ORACLE

Oracle Database 23C 的现在和未来

Cary Dong

Safe harbor statement

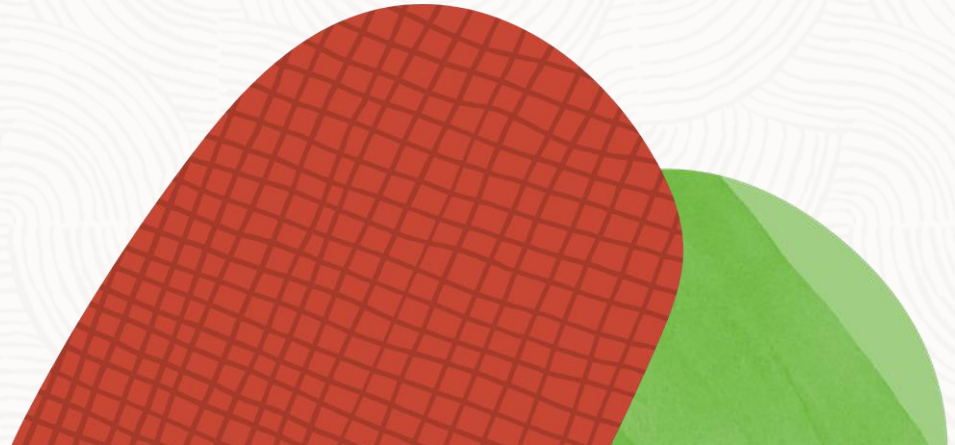
The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, timing, and pricing of any features or functionality described for Oracle's products may change and remains at the sole discretion of Oracle Corporation.

The materials in this presentation pertain to Oracle Health, Oracle, Oracle Cerner, and Cerner Enviza which are all wholly owned subsidiaries of Oracle Corporation. Nothing in this presentation should be taken as indicating that any decisions regarding the integration of any EMEA Cerner and/or Enviza entities have been made where an integration has not already occurred.

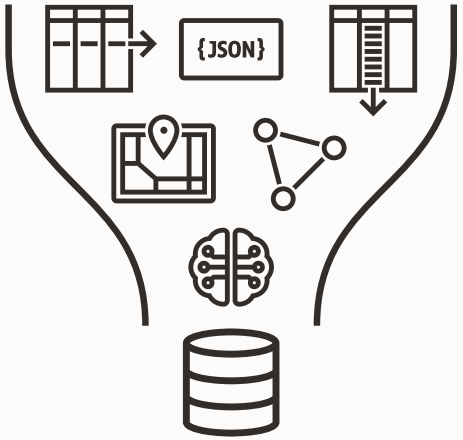
Make it easy to develop and run modern apps
and analytics for all use cases at any scale

让开发和运行
任何规模的任何应用和分析
都非常容易

Oracle数据库愿景



如何实现我们的愿景 –
完善而简单的平台，满足所有数据管理需求



融合数据库

完美支持所有现代数据类型、工作负载、开发风格

完备的一致、可扩展、可用且安全

自治数据库

融合数据库作为自动驱动、自动安全、自动修复的云数据库提供

易用的云数据库，用于以任何规模或关键性运行任何应用程序

不断增强的版本迭代...

2013

12c

- 多租户体系架构
- 支持JSON
- In-Memory选项
- 应用连续性保护
- 透明表空间加密
- 分片技术

长期支持版本

2018

18c

- 集成Active Directory
- 基于RAC的分片
- PDB级别快照轮播
- 私有临时表
- 内存优化行存储
- 外部表支持In-Memory

创新版本

2019

19c

- 高速数据摄入
- Active Data Guard DML重定向
- 自动索引
- 混合分区表
- JSON支持增强
- 安全特性增强

长期支持版本

2021

21c

- 源生区块链表
- 自动机器学习
- PDB级别的Data Guard
- 源生JSON数据类型
- SQL宏
- JavaScript存储过程

创新版本

23c App Simple

下一个长期支持版本

- 为云服务和本地部署持续创新的融合数据库
- 为简化开发持续推出的新特性，以支持任意规模的任何类型的应用和分析

BETA 版已发布，可在Oracle Cloud或本地部署环境中使用

https://tinyurl.com/OracleBeta

现已发布

Oracle 23c 免费开发版

帮助您了解 Oracle Database 23c 如何使数据驱动应用程序的开发和运行变得更加简单

现在就去下载

Oracle 23c 数据库服务

Oracle云提供受托管的数据库服务，以便开发人员构建新的微服务、图形、文档或关系应用程序

Oracle 23c – 下一个长期支持版本

New In 23c

SQL Domains

Schema Level Privileges

Oracle Database 23c App Simple

Read-Only Per-PDB Standby

Property Graphs

Real-time SQL Plan Management

JSON Schema

Microservice Support

JSON / Relational Duality

AI Vector Search

True Cache

SQL Firewall

Priority Transactions

Globally Distributed Database

Rolling Patching

JS Stored Procedures

Developer Role

Shrink Tablespace

Boolean Datatype

新特性 | Oracle 23c JSON关系二元性

存储格式

first_name	order	quantity
Jake	{ “product” : ”TV”, “price” : 250 }	1

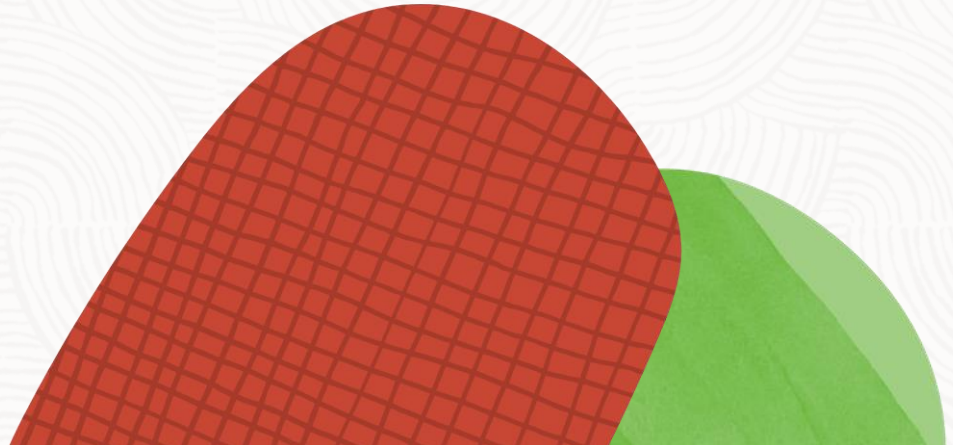
交互格式

```
{
 "first_name" : “Jake",
 "order" :
   {
      "product" : "TV",
      "price" :  250
   },
 "quantity" : 1
}
```

JSON 关系二元性视图提供简单的JSON格式数据访问方式，同时使用关系模型来提升数据的存储和维护效率

JSON 关系二元性

演示



File Edit View Run Kernel Tabs Settings Help

Oracle23c Examples.ipynb

Code

Python 3 (ipykernel)

```
[ ]: %%sql
select c.id, c.first_name, c.last_name, c.dob, c.email, c.address, c.zip,
       o.id, o.product_id, o.order_date, o.total_value, o.order_shipped
from new_customers c , new_orders o
where c.first_name = 'Dom'
and last_name = 'Giles'
```

We can now create our duality view `CUSTOMERS_DV`. The thing to note is that it follows the structure of the document we are looking to create. Anyone who's familiar with GraphQL shouldn't have any problems getting to grips with the syntax. Alternatively, you can define them using a "select" statement like syntax.

```
[ ]: %%sql
CREATE or REPLACE JSON RELATIONAL DUALITY VIEW customers_dv AS
    new_customers @insert @update @delete
    {CustomerID      : id
     FirstName       : first_name
     LastName        : last_name
     DateOfBirth     : dob
     Email           : email
     Address         : address
     Zip             : zip
     orders : new_orders @insert @update @delete
       [ {OrderID             : id
          ProductID           : product_id
```

Simple 0 2 Python 3 (ipykernel) | Idle Mode: Command Ln 5, Col 1 Oracle23c Examples.ipynb 1

新特性 | Oracle 23c 可操作属性图

- **第一个** 支持SQL/PGQ 标准的商业数据库
- 简单化**传统交易型**系统中的属性图分析，SQL更加简单易于维护

Before SQL Property Graph

```
-- Snippet from a 50+ Line SQL Query
SELECT "v1.id" AS "THOM_ID", "v2.id" AS "LARRY_ID"
FROM (SELECT "V1"."ID" AS "v1.id", "V2"."ID" AS "v2.id"
FROM "GRAPHUSER"."USERS" "V1", "GRAPHUSER"."USERS" "V2", (WITH t1(src_table, src_key,
         dst_table, dst_key,
         exp, lvl)
AS (
  -- Anchor member.
  SELECT "src_table" AS src_table, "src_key" AS src_key, "src_table" AS dst_table, "src_key" AS dst_key,
  '' AS exp, 0 AS lvl   FROM(SELECT 'USERS' AS "src_table", v1.id AS "src_key"
FROM "GRAPHUSER"."USERS" "V1"
WHERE ("V1"."ID" = 1))
  UNION ALL
  -- Recursive member.
  SELECT t1.src_table, t1.src_key,
         t2."dst_table" AS dst_table, t2."dst_key" AS dst_key,
         t1.exp || t2."exp" AS exp,
         t1.lvl + 1 AS lvl
    FROM (SELECT 'USERS' AS "src_table", "anonymous_1".follower_id
    AS "src_key", 'USERS' AS "dst_table", "anonymous_1".followed_id AS "dst_key", '' AS "edge_table",
    '' AS "edge_key", (('<EXPRESSIONS>' || '') || '</EXPRESSIONS>') AS "exp"
FROM "GRAPHUSER"."FOLLOWS" "anonymous_1"
WHERE (NOT("anonymous_1"."FOLLOWER_ID" IS NULL) AND NOT("anonymous_1"."FOLLOWED_ID" IS NULL))) t2, t1
... 40+ lines
```

SQL Property Graph

```
SELECT DISTINCT Thom_ID, Harry_ID from graph_table(social_graph
MATCH (v1)-[IS follows]->{1,5}(v2)
WHERE v1.id = 1 AND v2.id = 15
COLUMNS (v1.id AS Thom_ID, v2.id AS Harry_ID));
```

可操作属性图

演示

Oracle23c Examples.ipynb

Markdown

Python 3 (ipykernel)

Property Graph and PG/SQL support

Oracle Database 23c is the first commercial database to add support for the SQL/PGQ standard. This change adds in-database support for Graph models directly into the database. Graphs edges and nodes can now be modeled on new or existing datasets and queried with the SQL/PGQ extensions that make it simpler to navigate complex relations. And you can build your Graphs using relational or JSON Document datatypes... or a mix of both.

In this section we'll place a grph model on top of the two table we created earlier and query them with SQL/PGQ

Lets add a few more customers to start with

[458]:
```
%%sql
insert into new_customers (id, first_name, last_name, email, address)
values
(43242, 'Katie', 'Giles', 'kgiles@gmail.com', '10 john hope road'),
(75567, 'Elizabeth', 'Green', 'egreen777@hotmail.com', '2 main street'),
(215456, 'Susan', 'Giles', 'susan.giles@gmail.com', '122 arthur road'),
(96743, 'Mark', 'Blue', 'mblue55@hotmail.com', 'orchard house, home road'),
(47943, 'Robert', 'Yellow', 'robyel888@gmail.com', 'appartment 12, 120 west street'),
```

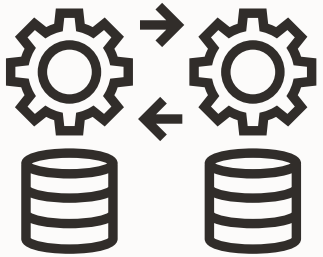
Simple | 0 | 2 | Python 3 (ipykernel) | Idle | Mode: Command | Ln 1, Col 296 | Oracle23c Examples.ipynb | 1

新特性 | Oracle 23c (23.3) 为开发者提供便利

SQL 增强

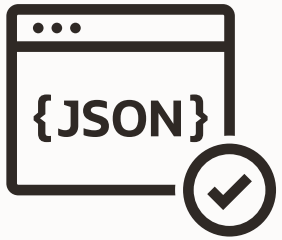
SQL 域

无锁列值保留



对微服务的支持

JavaScript
便于开发

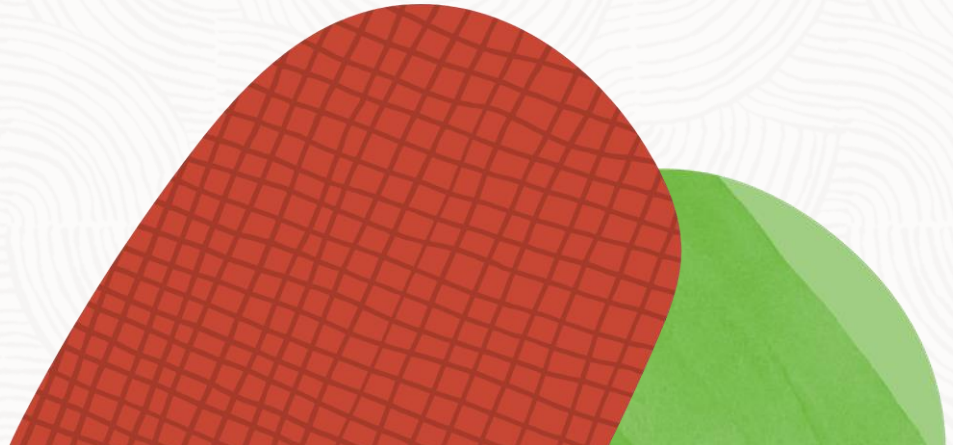


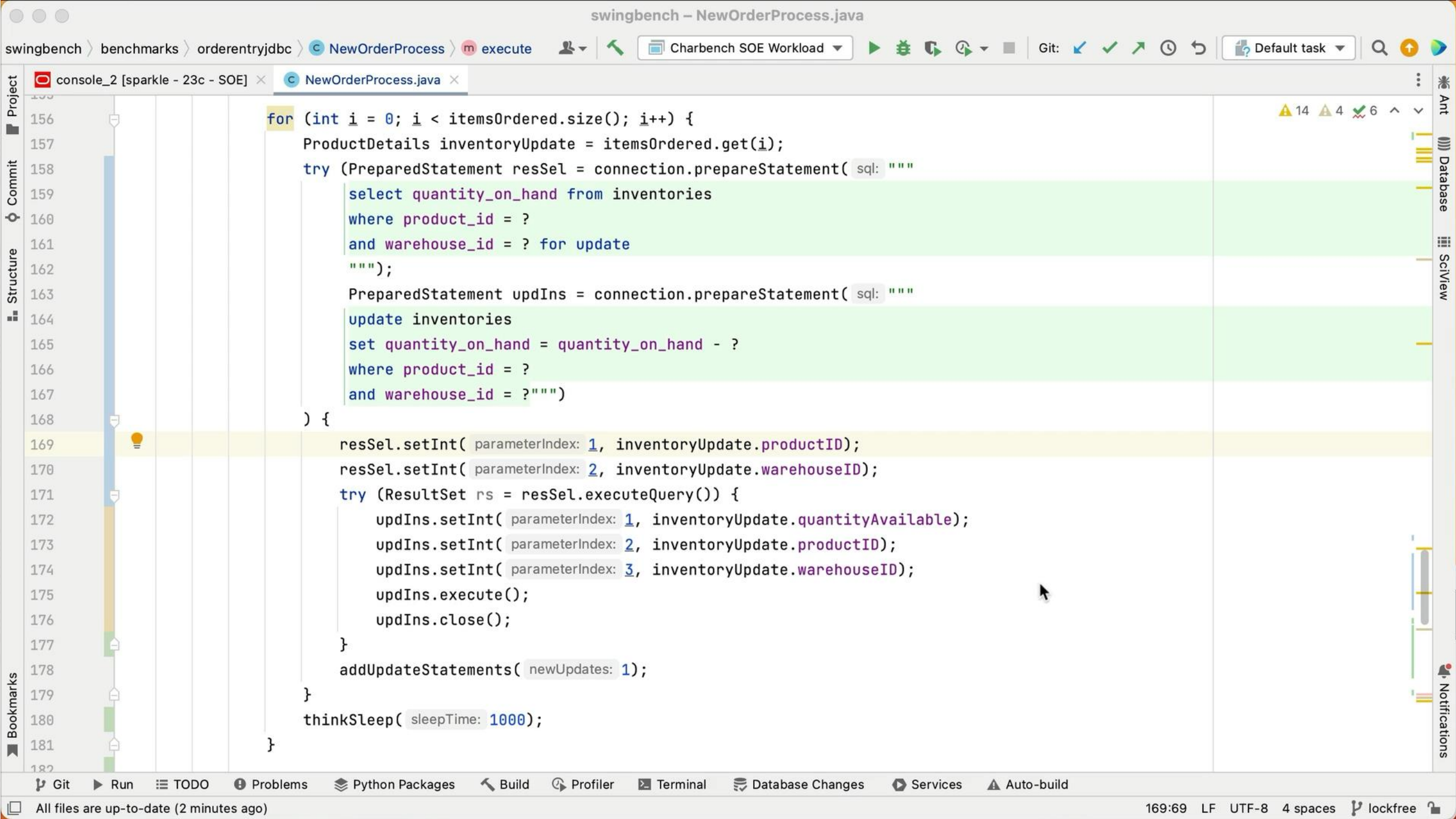
JSON Schema

15 Oracle Copyright © 2023, Oracle and/or its affiliates

无锁列值保留

演示



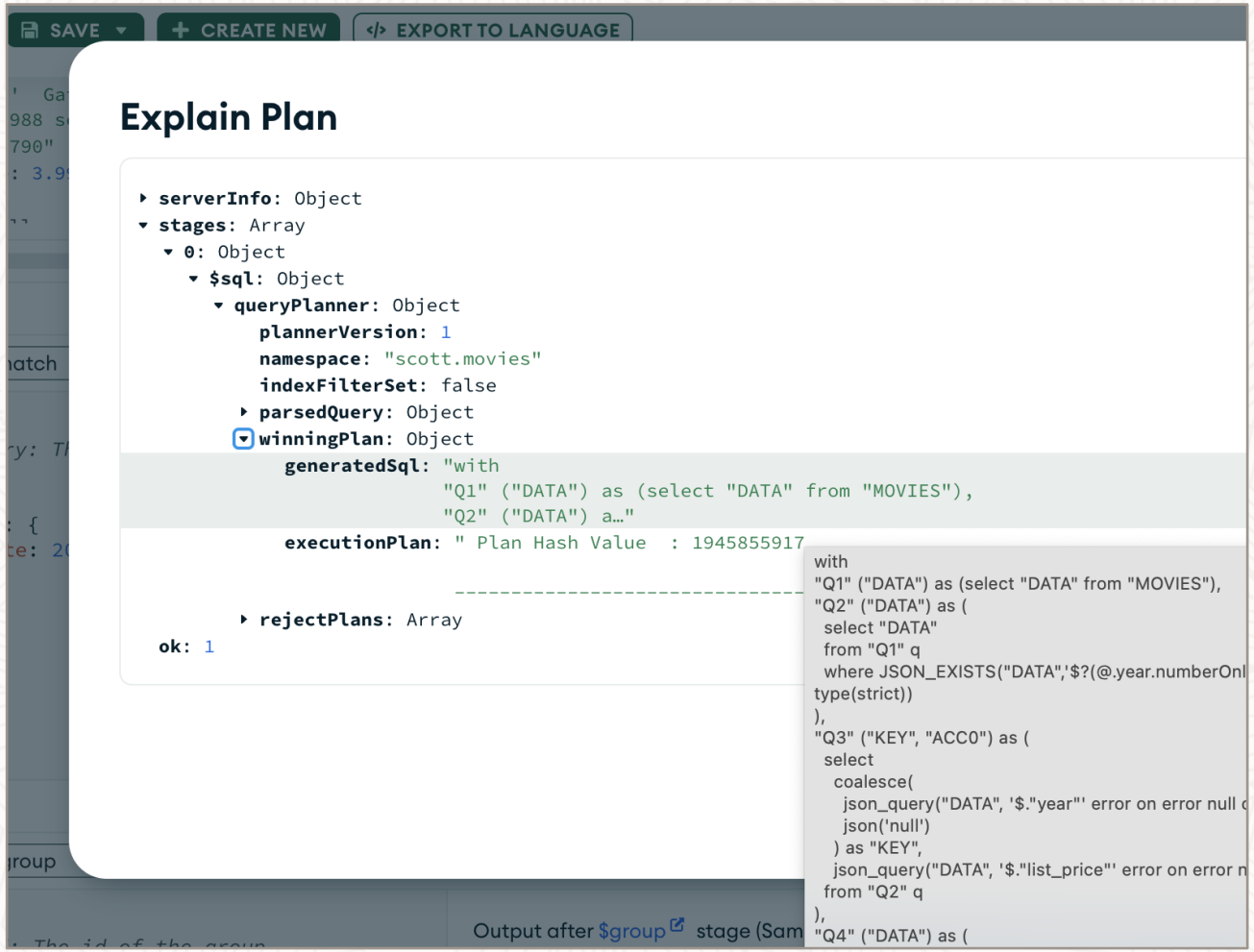

swingbench – NewOrderProcess.java
swingbench 〉 benchmarks 〉 orderentryjdbc 〉 NewOrderProcess 〉 execute
Charbench SOE Workload
Git:
Default task
console_2 [sparkle - 23c - SOE]
NewOrderProcess.java
14 4 6
for (int i = 0; i < itemsOrdered.size(); i++) {
ProductDetails inventoryUpdate = itemsOrdered.get(i);
try (PreparedStatement resSel = connection.prepareStatement(sql: """
select quantity_on_hand from inventories
where product_id = ?
and warehouse_id = ? for update
""");
PreparedStatement updIns = connection.prepareStatement(sql: """
update inventories
set quantity_on_hand = quantity_on_hand - ?
where product_id = ?
and warehouse_id = ?""")
) {
resSel.setInt(parameterIndex: 1, inventoryUpdate.productID);
resSel.setInt(parameterIndex: 2, inventoryUpdate.warehouseID);
try (ResultSet rs = resSel.executeQuery()) {
updIns.setInt(parameterIndex: 1, inventoryUpdate.quantityAvailable);
updIns.setInt(parameterIndex: 2, inventoryUpdate.productID);
updIns.setInt(parameterIndex: 3, inventoryUpdate.warehouseID);
updIns.execute();
updIns.close();
}
addUpdateStatements(newUpdates: 1);
}
thinkSleep(sleepTime: 1000);
}
Project
Commit
Structure
Bookmarks
Ant
Database
SciView
Notifications
Git
Run
TODO
Problems
Python Packages
Build
Profiler
Terminal
Database Changes
Services
Auto-build
All files are up-to-date (2 minutes ago)
169:69 LF UTF-8 4 spaces lockfree

新特性 | Oracle 23c (23.3) MongoDB API 支持聚合管道（Aggregation Pipelines）

转化为Oracle的分析SQL

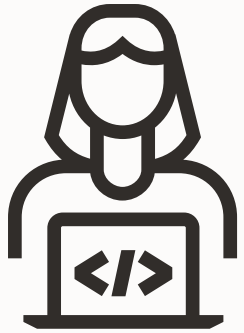
- Mongo管道操作等同的SQL操作
- 转化为SQL的流式操作并提升性能

支持的通用Stages和表达式
($group, $match, $project, $sort, etc.)



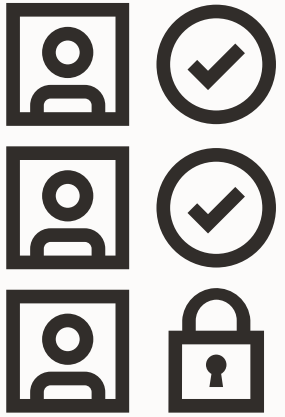
新特性 | Oracle 23c (23.3) 安全方面

SQL 防火墙



Developer 角色

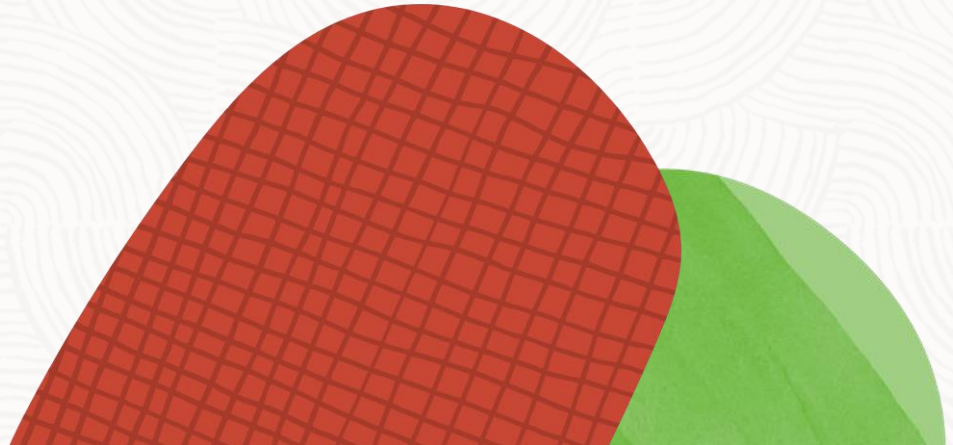
区块链表



Schema级别授权

Schema 级别授权

演示



Oracle SQL Developer : OCI_23c_APP_DATA~1

Connections

Oracle Connections
OCI_23c_APP_DATA
Tables (Filtered)
ADAMS_BREWER
ADAMS_RIOS_AND_DAUGHERTY
AGUILAR_RIVERA_AND_QUINN
ALEXANDER_MCDANIEL
ALEXANDER_PLC
ALEXANDER_RUBIO
ALLEN_AND_SONS
ALLEN_INC
ALLEN_LTD
ALLEN_MORGAN
ALLEN_OLSON
ALLEN_PARSONS_AND_WATKINS
ALLEN_SCOTT
ALLEN_WILLIAMS_AND_WHITE
ALVARADO_CALLAHAN
ALVAREZ_DAVIS
ALVAREZ_GROUP
ALVAREZ_PEREZ
ALVAREZ_ROGERS_AND_WILLIAMS
ANDERSON_AND_SONS
ANDERSON_COLLINS_AND_GOODMAN
ANDERSON_NGUYEN_AND_GRAVES
ANDRADE_NORTON_AND_MORRIS
ANTHONY_STEWART
ARCHER_PATEL
ARMSTRONG_LONG_AND_PARKER
ATKINS_OWEN
ATKINSON_LTD
AUSTIN_GROUP
AUSTIN_INC
AVILA_CHAMBERS_AND_HART

OCI_23c_APP_DATA~1 | OCI_23c_SYS | OCI_23c_APP_USER | grant_access.sql

0.17900001 seconds

OCI_23c_APP_DATA

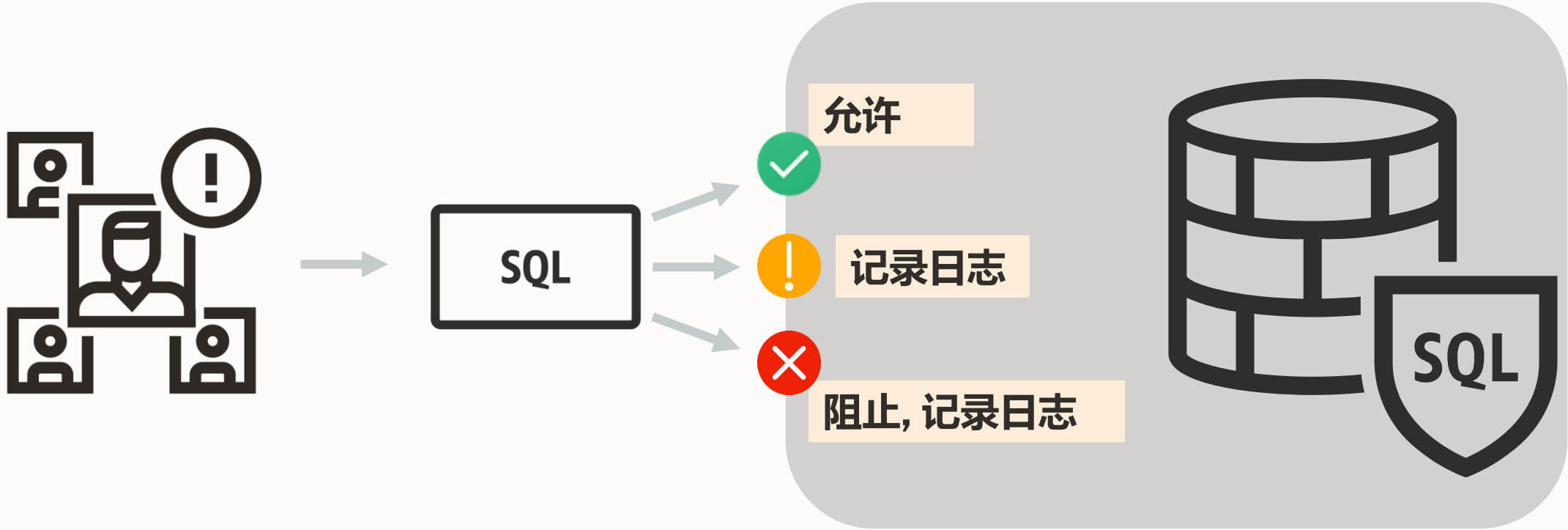
Worksheet | Query Builder

```
1  -- Work Request #246752-1
2
3  -- ADD new Table ORDERS_COMPLETED
4
5  CREATE TABLE IF NOT EXISTS ORDERS_COMPLETED (ORDER_ID NUMBER, ORDER_COMPLETED_ON TIMESTAMP, COMPLETED_BY NUMBER);
6
7  -- DROP Table CUSTOMERS_RELATIONSHIPS
8
9  DROP TABLE IF EXISTS CUSTOMERS_RELATIONSHIPS;
10
11 -- End of Work Request
12
13
14
15
```

Click on an identifier with the Command key down to perform "Go to Declaration"

Line 9 Column 40 | Insert | Modified | Unix/Mac: LI

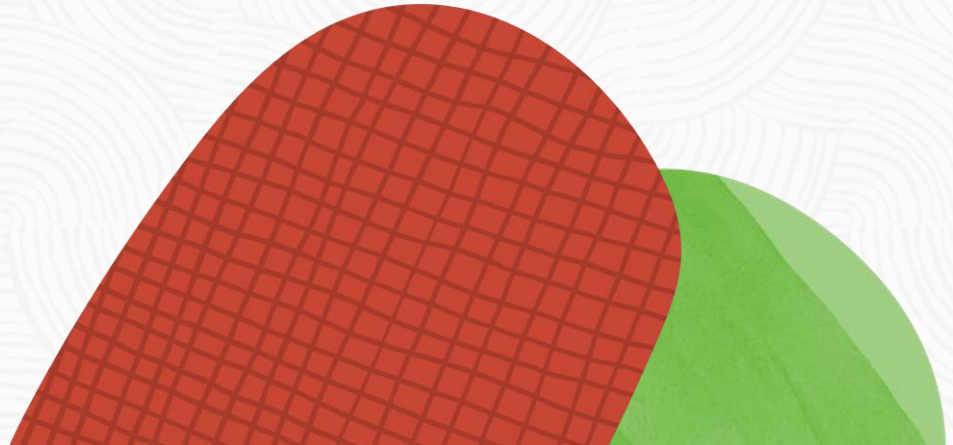
新特性 | Oracle 23c SQL 防火墙

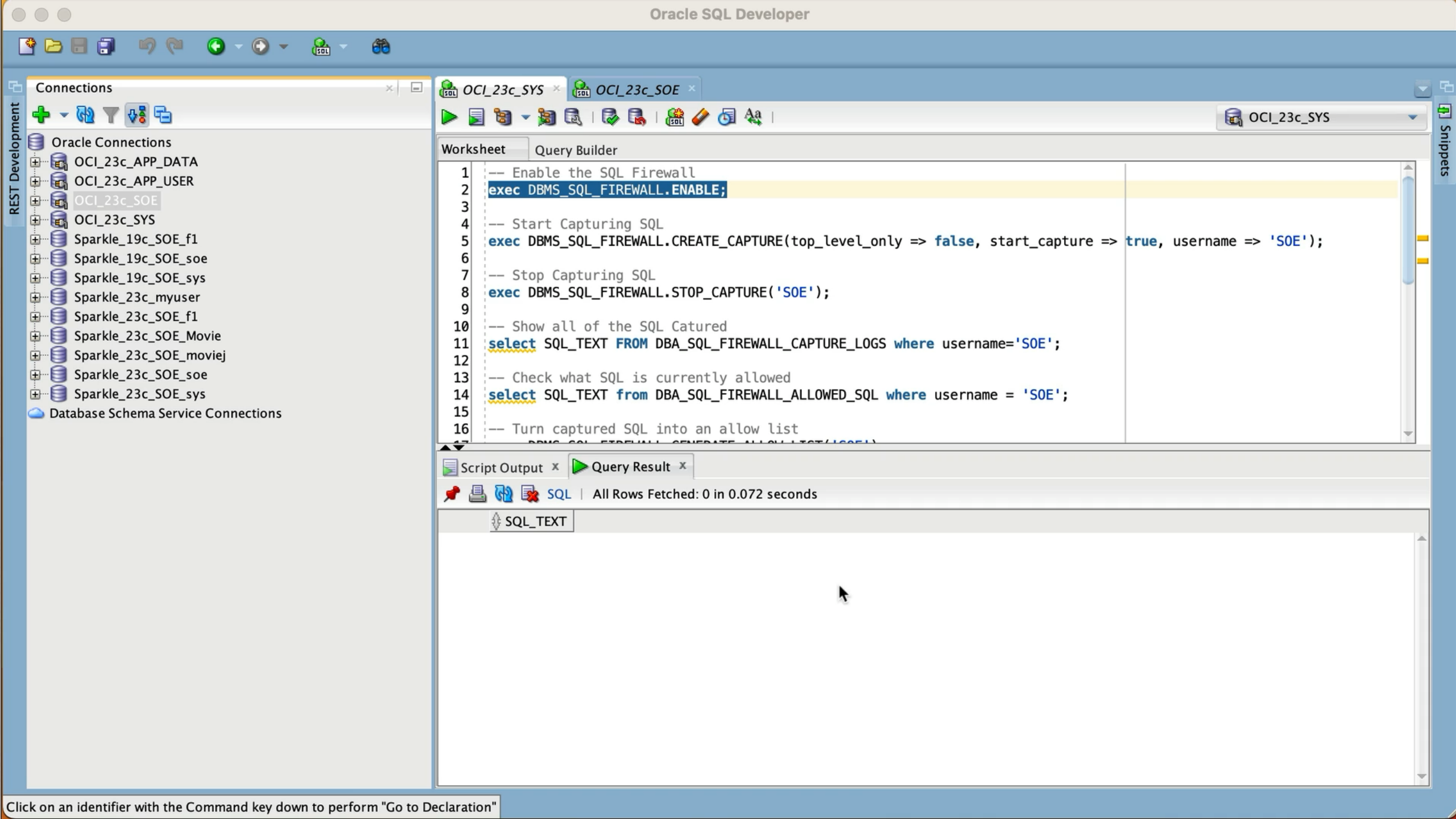


- SQL 防火墙**内置**于数据库内核中，确保它**无法被绕过**
- Oracle SQL 防火墙通过监视并阻止 **“非法 SQL”** 避免常规的数据库攻击以及为 SQL 注入攻击提供防护

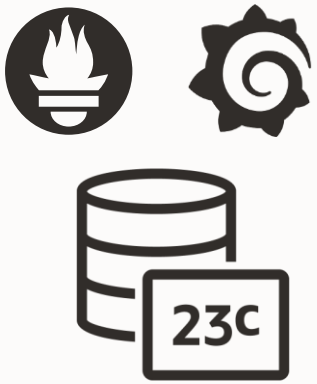
SQL 防火墙

演示

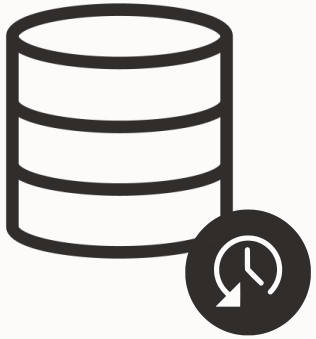



Oracle SQL Developer
Connections
Oracle Connections
OCI_23c_APP_DATA
OCI_23c_APP_USER
OCI_23c_SOE
OCI_23c_SYS
Sparkle_19c_SOE_f1
Sparkle_19c_SOE_soe
Sparkle_19c_SOE_sys
Sparkle_23c_myuser
Sparkle_23c_SOE_f1
Sparkle_23c_SOE_Movie
Sparkle_23c_SOE_moviej
Sparkle_23c_SOE_soe
Sparkle_23c_SOE_sys
Database Schema Service Connections
OCI_23c_SYS
OCI_23c_SOE
OCI_23c_SYS
Worksheet
Query Builder
1 -- Enable the SQL Firewall
2 exec DBMS_SQL_FIREWALL.ENABLE;
3
4 -- Start Capturing SQL
5 exec DBMS_SQL_FIREWALL.CREATE_CAPTURE(top_level_only => false, start_capture => true, username => 'SOE');
6
7 -- Stop Capturing SQL
8 exec DBMS_SQL_FIREWALL.STOP_CAPTURE('SOE');
9
10 -- Show all of the SQL Catured
11 select SQL_TEXT FROM DBA_SQL_FIREWALL_CAPTURE_LOGS where username='SOE';
12
13 -- Check what SQL is currently allowed
14 select SQL_TEXT from DBA_SQL_FIREWALL_ALLOWED_SQL where username = 'SOE';
15
16 -- Turn captured SQL into an allow list
Script Output
Query Result
SQL | All Rows Fetched: 0 in 0.072 seconds
SQL_TEXT
REST Development
Snippets
Click on an identifier with the Command key down to perform "Go to Declaration"

新特性 | Oracle 23c (23.3) 管理和问题诊断

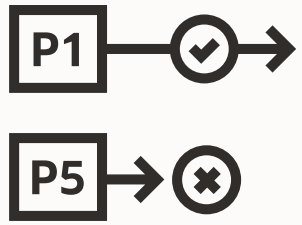


Prometheus/ Grafana 管理界面



备份的增强

出错信息和日志

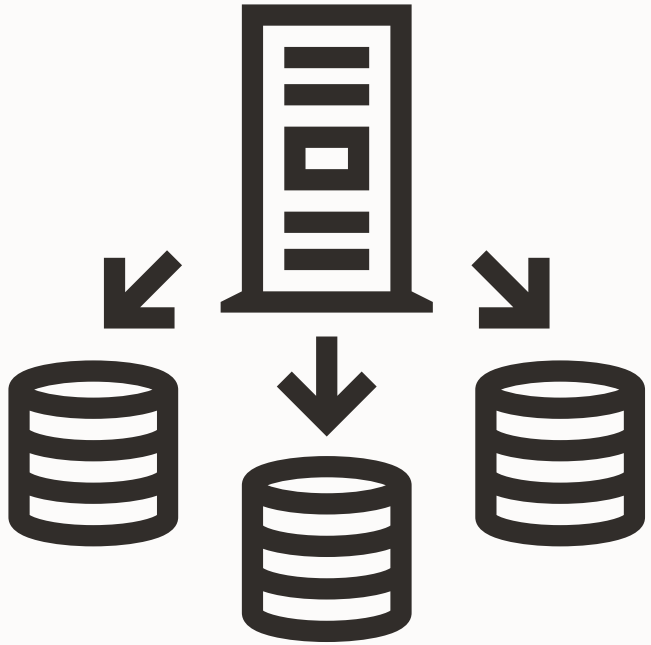


事务的优先级管理

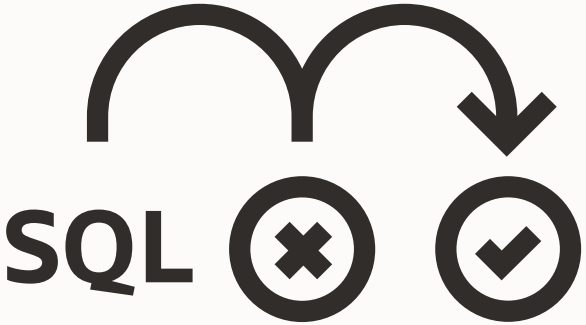
更多, 几百个功能增强

- 自动化事务隔离
- 新的OCI APIs
- 为特定时段指定 events 参数
- In memory 诊断增强
- SQLHC 和 SQLT 诊断工具合并
- 对[g]v$ 视图的增强以帮助诊断
- 为 hanganalyze dump 增加了新的功能
- 对快照过旧出错信息的进一步细化
- RAC 相关的功能增强以帮助诊断问题
- 透明数据库加密的诊断增强
- 重用被删除的 con#
- SMON进程的性能提升和稳定性提高
- PDB Open时的扩展能力增强
- 用于全局锁的实例间通信的增强
- 对 RDBMS 中过多Trace的改进
- 新的 RMAN 进度报告
- 跨多个PDB的SQL Monitor 增强
- SQL优化器的增强
- 集群运行状况监视器提高了可诊断性
- 新的 GoldenGate 性能相关的统计数据
- ASM Scrub 增强
- 增强的系统挂起自动解决和诊断
- SQL 异常的诊断的修复
- CRS 资源的自动优化
- 对加密数据库损坏的保护增强
- redo dumps 的增强
- RDBMS 总字典的健康检查增强
- 更快地定位损坏对象
- 后台工作进程的增强
- 主要块更改跟踪 (BCT) 的改进
- … 还有更多!

新特性 | Oracle 23c (23.3) 可靠性和可扩展性

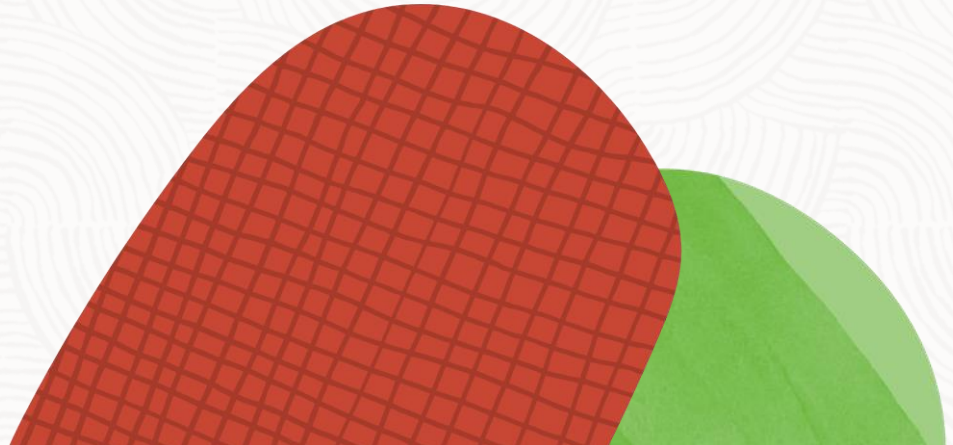


基于Raft复制协议的全球分布式数据库



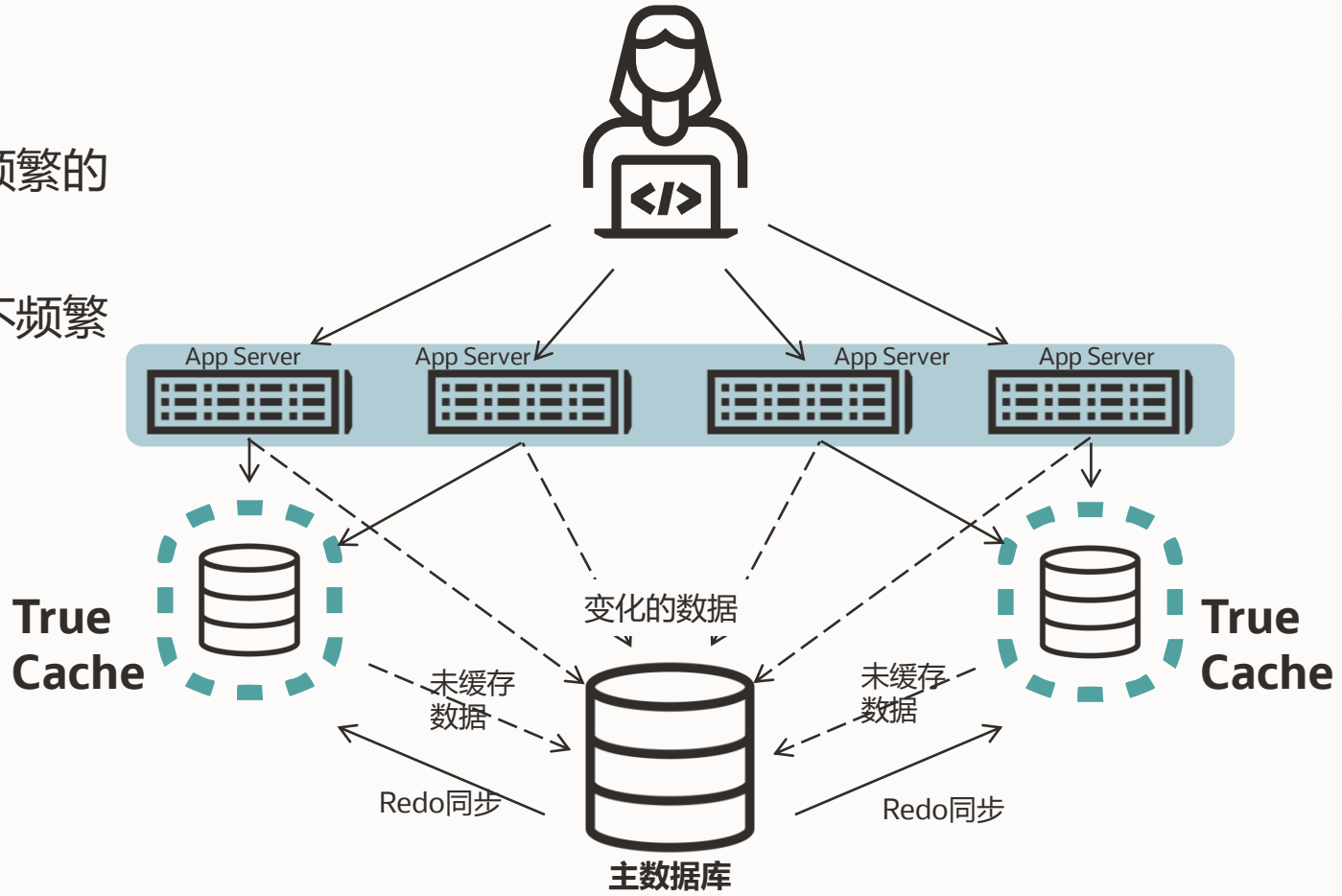
实时的SQL 执行计划管理

Oracle 23ᶜ, 即将引入的新特性



即将发布 | Oracle 23c True Cache

- 实线表示相对较频繁的操作
- 虚线表示相对较不频繁的操作



应用连接到True Cache 完成SQL 查询

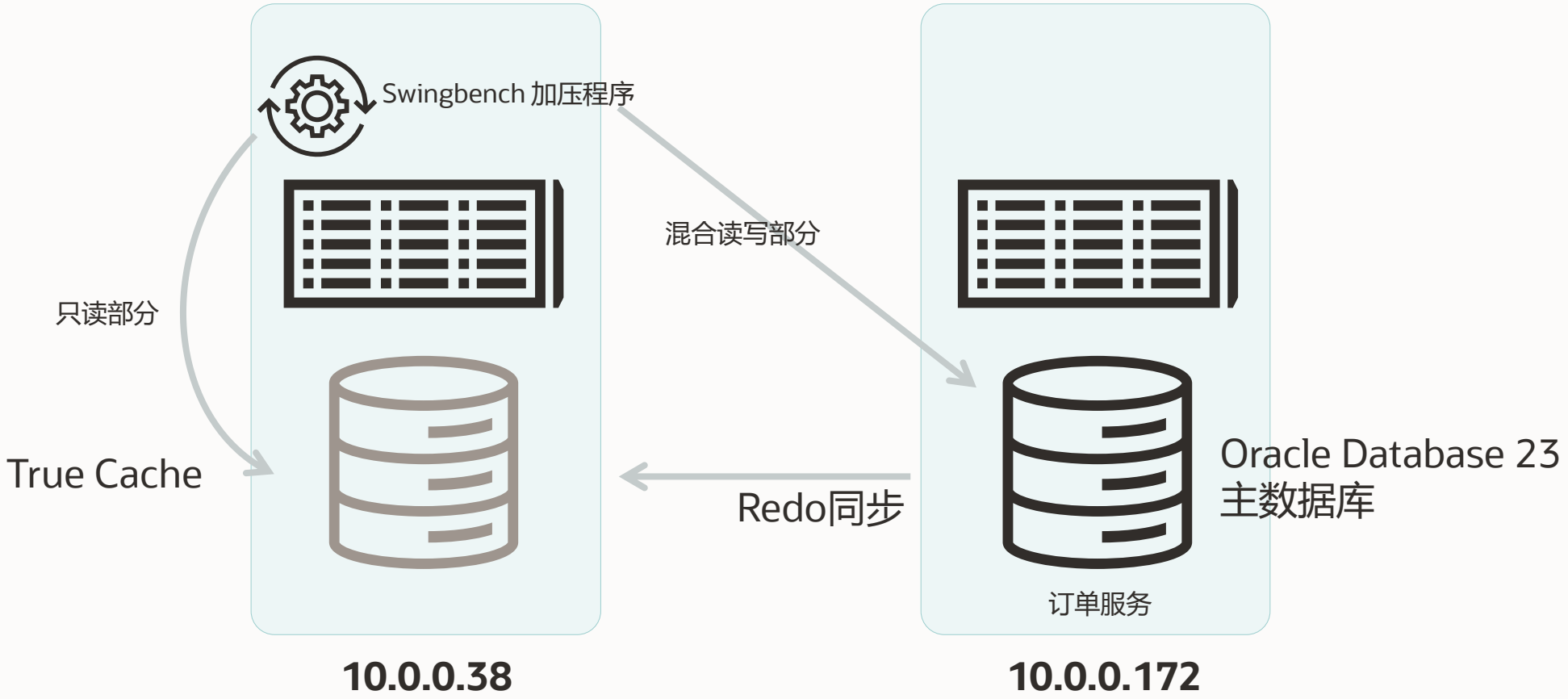
True Cache 是纯内存的SQL缓存，其数据严格一致，且由Oracle自动管理

True Cache

演示

True Cache 演示

本演示为一个订单处理应用，基于Oracle 23c 数据库。如果我们在代码中说明数据只读，则可以通过True Cache来响应请求。True Cache 自动与主数据库中的数据保持同步。受益于True Cache和应用程序部署在一起带来的低延时，应用程序的性能得到较大提升

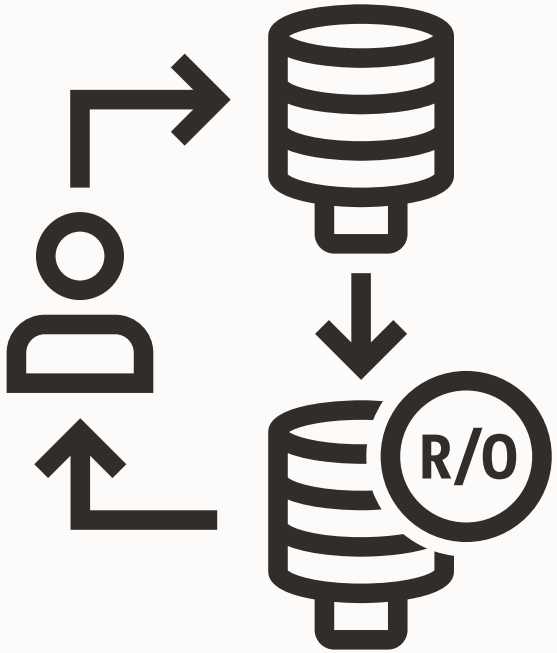


dgiles — opc@doms-server:/home/opc/swingbench/bin — ssh opc@132.145.16.179 — 127×33

~ — opc@doms-server:/home/opc/swingbench/bin — ssh opc@132.145.16.179 | ./dbtimemonitor | java ◂ dbtimemonitor -c truecache.xml -debug ...

```
(venv38) [opc@doms-server bin]$ ./charbench -cs //10.0.0.172/oe oe -p soe  -uc 64 -rt 0:01 -c ../configs/SOE_Client_Side_TrueCache.xml -mr -intermin 0 -intermax 0 -min 0 -max 0 -v users,errs,trem,tpm,tps,vresp -en SQ,WQ,WA
```

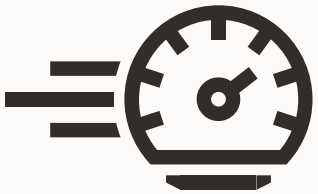
即将发布 | Oracle 23c 高可用性和高可扩展性



PDB 级别的只读备库

无需停库的滚动补丁

性能方面的增强



向量化的查询处理: 在一些 TPC-H 或基于星型模型（SSB）的性能测试中有 **1.2 – 2** 倍的提升

OCI 管道化支持 : SQL*Net 能够有更高效的 CPU使用

RAC环境中 Sequences的顺序生成: 实现 **2** 倍 以上的 insert性能提升

窗口函数: 并行处理效率更高, 对数据分布不均匀的处理更加科学。性能提升 **50% 到 1000%**

OFS: 并行文件系统操作: DBMS_FS 可通过多线程来完成并行操作，在 mount, create, delete 都实现 性能提升

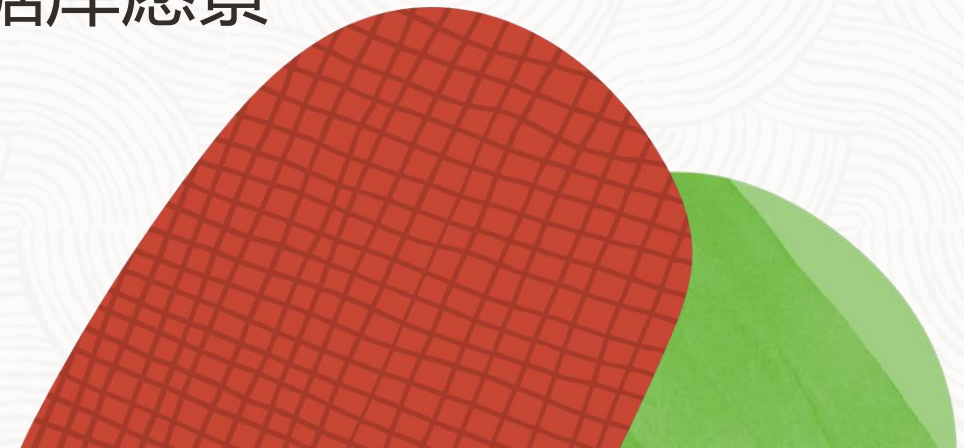
In-Memory 增强 : 多个 bit 的布隆过滤使性能提升 **50%** 以上，以及 SQL 执行过程中的其它增强

以及更多，更多...

Make it easy to generate and run modern apps
and analytics for all use cases at any scale

让生成和运行
任何规模的任何应用和分析
都非常容易

基于生成式 AI 的
Oracle 数据库愿景



生成式AI的关键组件

大语言模型(LLM)

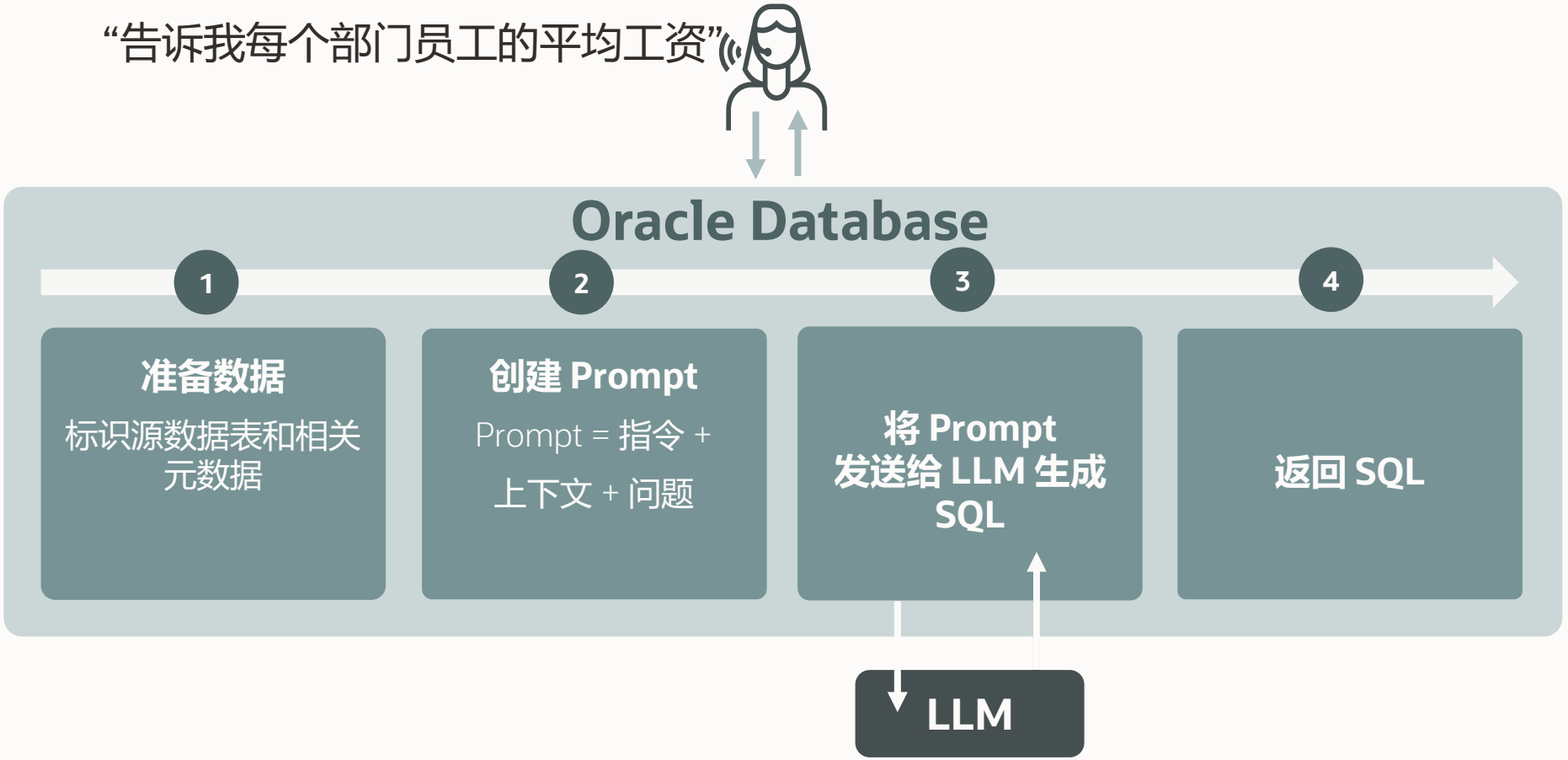
- 旨在理解和生成人类语言的人工智能技术
- 经过大量文本数据的训练
- 掌握单词、句子甚至整个文档之间的关系
- 接受自然语言输入
- 生成连贯且适合上下文的响应、故事或其他形式的文本

向量数据库

- 旨在处理和有效搜索可用于表示文档、图像、视频和音频的向量的数据库类型
- 旨在以允许快速、准确的相似性搜索的方式存储和索引向量
- 适用于推荐系统、图像和视频分析、自然语言处理等任务

即将发布 | 借助LLM从自然语言生成SQL语句

生成式 AI 应用案例



向量用来表示非结构化数据的语义

例如图像，文档，视频，等等.

向量是一个数字序列，我们称之为维度，用来表达数据的关键特征

向量 表述的是数据的 **语义内涵** 而不是底层的文字或像素

向量由深度学习的嵌入式模型来生成

向量检索

这种技术将人们的问题映射成数据库内可检索的数据

向量检索的商业应用场景

相似性搜索

为工作岗位寻找候选人
图像和视频检索
识别相似症状的病人
法律电子取证

基于内容的过滤

个性化推荐
基于图像的搜索

自然语言处理

文本分类识别
SQL 生成

数据分析

异常情形识别
特征分析

计算机视觉

人脸识别
生物特征识别
对象检测

生物研究

DNA相似性分析
分子结构检索

地理信息系统

空间信息分析
地图渲染

工业应用

质量控制
预测性维护
机械故障侦测

对 LLM 的担忧

Hallucinations幻觉

听起来似是而非的虚假信息

肯定的响应无法通过训练数据证明

Security安全

模型访问控制：知识产权盗窃或模型操纵

窃取数据或行为不可预测的网络安全风险

错误信息和深度伪造

Memory记忆

基于某个时间点的数据快照训练

不了解私有或企业数据

AI伦理

偏见和公平

透明度和隐私

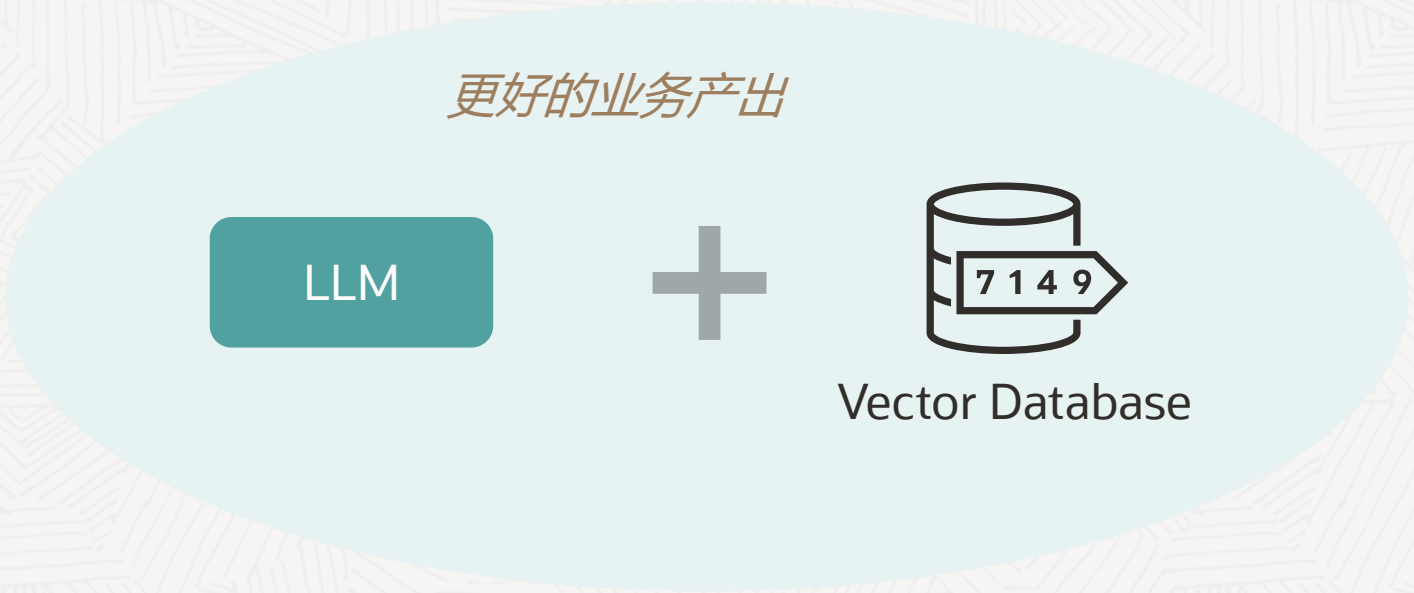
问责

向量数据库和LLM的角色定位

解决LLM响应中固有的幻觉和记忆问题
通过添加企业特定内容来增强提示以生成更好的响应
通过使用最相关的内容避免超出LLM标记限制

向量数据库和LLM的角色定位

避免使用敏感客户数据进行LLM的训练和微调
比微调LLM更便宜，而微调LLM的更新成本可能较高
实时更新的知识库
缓存以前的LLM提示/响应以提高性能并降低成本



Oracle AI 向量检索简介

Oracle 23c 中引入的一组新功能

旨在简单易用且易于理解

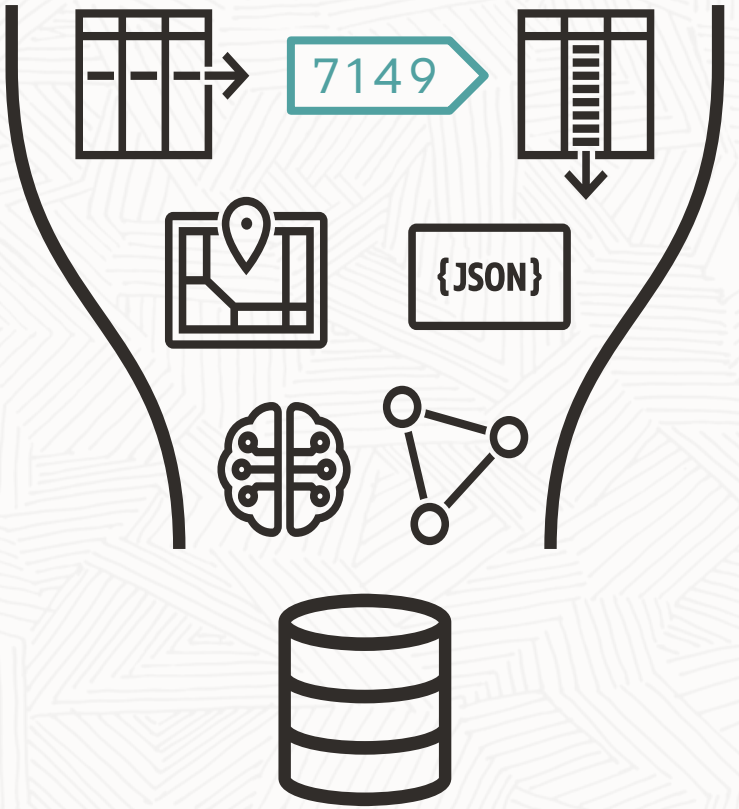
新的VECTOR数据类型用于存储向量Embedding

新的SQL语法和函数轻松表达相似性搜索

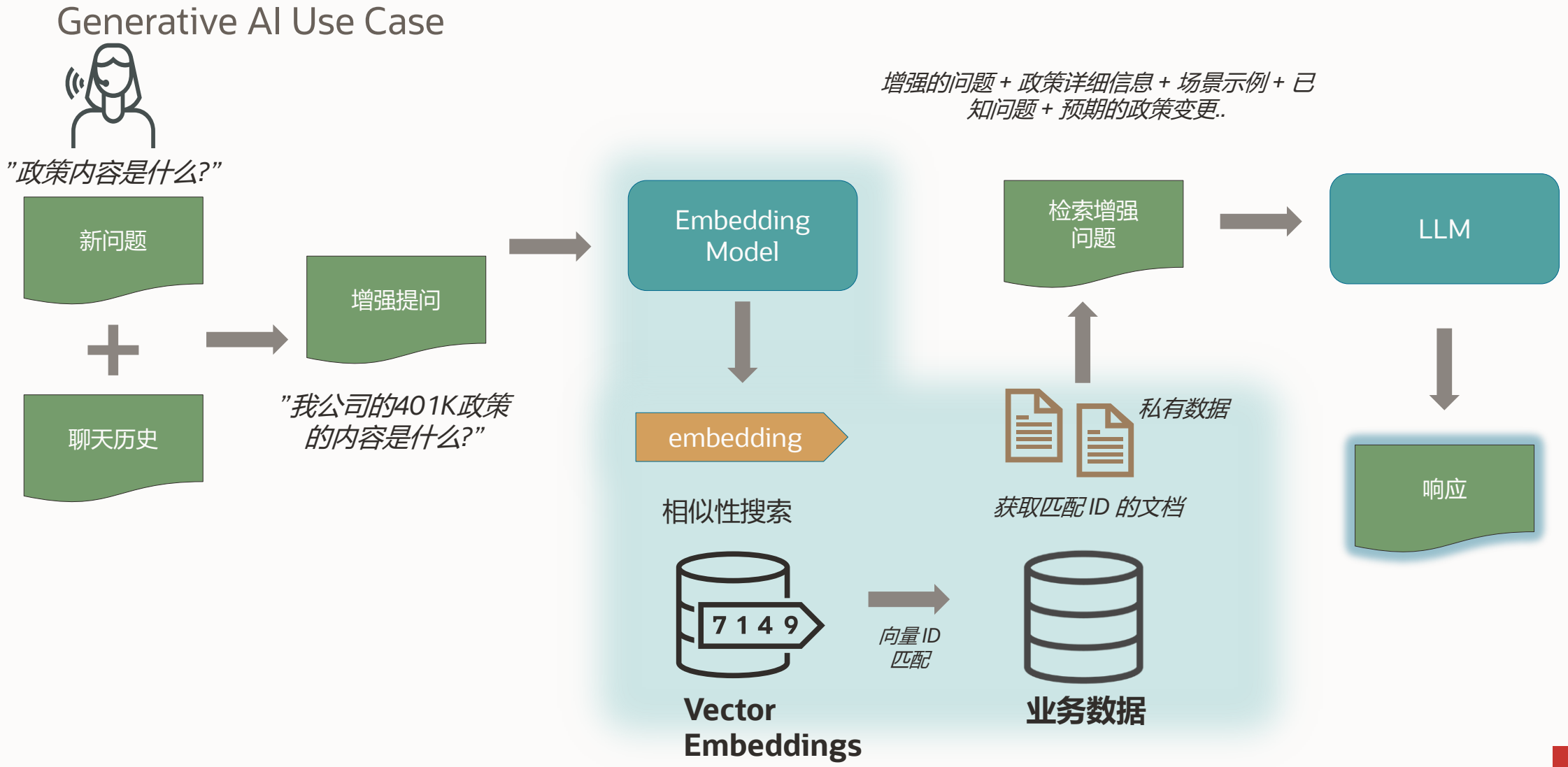
新的近似搜索索引，经过封装和调优，以实现高性能和质量

在查询中执行与业务数据（例如客户和产品）相关的向量检索

在同一数据库中处理向量和其他工作负载



具有“企业知识”的基于LLM的聊天机器人



将 AI 向量搜索与客户和产品的业务数据相结合的检索

将客户数据、产品数据和人工智能搜索结合在 5 行 SQL 代码中！

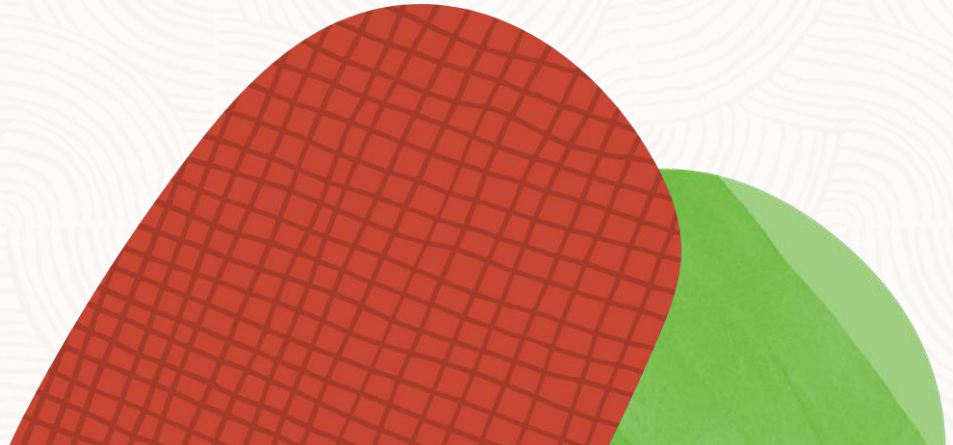
单一集成解决方案，所有数据完全一致

查找与此图片类似且符合客户首选城市和预算的房屋


```
SELECT …
FROM   house_for_sale
WHERE  price <= (SELECT budget      FROM customer …)
AND    city  in (SELECT search_city FROM customer …)
ORDER BY vector_distance(house_vector, :input_vector);
```

升级到Oracle 23c

让开发变简单



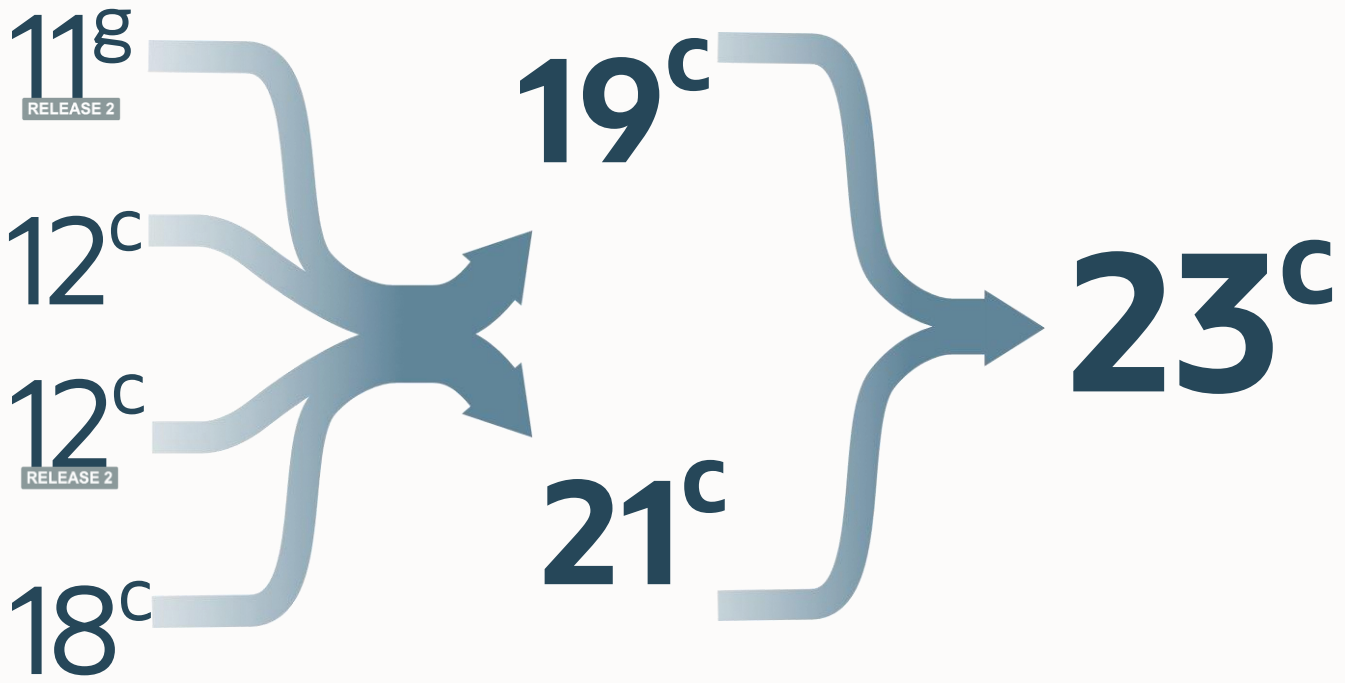
Oracle 23c 版本发布路线图

Oracle 23c (23.3) 基本服务

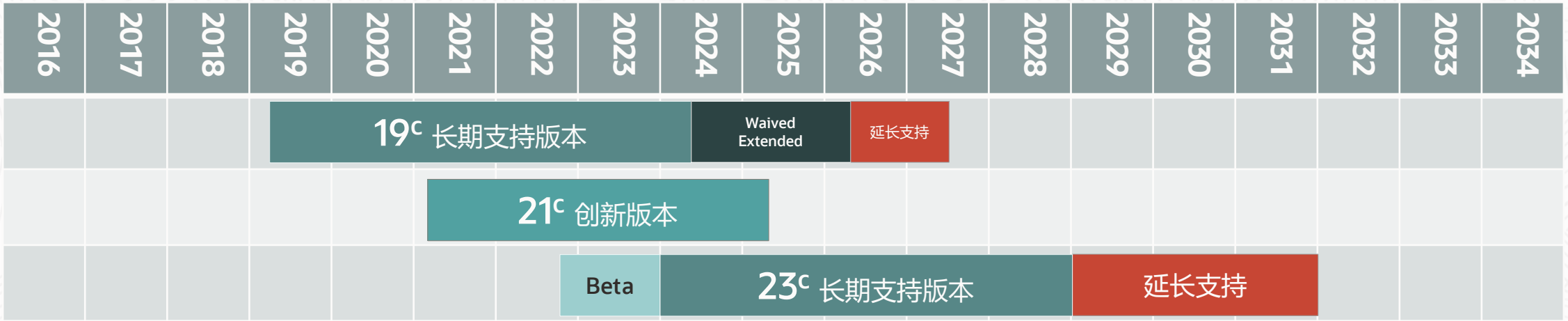
- 不支持RAC, 不提供标准版
- 2023年9月19日起可用

Oracle 23c (23.3) 开发者免费版

- Oracle Linux 8 / Redhat Linux 8
- 2023年9月19日起可用



各数据库版本的支持时间线



- 创新版本 - 2 年标准支持, 无延长支持
- 长期支持版本 - 5 年标准支持, 3 年延长支持
- 最新信息请查阅MOS文档: Release Schedule of Current Database Releases (Doc ID 742060.1)

谢谢

Checkout Oracle Database 23c
http://www.oracle.com/database/23c