

# Oracle 数据库高性能解决方案之 DB In-Memory

公益讲座11: 00准时开始, 请大家先浏览云技术微信公众号技术文章。资料会在各群同步发布, 已入群客户请勿重复入群!



20-20

数据库和云讲座群



甲骨文云技术公众号



B站专家系列课程



# Oracle 数据库高性能解决方案 之 DB In-Memory

甲骨文技术公益课 - 数据库专场

2023 年 4 月 28 日 11:00

线上直播



# 基于 Oracle 数据库 免费企业数据健康检查

- 及时了解数据库健康状况，发现并解决潜在问题
- 维护数据库系统良好状态，保护数据资产的安全
- 提升数据库性能、稳定性和安全性，降低业务风险

免费咨询热线：

**400-699-8888**

\* 活动最终解释权归甲骨文公司所有

# 内存计算技术的定义和市场趋势

[https://en.wikipedia.org/wiki/In-memory\\_processing](https://en.wikipedia.org/wiki/In-memory_processing)

内存计算的定义（维基百科）：内存计算作为计算机科学中的一种新兴技术，主要用于处理存储在内存中的数据。目前访问和处理磁盘存储中的数据已经不能满足现代商业智能BI的需求，而将数据放入内存后，可以更快地访问数据，实现实时分析数据，更快的支撑业务报表和决策。

## In-memory Computing Market Size

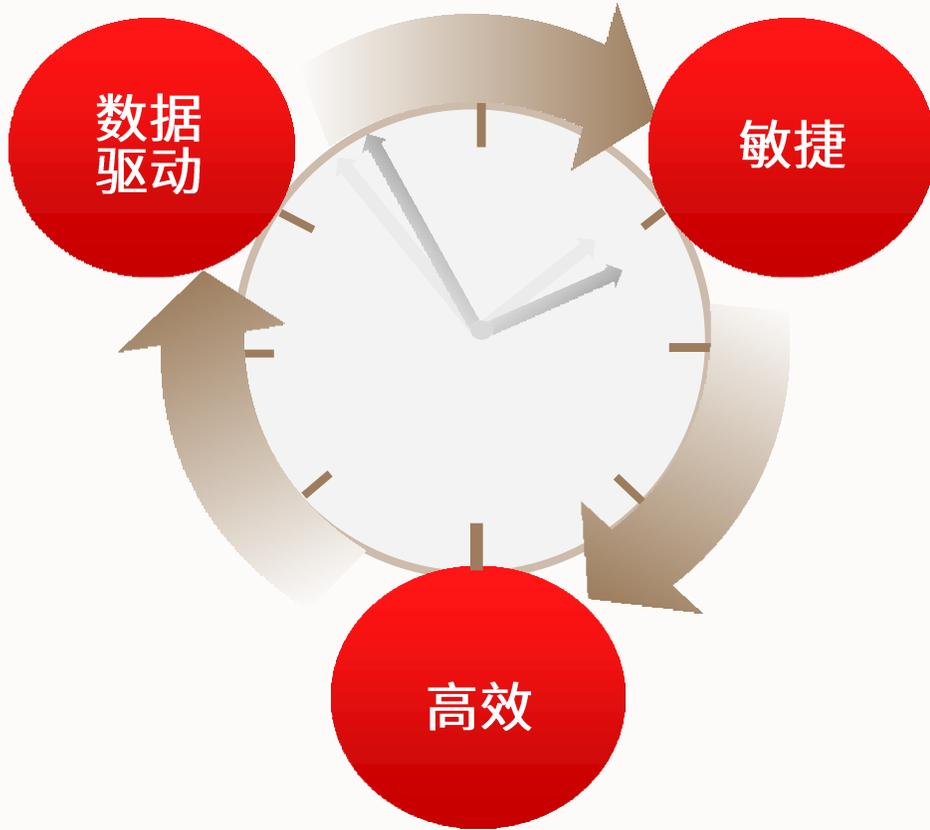


- ✓ 调研机构Mordor Intelligence预测，未来2023-2028年，内存计算市场的复合年增长率为25.37%，处于快速上升阶段；
- ✓ 现代企业面临激烈的市场竞争，决策者已不能够接受，需要在数据产生后，等候几小时，甚至超过一天时间，才能海量数据中提取出支持经营决策关键信息；
- ✓ 因此企业不但数据量在快速增长，对大数据更快处理和的需求也不断增长，这种对更好、更快的分析性能的永无止境的需求也推动了内存计算技术的发展。

[In-memory Computing Market Size & Share Analysis - Industry Research Report - Growth Trends \(mordorintelligence.com\)](https://www.mordorintelligence.com/industry-reports/in-memory-computing-market-size-share-analysis)



# 内存计算技术给不同行业带来的收益

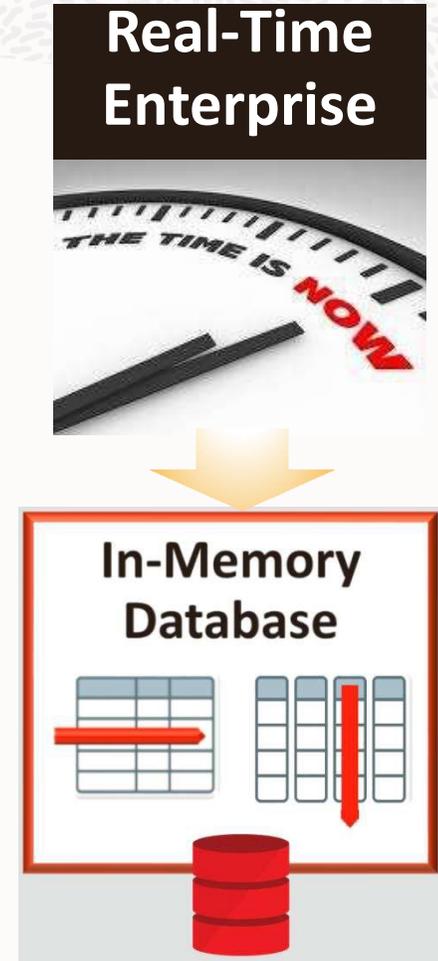


- **保险业**，通过实时定价分析，改进投资组合并降低成本；
- **零售业**，通过基于位置的实时分析，自动向客户发送个性化的移动优惠券；
- **制造业**，使用实时分析来监控生产质量、调整装配参数；
- **金融服务**，进行实时的、跨渠道的风险/欺诈分析，不仅限于欺诈事件发生后；
- **电信和宽带产商**，使用实时拥塞度量进行网络优化

# In-Memory内存计算是支持企业进行实时分析的必选技术

## IT系统已经进入内存计算的时代

- 企业必须具备“实时分析”的能力
  - ✓ In-Memory内存计算技术对实时数据分析非常重要,可实现分析业务10-100倍的性能提升;
  - ✓ 数据一产生即可进行分析,不必等待数据搬运到专门的分析系统,导致决策时间推迟。
- 现代硬件设备的内存规模有利于In-Memory技术的广泛推广
  - ✓ 以Oracle X9M-2为例: 单个计算节点就具备最大 2TB DRAM内存;
  - ✓ In-Memory内存计算技术已经是一项成熟技术。



# Oracle几种In-Memory技术不同使用层面

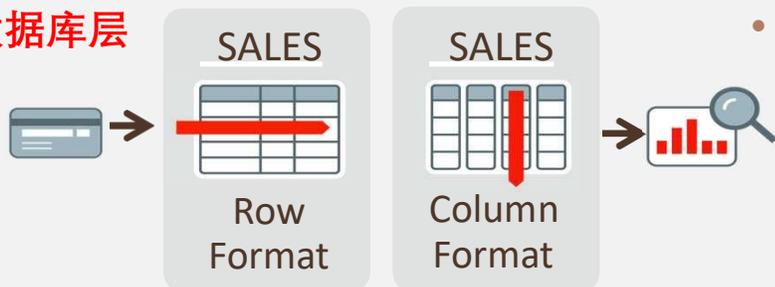
## 应用层



### • TimesTen Application-Tier Database Cache内存缓存技术

- 具有超高并发、低延迟需求的热数据查询应用
- 微秒级响应时间
- 作为Oracle数据库的高性能内存缓存

## 数据库层



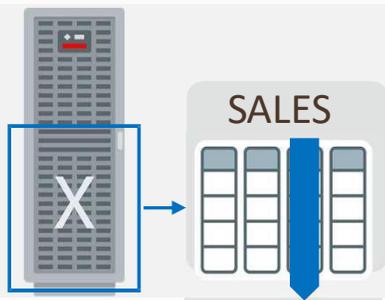
### • Database In-Memory选项

- 数据库内存中行列双格式
- 数十亿行/秒分析处理性能
- OLTP和OLAP混合负载下，提升2-3倍性能

### • TimesTen In-Memory Database

- 关键OLTP应用
- 超高并发、低延迟的交易事务
- 可完全独立部署，也可从Oracle数据库同步部分数据

## 存储层



### • Exadata 存储层的In-Memory技术

- Exadata闪存缓存中的In-memory数据格式
- 存储节点的智能扫描速度提升5-10倍
- In-memory列式格式的高压缩特性，使闪存缓存的有效数据容量提升15倍

# 什么是Database In-Memory?



# 什么是Database In-Memory

您最喜欢的数据格式是什么?

## 行格式

OLTP 交易在行格式上运行更快



## 列格式

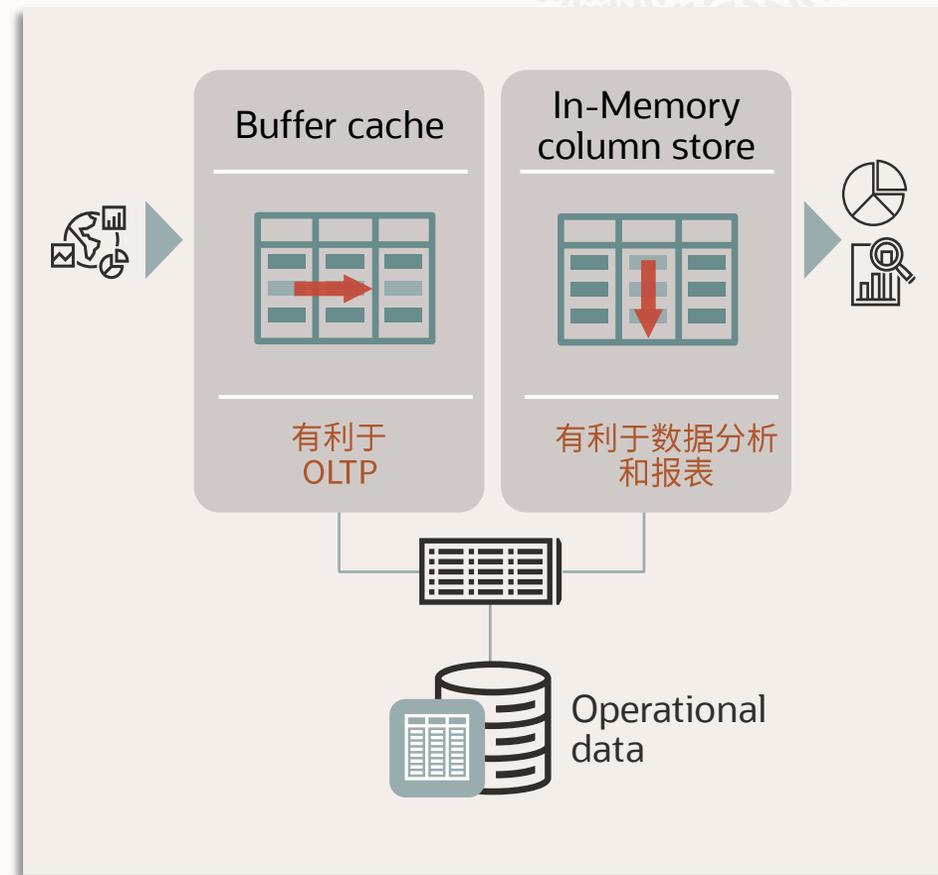
分析在列格式上运行更快



(+)

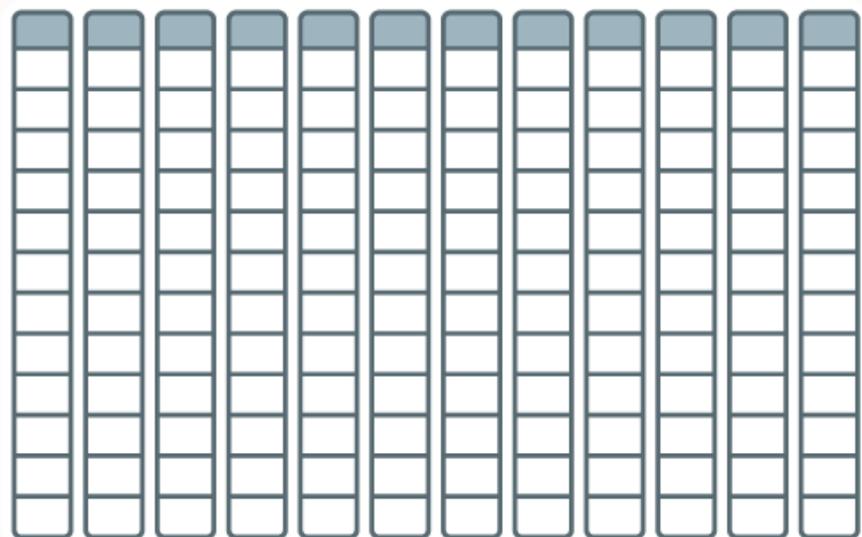
## 双格式

对两者最佳  
快速分析  
快速OLTP  
(不需要分析索引)



# Oracle In-Memory 列技术

内存纯列格式



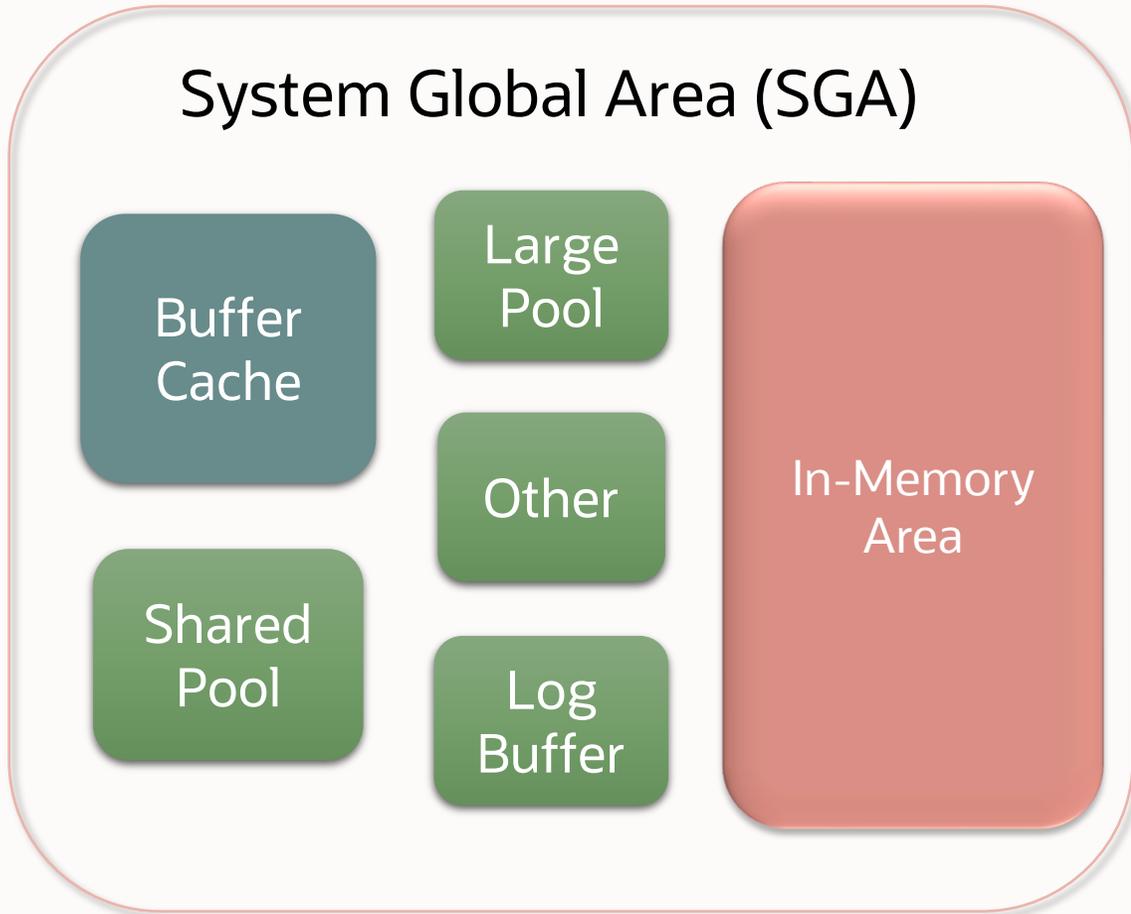
SALES

SALES		



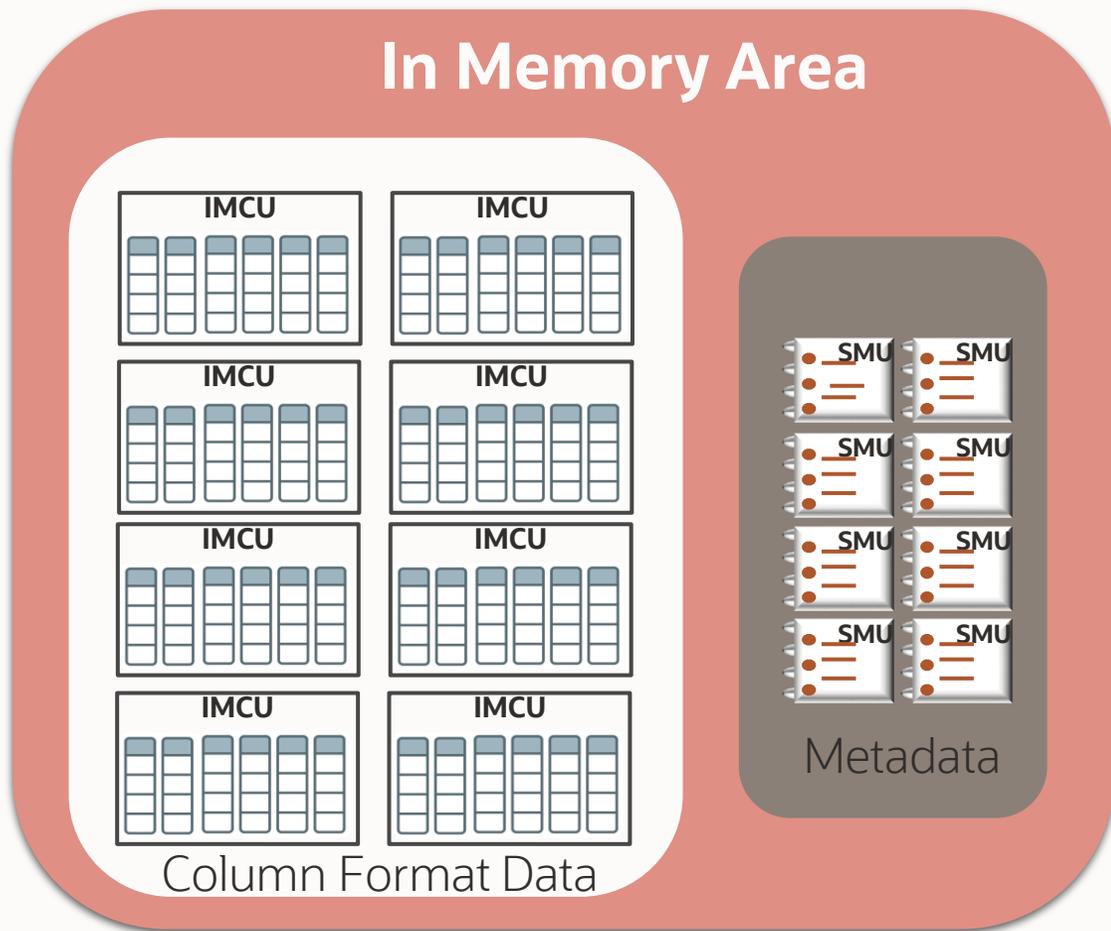
- 内存纯列格式
  - 无需持久化，开销很小
  - 不修改磁盘中的格式
  - 在所有平台中提供
- 可在表空间，表，分区和子分区，物化视图级别启用(仅缓存需要的数据)
  - 可以指定部分列
  - 内存区域大小由inmemory\_size 参数控制
- 2 倍至 20 倍压缩率

# In-Memory内存结构: SGA静态区域



- 内存中以新的列格式存放数据
- 由 INMEMORY\_SIZE 参数控制
  - 最小大小为 100MB
- 可以在数据库运行时动态调整大小 (12.2)
- SGA\_TARGET 必须足够大以容纳 In-Memory 区域

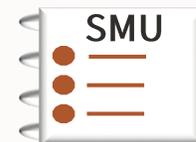
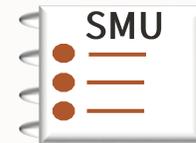
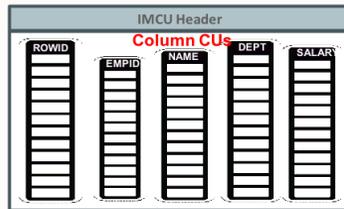
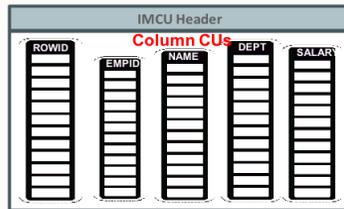
# In-Memory 区域的构成



- 包含两个子池:
  - IMCU pool: 存储内存压缩数据单元 (IMCUs)
  - SMU pool: 存储快照元数据单元 (SMUs)
- IMCUs包含列式数据
- SMUs包含元数据和交易信息

# In-Memory 对象组成

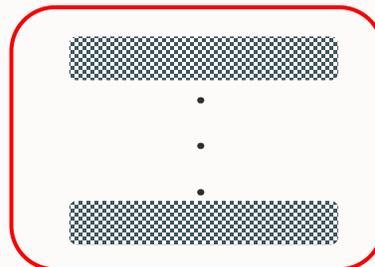
Employee Table		



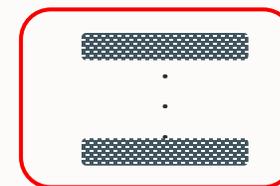
一个IMCU/SMU  
组成一个或多个  
1MB/64KB 内  
存段

一个对象在In-Memory中存储是采用压缩的方式，内存中的对象由一个或多个 IMCU/SMU

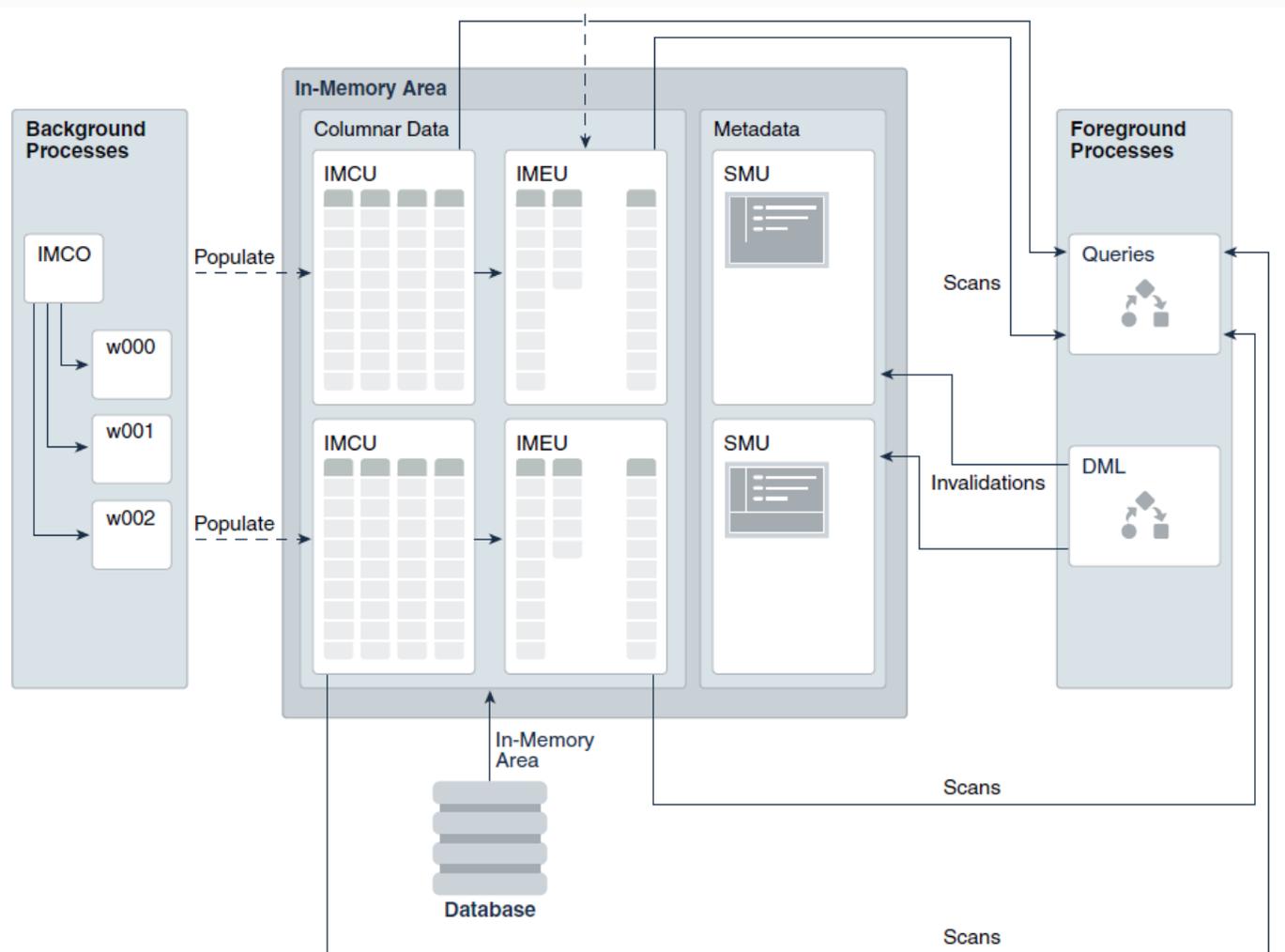
In-Memory Area  
(1MB Pool)



In-Memory Area  
(64 KB Pool)



# Oracle Database In-Memory架构



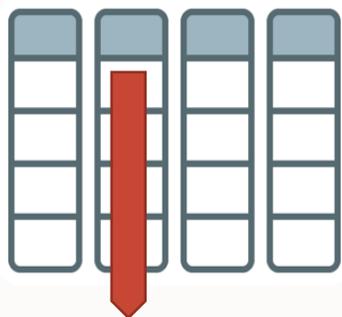
- In-Memory Area是SGA的一部分，由参数INMEMORY\_SIZE设定，包括IMCU(列式数据)、SMU(元数据)和IMEU(表达式)组成。
- 行式数据在实例启动或第一次数据访问时发布到内存
- 数据更新时，会在SMU中做标记，这
- 部分数据的读取会定向到行格式存储
- 当列式“脏数据”积累到一定程度，会进行刷新
- 发布和刷新在后台由IMCO调度，SMCO执行



# In-Memory 技术概述

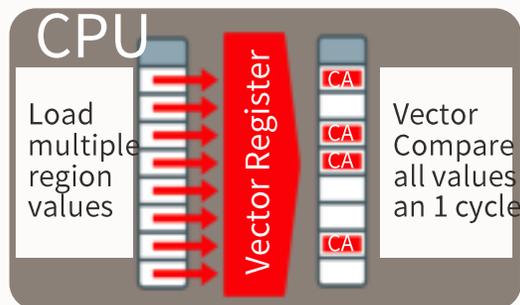
## 极大的加速分析数据的访问能力

### 纯列格式



仅访问你需要的列

### SIMD向量处理



单CPU指令同时处理多列数据

### 存储索引



从列中裁剪任何不必要访问的数据

### 压缩



在压缩格式下扫描 & 过滤数据



# 列格式

在查询时只有写在SQL中的列才被访问

- 只扫描需要的列
  - 不需要读取每个“行”并遍历每个列来查找值
- 21c 中，引入了In-Memory混合扫描

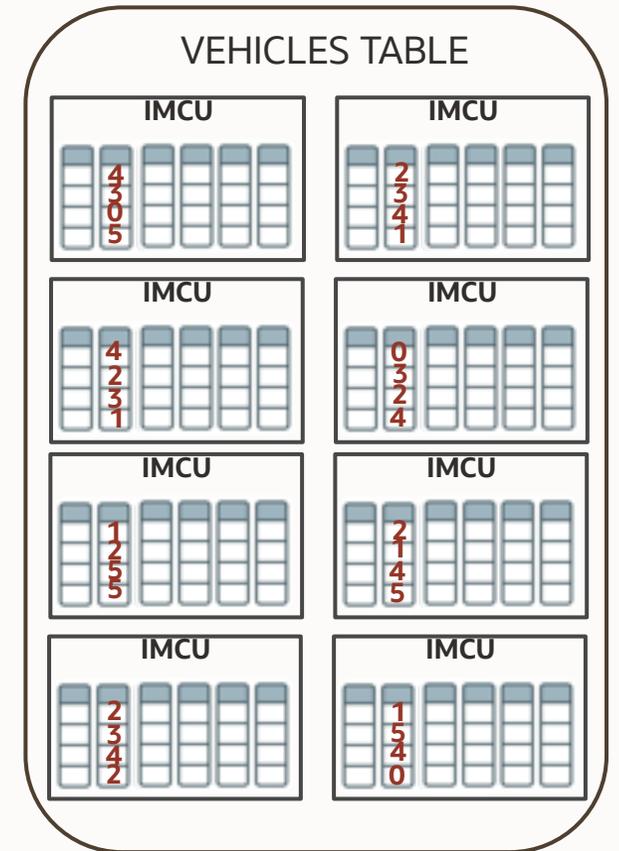


# 压缩

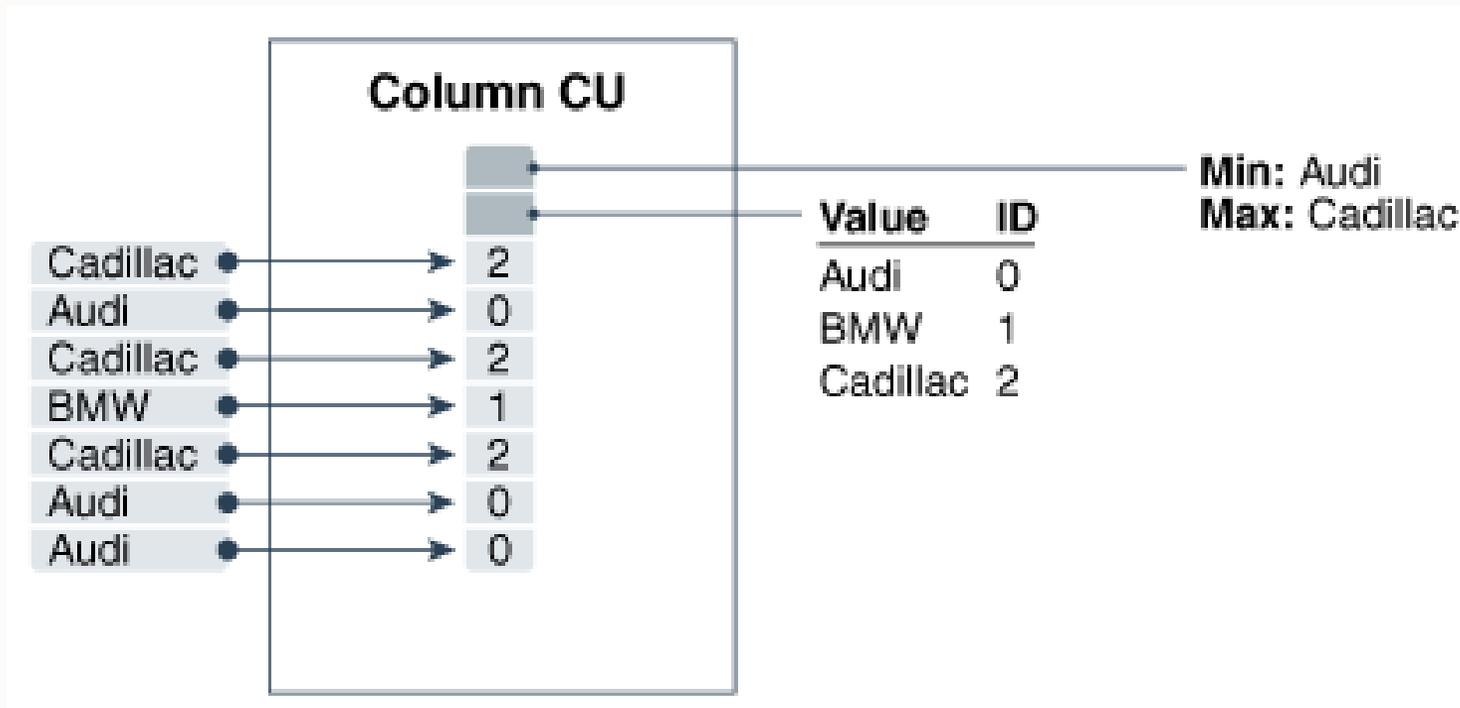
- 多级别的压缩
  - FOR DML
  - FOR QUERY LOW/HIGH
  - FOR CAPACITY LOW/HIGH
- Query Low 和 High使用字典和运行长度编码-直接对压缩数据进行评估
- Capacity Low 和 High都增加了额外的“zip-like”的压缩

## Common Dictionary

NAME	ID
AUDI	0
BMW	1
CADILLAC	2
PORSCHE	3
TESLA	4
VW	5



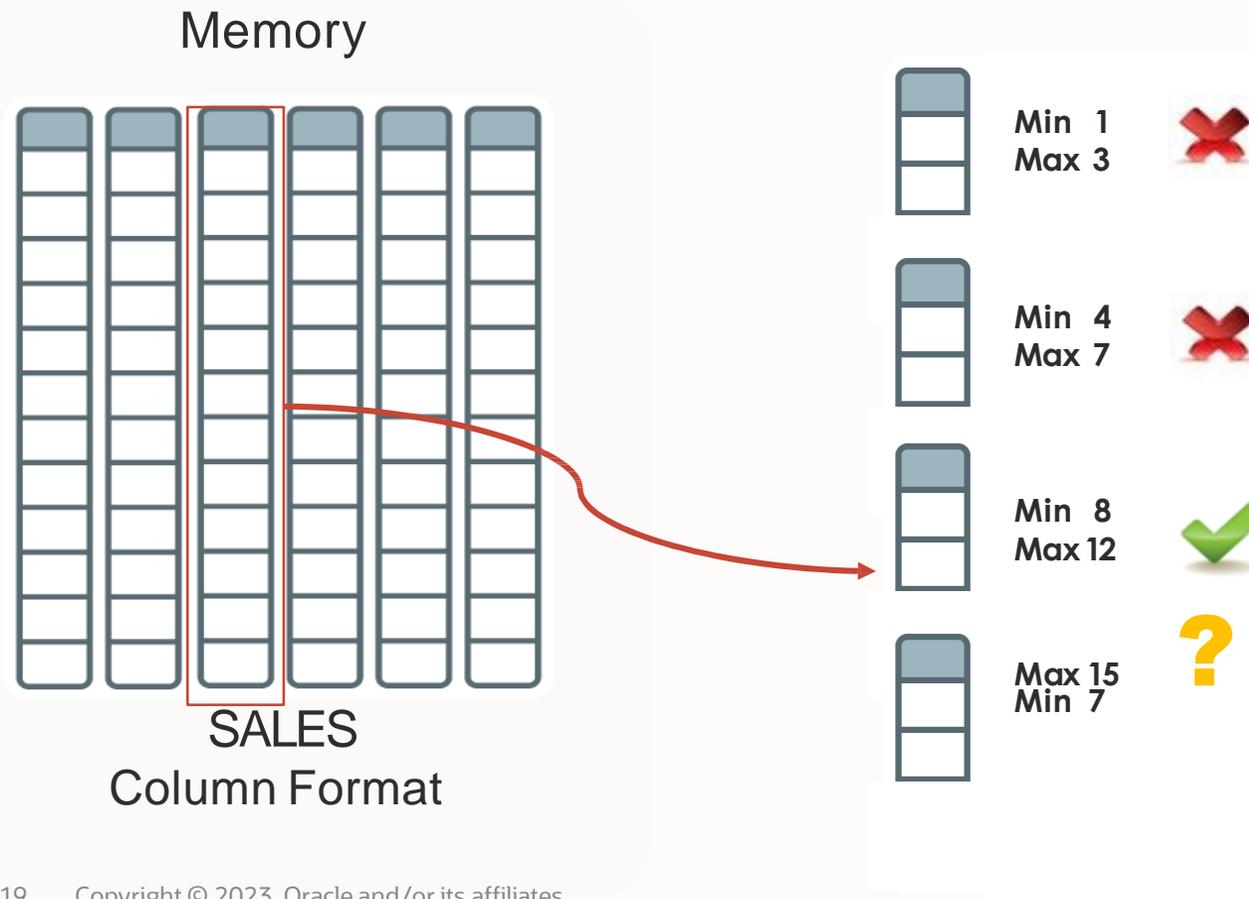
# 直接在压缩数据上扫描和过滤



- 数据以压缩格式发布到内存
- 采用字典编码
  - CU内唯一值的排序列表
  - 列的值被字典ID取代
- 可能会采用其它的压缩格式 (Run Length Encoding, OZIP)
- 压缩格式允许WHERE语句的谓词直接对压缩数据进行评估
- 比buffer cache扫描更少的数据

# In-Memory 存储索引

只查找需要的数据

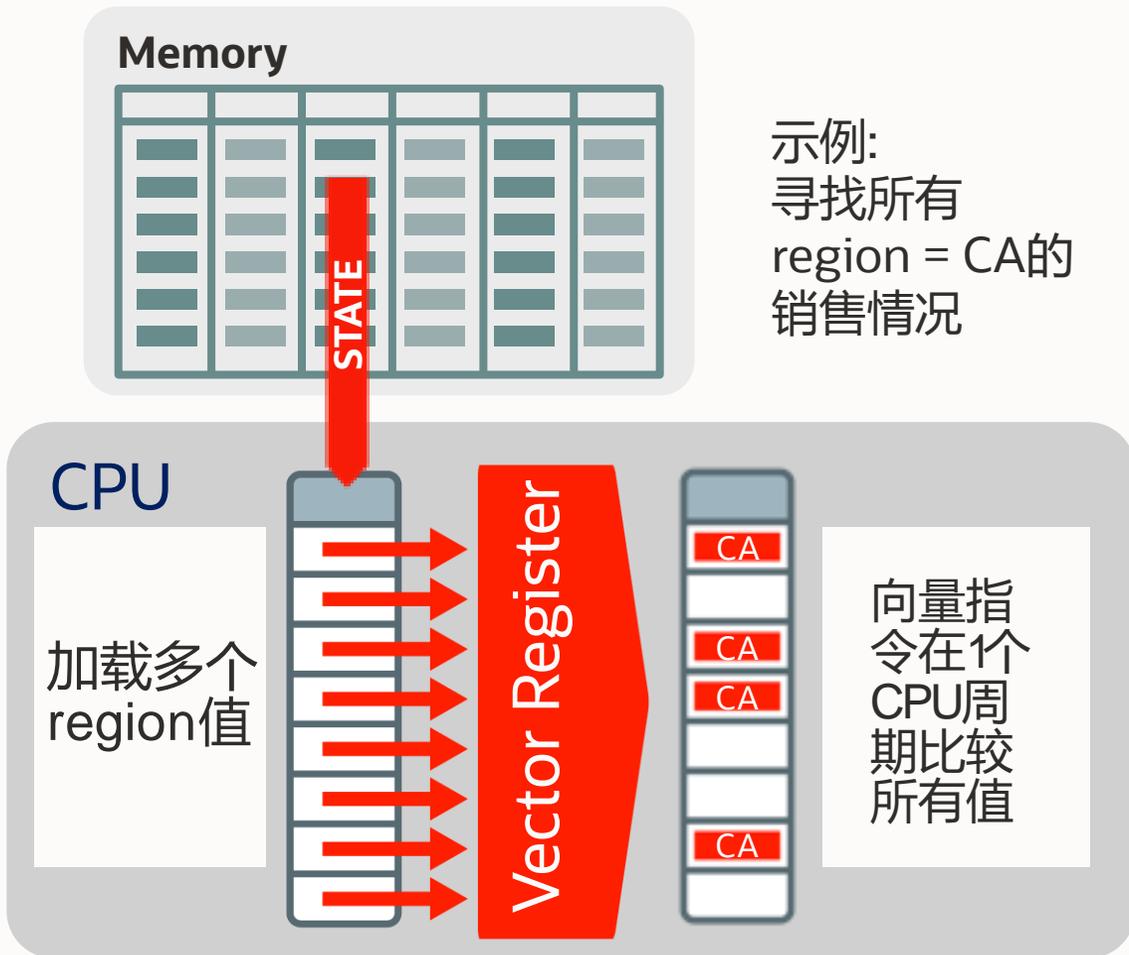


- 例如: 在表sales中查找 a store\_id = 8

- 每一列由多个列单元组成
- 每一个列单元的最小/最大值都被记录在存储索引中
- 存储索引为所有查询语句提供类似分区修剪, 可大幅提高查询的性能。

# In-Memory支持SIMD矢量处理

数据扫描提速了一个数量级



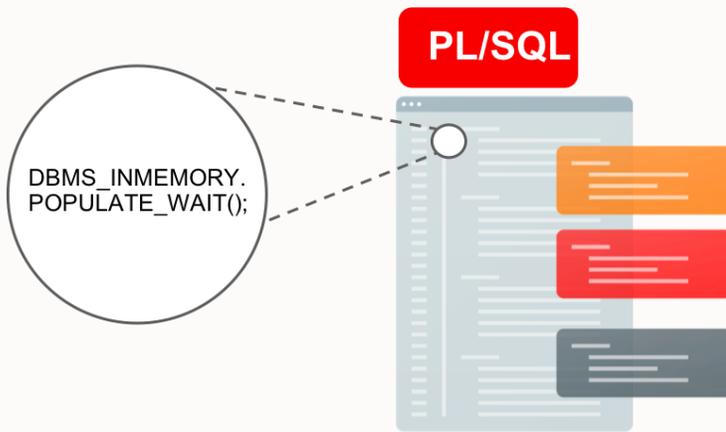
> 10x Faster

- 每个CPU核扫描本地内存中的列
- 使用超快的SIMD 向量指令扫描
  - 最初用于图形处理和科学计算
- 每个CPU Core进行 **数十亿行/秒** 的扫描速率
  - 行格式是 百万/秒

分析型数据扫描提速了一个数量级

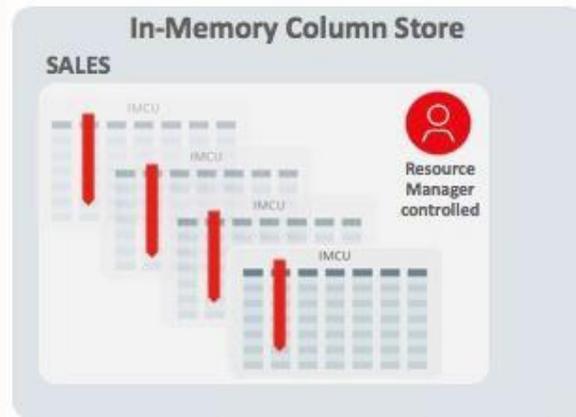
# Database In-Memory 19c 新增功能

## 增强的管理



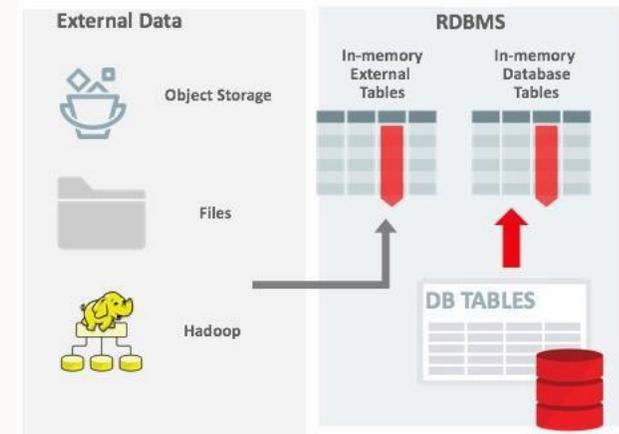
POPULATE\_WAIT 函数  
前台发布并等待发布结束

## Dynamic Scan增强



自动启用  
Resource Manager

## 增强外部表支持

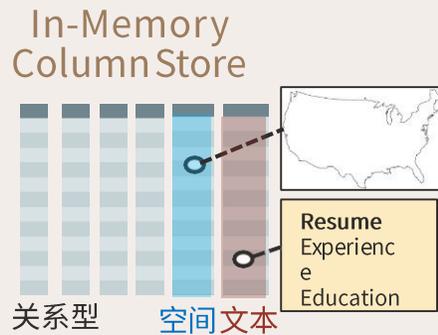


Hive/BigData 支持  
Parallel Query 支持  
混合分区外部表  
Data Guard 多实例 Redo Apply

# Database In-Memory 21c 新增功能

## 融合分析

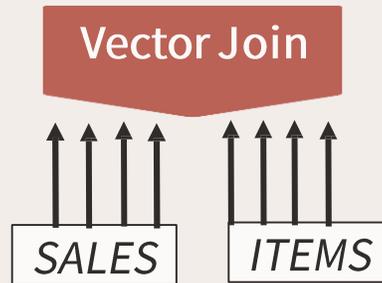
### In-Memory 空间与文本



- 10x 提速 空间分析
- 3x 提速 文本分析

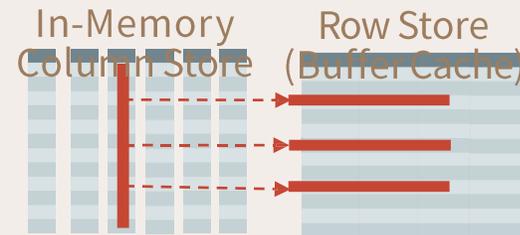
## 性能

### In-Memory Vector Joins



- In-Memory Joins 使用 SIMD vector 指令
- 5-10x 提速

### In-Memory Hybrid Scans

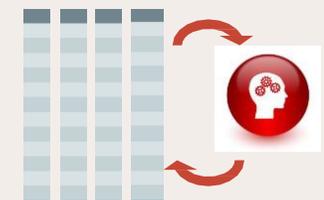


- In-Memory 扫描自动从buffer cache获取缺失数据
- 10x 提速 混合查询

## 自动化

### In-Memory 自治管理

In-Memory  
Column Store

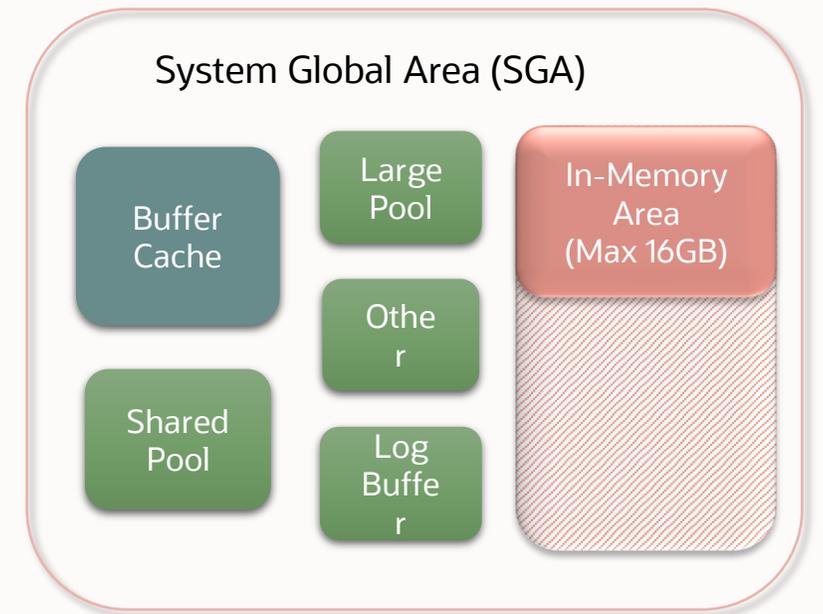


- 自动In-Memory发布 / 清除
- 无需为表定义 INMEMORY属性



# Database In-Memory Base 级别特性

- 客户现在可以使用最多 16GB 的 IM 列存储，而无需许可 Database In-Memory
- 选项此基本级别功能的目的是让客户看到内存中数据库的价值
- 并非所有数据库内存功能都可用于基本级别
- 功能提供 21c，从 19.8 Ru 后可以使用



# Database In-Memory持续创新之旅

21c

- **Self Managing In-Memory**
- In-Memory Spatial Analytics
- In-Memory Full Text Columns
- External Table Enhancements
- Hybrid Scans
- JSON Data Type
- Vector Joins
- Base Level Feature

19c

- Performance
- External Tables: Hive & HDFS
- Memoptimized Rowstore – Fast Ingest

18c

- **Automatic In-Memory**
- In-Memory Dynamic Scans
- In-Memory External tables
- In-Memory Optimized Arithmetic
- Memoptimized Rowstore – Fast Lookup

12.2

- Join Groups
- In-Memory Expressions
- JSON/OSON support
- **Massive capacity - In-Memory on Exadata flash**
- Auto population policies
- Fast-Start
- Active Data Guard

12.1

- Pure In-Memory column format
- Scan & Filter on compressed data
- Fast joins
- Data pruning via storage indexes
- SIMD vector processing
- In-Memory aggregation

21c

Self-Managing, Convergence

19c

Performance, Automation

18c

Performance, Capacity

12.2

12.1

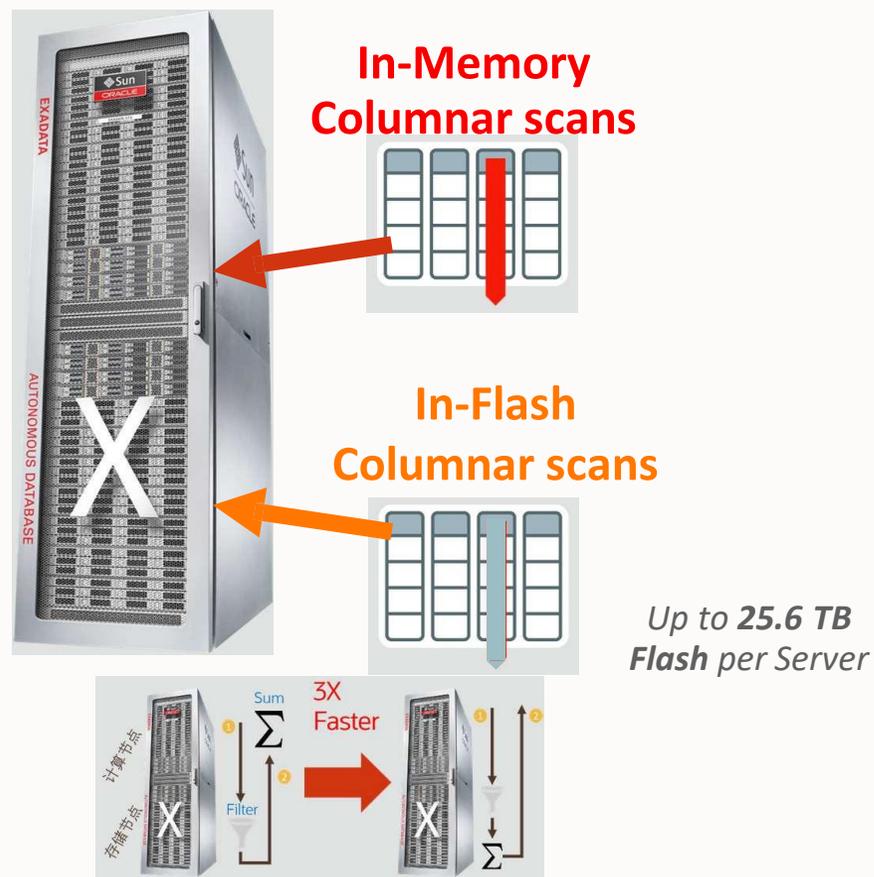


ORACLE

Exadata 是Database In-Memory最佳平台

# Oracle数据库In-Memory的最佳运行平台是Exadata

- Exadata也会在计算节点的内存缓存里使用IM列数据格式
  - 计算节点CPU访问本地内存，比访问存储节点快一个数量级
  - Exadata 高存储带宽，更快速填充内存中的IM列
  - RDS和Exafusion等专用协议极快互连，有利于IM横向扩展
- 还把IM列格式使用扩展到分布式**存储节点大容量闪存缓存中**
  - 启用IM后，HCC、OLTP压缩表、未压缩表都将可使用IM数据格式自动缓存在闪存缓存flash cache中
  - 存储节点内存中也会自动采用IM进行性能增强，IM的SIMD 向量处理等优化除在计算节点外，在各存储节点也会进行
- In-Memory 在Exadata中还提供**一系列独有功能特性**
  - 跨计算节点的IM内存镜像容错
  - ADG主备库支持IM缓存不同的数据，加速不同的业务
  - 自动化的IM缓存管理，包括内存/持久化内存/闪存/磁盘



ORACLE

# Database In-Memory如何实施

# Database In-Memory – 实施简单

第1步

配置内存容量：

`inmemory_size = XXXX GB`

第2步

设置内存对象：

`alter table |partition ...inmemory;`

第3步

运行您的应用



第4步

删除不必要的分析型索引，加速OLTP应用

# 发布: 为对象启用In-Memory

```
ALTER TABLE sales INMEMORY;
```

```
ALTER TABLE sales NO INMEMORY;
```

```
CREATE TABLE customers .....  
PARTITION BY LIST  
  (PARTITION p1 ..... INMEMORY,  
  (PARTITION p2 ..... NO INMEMORY);
```

- 新 **INMEMORY** 属性
- 可选的segment类型
  - 表
  - 分区
  - 子分区
  - 物化视图
- 以下类型不支持
  - IOTs
  - Hash clusters

纯OLTP特性

# 发布: 优先级

- 对象发布顺序由 PRIORITY 子句控制:
  - ALTER TABLE sales **INMEMORY PRIORITY HIGH;**
- 级别:
  - CRITICAL > HIGH > MEDIUM > LOW
  - 控制发布的顺序而非速度
- 默认 PRIORITY 为 NONE
  - 在第一次访问时发布

- 发布由新的一组后台进程完成
  - ora\_w001\_orcl
- 进程数量由参数控制
  - INMEMORY\_MAX\_POPULATE\_SERVERS

```
oracle@srv80101:~/In_Memory_Beta/lesson4
top - 15:32:09 up 7 days, 23:45, 7 users, load average: 14.72, 4.14, 1.55
Tasks: 622 total, 36 running, 586 sleeping, 0 stopped, 0 zombie
Cpu(s): 96.7%us, 1.9%sy, 0.0%ni, 0.0%id, 1.2%wa, 0.0%hi, 0.1%si, 0.0%st
Mem: 148834648k total, 146686500k used, 2148148k free, 187748k buffers
Swap: 2096440k total, 92k used, 2096348k free, 131648316k cached

  PID USER      PR  NI  VIRT  RES  SHR  S  %CPU  %MEM    TIME+  COMMAND
 24673 oracle    20   0 120g  1.7g  1.6g  R  79.0   1.2   6:13.27 ora_w014_orcl
 24569 oracle    20   0 120g  2.5g  2.4g  R  76.7   1.7   9:15.98 ora_w003_orcl
 24663 oracle    20   0 120g  1.7g  1.7g  R  74.4   1.2   6:52.98 ora_w00z_orcl
 24627 oracle    20   0 120g  2.0g  1.9g  R  73.1   1.4   7:57.44 ora_w00o_orcl
 24625 oracle    20   0 120g  2.2g  2.1g  R  72.4   1.5   8:42.75 ora_w00n_orcl
 24667 oracle    20   0 120g  2.0g  1.9g  R  72.1   1.4   7:51.26 ora_w011_orcl
 24571 oracle    20   0 120g  2.5g  2.3g  R  71.8   1.8   9:52.78 ora_w004_orcl
 24657 oracle    20   0 120g  1.8g  1.7g  R  71.1   1.3   6:41.06 ora_w00w_orcl
 24669 oracle    20   0 120g  2.2g  2.1g  R  70.8   1.6   8:56.33 ora_w012_orcl
 24683 oracle    20   0 120g  1.7g  1.7g  R  70.5   1.2   6:46.79 ora_w018_orcl
 24621 oracle    20   0 120g  2.0g  1.9g  R  70.1   1.4   8:12.00 ora_w001_orcl
 24687 oracle    20   0 120g  1.9g  1.8g  R  70.1   1.4   7:58.64 ora_w019_orcl
 24611 oracle    20   0 120g  2.3g  2.0g  R  69.8   1.6   8:13.25 ora_w00g_orcl
 24619 oracle    20   0 120g  1.9g  1.8g  R  68.5   1.3   6:56.29 ora_w00k_orcl
 24671 oracle    20   0 120g  1.9g  1.8g  R  68.2   1.3   7:13.82 ora_w013_orcl
 24675 oracle    20   0 120g  1.7g  1.6g  R  67.5   1.2   6:42.18 ora_w015_orcl
 24659 oracle    20   0 120g  1.9g  1.8g  R  67.2   1.3   6:53.53 ora_w00x_orcl
 24631 oracle    20   0 120g  2.3g  2.3g  R  66.9   1.6   9:28.48 ora_w00p_orcl
 24654 oracle    20   0 120g  1.8g  1.7g  R  66.9   1.3   6:57.79 ora_w00v_orcl
```



# 监控: In-Memory Column Store 的发布

## V\$IM\_SEGMENTS

- 显示哪些对象已经发布到内存中
- 显示内存中每一个segment的大小
- 显示还剩余多少未发布

```
SQL> select segment_name, populate_status, inmemory_priority, inmemory_size, bytes_not_populated from v$im_segments;
```

SEGMENT_NAME	POPULATE_STATUS	INMEM_PRIORITY	INME_SIZE	BYTES_NOT_POPULATED
ACCOUNTS	STARTED	HIGH	196606	2434886912
SALES	COMPLETED	CRITICAL	135790592	0



# 查看内存区域使用情况



V\$INMEMORY\_AREA:  
Current size of pools in  
the In-Memory area

```
SQL> SELECT * FROM v$inmemory_area;
```

POOL	ALLOC_BYTES	USED_BYTES	POPULATE_STATUS
1MB POOL	5,179,965,440	3,241,148,416	DONE
64KB POOL	570,425,344	9,568,256	DONE

V\$IM\_SEGMENTS:  
List of segments  
currently populated in  
the In-Memory column  
store

```
SQL> SELECT owner, segment_name, populate_status,  
inmemory_size, bytes_not_populated  
FROM v$im_segments;
```

OWNER	NAME	STATUS	In-Memory Size	Bytes Not Populated
SSB	LINEORDER	COMPLETED	3,206,086,656	0
SSB	DATE_DIM	COMPLETED	1,179,648	0
SSB	SUPPLIER	COMPLETED	2,228,224	0
SSB	PART	COMPLETED	18,022,400	0
SSB	CUSTOMER	COMPLETED	23,199,744	0



# 关键的一些视图

确认所有数据已完全发布!

- v\$inmemory\_area – 查看是不是有充足的内存?
- v\$im\_segments – 查看是不是所有的对象都已经完全发布?
- v\$im\_header – 查看按对象情况下的所有行是不是已经发布完?
- v\$im\_smu\_head – 变更影响了多少日志空间?
  
- v\$imeu\_header – IMEs已经发布完成?
- user\_joininggroups – 是否已经创建并填充了Join组?



# Database In-Memory与Optimizer

- 是否使用In-Memory Column Store由Optimizer决定
- 决定基于传统的表统计信息和新的In-Memory统计信息
  - 表的哪些列已发布
  - 表的数据发布了多少
  - 通过Where条件可以过滤掉多少数据

```
SELECT /* POST_IMPOP */ c.cust_id,  
c.cust_last_name, c.cust_first_name,  
SUM(s.amount_sold) FROM sh.sales s,  
sh.customers c WHERE s.cust_id = c.cust_id  
AND c.cust_last_name LIKE 'Aa%' GROUP BY  
c.cust_id, c.cust_last_name, c.cust_first_name  
ORDER BY c.cust_id;
```

Id	Operation	Name	Rows	Bytes	Cost (%CPU)	Time	Pstart	Pstop
0	SELECT STATEMENT				73 (100)			
1	SORT GROUP BY		3280	115K	73 (21)	00:00:01		
* 2	HASH JOIN		22267	782K	71 (19)	00:00:01		
3	JOIN FILTER CREATE	:BF0000	171	4446	37 (9)	00:00:01		
* 4	TABLE ACCESS INMEMORY FULL	CUSTOMERS	171	4446	37 (9)	00:00:01		
5	JOIN FILTER USE	:BF0000	918K	8973K	31 (23)	00:00:01		
6	PARTITION RANGE ALL		918K	8973K	31 (23)	00:00:01	1	28
* 7	TABLE ACCESS INMEMORY FULL	SALES	918K	8973K	31 (23)	00:00:01	1	28



# Oracle In-Memory Advisor

Object Type	Object	Estimated In-Memory Size	Analytics Processing Seconds	Estimated Reduced Analytics Processing Seconds	Estimated Analytics Processing Performance Improvement Factor	Benefit / Cost Ratio (Improvement Factor / In-Memory Size)
Table	SOE.LOGON	451.76MB	2114	1,887	9.3X	20.586
Table	SOE.CARD_DETAILS	607.32MB	8346	7,248	7.6X	12.514
Table	SOE.ADDRESSES	1.09GB	5237	4,621	8.5X	7.798
Partition	SOE.PRODUCT MOCKUP.Y2014Q1	812.6MB	2003	1,489	3.9X	4.799
Table	SOE.CUSTOMERS	1.10GB	108	95	8.2X	7.455
Table	SOE.ORDER_ITEMS	2.19GB	7128	6,393	9.7X	4.429
Table	SOE.ORDERS	1.34GB	3512	2,917	5.9X	4.403
Table	SOE.PRODUCT_INFORMATION	1.78MB	2873	2,205	4.3X	2.416
Partition	SOE.PRODUCT MOCKUP.Y2013Q4	1.62GB	97	1,489	3.7X	2.284
Partition	SOE.PRODUCT MOCKUP.Y2014Q2	3.37GB	642	493	4.3X	1.276

- In-Memory Advisor – 可在 OTN 上免费下载 11.2.0.3+ 数据库
- In-Memory Advisor – 参考OTN 或 MOS Note 1965343.1
- 通过 AWR 和 ASH 存储库分析现有的数据库工作负载，生成建议报告
- 提供发布到IM 列存储中受益最大的对象列表



**Note:** Database Diagnostics and Tuning Pack licenses required

# Database In-Memory指南和工具

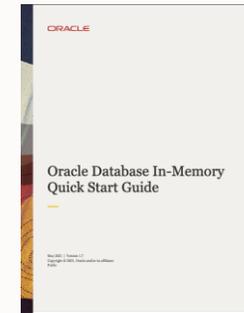
## 行动起来…

- 验证用例是否符合 Database In-Memory 的优势
- 使用 In-Memory Advisor 验证收益
- 不要“浅尝辄止”，通过使用指南获得更多信息
- 使用最新的 Oracle 数据库版本和最新的 RU，从最新的创新和修复中获益

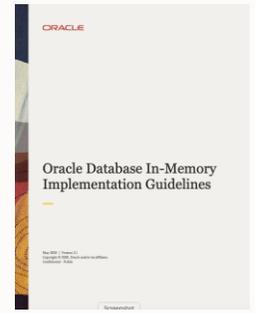
## 这些指南可以帮助我们尽快入门…



快速入门(一页)  
([see here](#))



快速入门指南  
([see here](#))



实施指南  
([see here](#))



# Database In-Memory学习资源



## Database In-Memory Blog

<https://blogs.oracle.com/in-memory>

<https://blogs.oracle.com/in-memory/dbim-resources>

## Database In-Memory Hands-on-Lab

<https://apexapps.oracle.com/pls/apex/dbpm/r/livelabs>

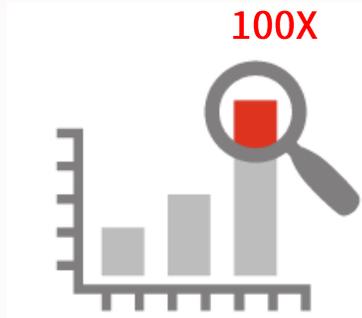
## Database In-Memory Documentation

<https://docs.oracle.com/en/database/oracle/oracle-database/21/inmem>

# Oracle Database In-Memory

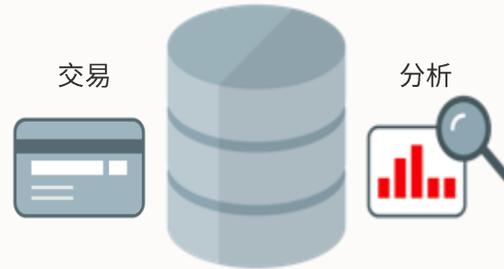


## 实时分析



赋能实时业务决策

## 加速混合负载



在运营系统上运行  
分析

## 没有风险



久经考验的横向扩展  
可用性和安全

## 实施简单



应用无需改动  
内存没有限制

ORACLE

# 客户案例分享

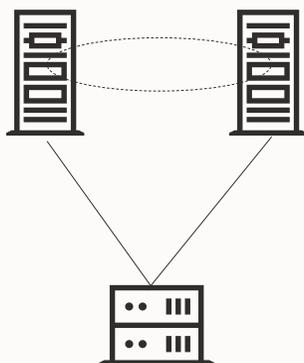
---

# 项目 S 背景



SAP ERP 上线

2010



2 SUN T5-4 + SAN 存储

2015

系统忙

- 每日数百人次登录

数据多

- 2.3TB 数据, 日增数十GB

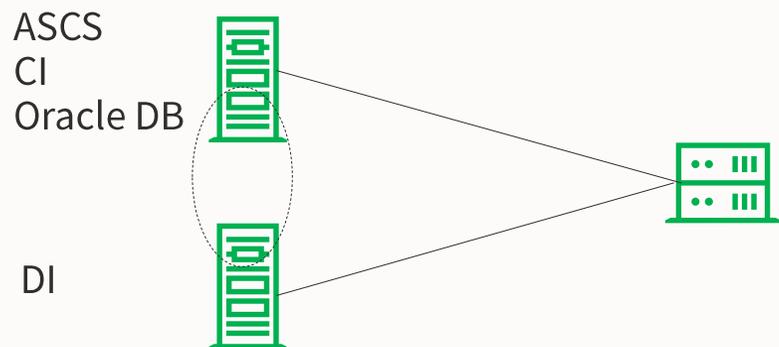
程序慢

- 部分常用代码慢, 超过1小时以上30个, 超过5小时10个
- 月结久, 集中出报表, 压力大, 历时数天
- 存储性能差, 单次 I/O > 5ms

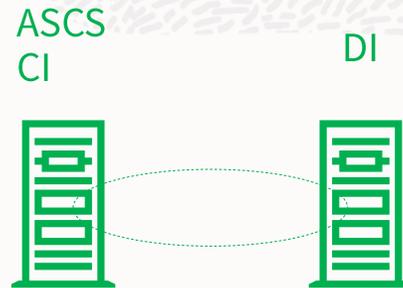
~60 家分公司和套账

2020

# 项目 S 解决方案



	现状	问题
硬件平台	Sun T5-4	使用5年, 设备老化
架构	主/备	资源浪费, 不能横向扩展
存储	SAS 阵列	单次 I/O > 5ms
数据库版本	11.2.0.4	不支持 DBIM
操作系统	Solaris	停止研发



Exadata



升级到 19c



启用 Database In-Memory



持续优化



# 总体性能提升

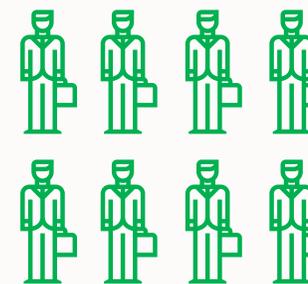
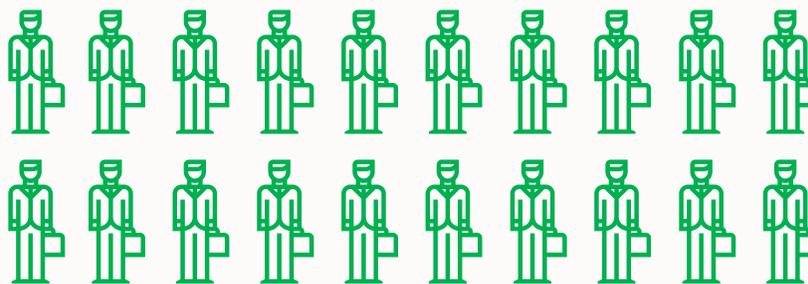
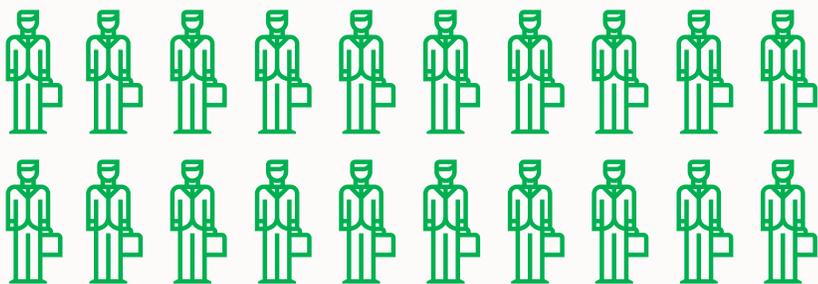
共1562个程序，实际仅计算前台程序收益1252个

7月 (升级前)		9月 (升级后)		节省时间(小时/月)
执行总次数	执行总时间	执行总次数	执行总时间	
66,948,677	14,741	77,991,953	6,778	7,963

9月的执行总次数比升级前的7月多 **16%**

但执行总时间下降 **54%**

• 每年约节省人员: **48**



# Stef

- STEF是欧洲冷链物流领导者 STEF is the European leader for temperature sensitive logistics
- 商业智能信息系统是STEF日常活动检测的关键 Business Intelligence Information System is key in STEF day-to-day activities monitoring
- STEF管理从生产到分校的完整供应链 It manages the complete supply chain from Production to Distribution
- STEF拥有超过15000家大客户，业务遍及西欧的7个主要国家 Stef has 15000 large



## 商业挑战：

- STEF的商业经理需要更多更快的决策支持信息
- 在使用DBIM之前，业务经理无法访问所有的数据用来做出决策
- 在访问数据管理业务的同时，数据也经常通过新事务进行更新
- 每天大约有1亿条新数据插入，以及1000万条的更改和删除

## 使用DBIM后：

- 最主要的查询语句性能提升10-100倍
- 使用DBIM强大的报告和分析功能，从运输订单到计费，为终端客户提供更高效实时的服务
- 轻松满足高要求的分析要求

# 汉莎航空Lufthansa

汉莎航空是德国最大的航空公司，也是欧洲载客量最大的航空公司之一。



## 商业挑战：

- 在全球60多个机场检测维护超过2000架飞机
- 每6-10年进行一次D-Check，飞机几乎完全直接拆卸再重装
- 每一个零件都经过检查测试是否需要维修更换，从咖啡机到发动机到起落架
- 维修在自己的车间或由分包商完成
- 24/7/365全年无休运营

## 使用DBIM后：

- 对重要查询语句高达100x倍的性能提升
- 节约90%的IO
- 节约磁盘空间 (indexes dropped)

免费咨询热线：400-699-8888



# Database In-Memory 助力实时企业

## Mankind Pharma



- 分析报表 **11x** 提速
- 删除分析型索引提速OLTP
- 数据库大小 **减少 90%**

## BOSCH – SAPCRM



- **删除** 所有定制的索引
- 分析查询 **2-20X** 提速, DML **2-3X** 提速
- 无需改变应用

## LION – SAPERP



- 分析查询 **4X** 提速
- 交易 **2X** 提速
- 分析查询现在可以基于1亿销售点交易数据运行

## Lufthansa



- 分析查询 **100x** 提速
- 提升了数据摄取的速度
- 减少了数据库大小



# 甲骨文内存数据库DBIM = 简单易用 + 高性能

传统模式

更高的成本和  
复杂性

物理视图

分析索引

管理风险

DBA的管理代价

管理多个存储及数  
据备份

管理不同的计算

数据查询前需要进  
行ETL处理

现代模式  
(简易高效)

简单而统一的  
数据存储, 提  
供高达100x的  
性能提升

Database  
In-Memory

您的选择?





# 基于 Oracle 数据库 免费企业数据健康检查

- 及时了解数据库健康状况，发现并解决潜在问题
- 维护数据库系统良好状态，保护数据资产的安全
- 提升数据库性能、稳定性和安全性，降低业务风险

免费咨询热线：

**400-699-8888**

\* 活动最终解释权归甲骨文公司所有

ORACLE 甲骨文

# Active Data Guard 容灾与保护 - Part2

## 实战演练工作坊系列(九)



**梅玲**

- 资深解决方案工程师
- 10年+ Oracle数据库架构经验
- 曾服务金融，电信多个行业项目实施

### 内容简介

通过工作坊动手实践Demo深入体验Oracle Active Data Guard技术的相关功能和新特性，并结合实验进一步了解特性的适用场景。

在Part 2部分我们会介绍ADG备库DML重定向，Far Sync实例的配置，以及Switchover 和 Failover

**实验包括：**

- ADG备库DML重定向
- Far Sync实例的配置
- 计划内主备切换switchover
- 计划外容灾切换failover



直播时间：5月5日 11:00 - 12:00  
 扫描二维码进入直播  
 Zoom ID 976 6962 5763  
 密码：98039717



微信扫一扫预约



数据库和云讲座群

20-20



甲骨文云技术公众号



技术专家1v1深入交流



ORACLE