quantization Support for Distributed HNSW Indexes
- Setting Multi-Factor Authentication for Local Deep Sec End User
- Support for DATEADD Function
- Transport-Lag-Based Auto Redo Prioritization
- Unicode 17.0 Support
- User-Defined Identifying Keys for JSON-Relational Duality Views
- Using OCI IAM DB Token to Authenticate with Deep Data Security

Release Update 23.26.2

- Automatic Block Repair in Active Data Guard per Pluggable Databases
- Automatic Standby Pluggable Database Instantiation and Standby Redo Log Creation
- DBMS_SEARCH Autocomplete for Hybrid Vector Indexes
- DML Redirection for Data Guard per Pluggable Database
- Deep Security Support in Java
- Duality View Inline Write Augmentation
- EBCDIC Binary Collation Support
- IBM-Compatible EBCDIC Client Character Set Support
- In-Place Conversion to Immutable Tables and Parallel CTAS Support
- JOIN TO ONE Join Method and Syntax
- Local HNSW Indexes
- New SQL Functions: VECTOR_ORIGIN, VECTOR_RANDOM, and VECTOR_RANDOM_PER_ROW

- Offload Creating Vector Indexes to Remote GPU Container
- Oracle Deep Data Security (Deep Sec)
- Parallel Verification of Blockchain Tables
- Private AI Container for Vector Embedding Generation
- Property Graph: In-Database Graph Algorithms
- Support for DECODE and CASE in SQL/JSON Path Expressions
- Support for Nested WITH Clause and Correlation
- TLS Authentication for Redo Transport
- Validation Directives for JSON Relational Duality Views
- WAIT and NOWAIT Support for All DML Statements

Release Update 23.26.1

- ALTER INDEX Support for JSON Search Index
- Aggregation Filters
- Assertions, a Powerful New Constraint Type to Simplify Data Integrity Enforcement
- Auto IVF Vector Index Reorganization
- Calendar Functions
- Configuration and Member Tagging for Oracle Data Guard
- DBMS_SECUREFILES Package for SecureFile LOBs
- Database Observability
- Distributed HNSW Indexes with RAC
- Enable Larger ONNX-Format Models
- Included Columns with HNSW Indexes
- JavaScript Web API for In-Database JavaScript
- Kerberos Constrained Delegation for DB Links
- Migrating LOBs Made Easier
- Network Monitor
- Online Index Rebuild
- Partition by Expression
- Post-Quantum Cryptography Support
- Scalar Quantized HNSW Indexes
- Select AI Capabilities
- Simple Dot-Notation Syntax for JSON Indexes
- Smart Connection Rebalance Recommendation
- Support for DATEDIFF Function
- Support to Create SQL Property Graphs from Database Views
- True Cache Support in OCI Session Pools
- Vector Data Type Support in Precompiler

Release Update 23.26.0

- AI Vector Search: JDBC Support Completion for SPARSE Vectors with a Binary Format
- Enhancements to GraphQL Syntax for Duality View Creation
- GraphQL Schema Introspection API
- GraphQL Table Function for SQL
- Heat Map Retention Time
- JSON Output for the Data Guard Command-Line
- JSON Replication
- Materialized View Query Rewrite with Cursor Sharing in the Presence of Bind Variables
- Memory Speed Columnar
- Move Data Chunks Between Shardspaces
- RESETTABLE Clause
- Restrict Switchover and Failover to a List of Databases
- SQL Objects and Collections in MLE JavaScript
- Sparse Vector Support in MLE JavaScript
- Support for the QUALIFY Clause
- Telemetry Streaming

Release Update 23.9

- Compile-Time Syntax Checking for JavaScript Procedures and Functions
- EXECUTE Privilege on JAVASCRIPT Not Required Anymore
- GROUP BY ALL
- Generate and Test UUID Using SQL
- IVF Index Online Reorganization
- JSON to Duality Migrator: Schema Inference using JSON Schema
- Multi-Factor Authentication for Database Authentication
- Multilingual Engine Supports Database-Resident Connection Pools
- Non-Positional INSERT Clause
- Oracle Update Advisor
- Oracle Update Advisor Support with Oracle FPP
- Partition Maintenance Operations and Direct Load with Global IVF and HNSW Indexes

Release Update 23.8

- Additional Predicate Support with Hybrid Vector Search
- Client Support for SPARSE Vectors

- Dynamic Statistics for PL/SQL Functions
- Elastic Vector Memory Management
- Included Columns Support for JSON, BLOB and CLOB Data Types
- JSON Type Modifier Enhancements: Data Size and Array Specifications
- JSON to Duality Migrator: Hints Configuration Field
- JSON_TEXTCONTAINS and JSON_EXISTS Support in DBMS_HYBRID_VECTOR.SEARCH
- Oracle Database Cloud Backup Module for Azure Blob Storage
- Restricted Execution Contexts for In-Database JavaScript
- Sparse Vector Support in PL/SQL
- User-Defined Vector Distance Functions

Release Update 23.7

- AI Vector Search: Arithmetic and Aggregate Operations
- Additional Flexibility Defining JSON-Relational Duality Views
- Cloud Developer Packages
- DBMS_DEVELOPER Package
- Dimension-Wise Arithmetic Support in PL/SQL
- Foreign Function Interface for JavaScript to Call PL/SQL Code Units
- Hybrid Vector Index for JSON
- In-Database Algorithms Support for VECTOR Data Type Predictors
- Included Columns in Neighbor Partition Vector Indexes
- JSON to Duality Migrator: Multi-Collection Import API
- JSON to Duality Migrator: Validation of Schema and Data
- Materialized Expression Columns
- PL/SQL BINARY Vector Support
- PL/SQL JACCARD Distance Support
- Set COMPATIBLE to 23.6.0 to Enable New AI Vector Search Features in 26ai
- Sharding Support for AI Vector Search
- Smallfile Tablespace Shrink
- Support for Image Transformer Models with AI Vector Search Using the In-Database ONNX Runtime
- Time Bucketing
- Vector Data Type Support in External Tables

Release Update 23.6

- AI Vector Search: New Vector Distance Metric
- CONSTRAINT_NOVALIDATE, a Data Pump Import TRANSFORM Parameter
- Disable Statistics Gathering During Autonomous Database Import Operations

- Enhancements to Oracle Data Redaction
- GoldenGate Replication of JSON-Relational Duality Views
- Hybrid Vector Index
- JSON Collection Views
- JSON Search Index Path Subsetting
- JSON-Relational Duality Views: Add Fields With Calculated Values
- JSON-Relational Duality Views: Hidden and Generated Fields
- Parallel Index Creation Parameters for Data Pump Import
- Partition-Local Neighbor Partition Vector Index
- Persistent Neighbor Graph Vector Indexes
- Replication Support for JSON Collection Tables
- Sessionless Transactions
- Sparse Vectors
- Transactional Support for Neighbor Graph Vector Indexes
- Vector Format Output for Feature Extraction Algorithm
- XMLTYPE_STORAGE_CLAUSE, a Data Pump Import TRANSFORM Parameter

Release Update 23.5

- Archiving and Unarchiving of Gold Images
- BINARY Vector Dimension Format
- Backup, Restore, and Relocation for FPP Server
- Custom Certificates for Oracle FPP Server Authentication
- Duplicated HNSW Vector Indexes on RAC
- General Improvements for Oracle FPP Job Scheduler
- General Purpose Cluster Configuration
- Gold Image Based Out of Place Patching
- JSON Collections
- JSON_ID SQL Operator
- Optimized Oracle Native Access to NVMe Devices Over Fabric
- Oracle AI Database Installer Command-Line Support
- Oracle DBCA Support for PMEM Storage
- Oracle DBCA Support for Standard Edition High Availability
- Oracle FPP Local Mode Without Java Container
- Oracle FPP Metadata on External Databases
- Oracle Grid Infrastructure Installer Command-Line Support
- Oracle Grid Infrastructure Installer Improvements
- Single-Server Rolling Database Maintenance
- Store Images and Transfer Working Copies as ZIP Files
- Vector Memory Pool Automatic Management
- Verifying Digital Signature and Integrity of Installation Archive Files