昆仑分布式数据库 &OracleRAC&Sharding技术分析

泽拓科技 Zettadb.com

泽拓科技KunlunDB

纲要

- Oracle 集群技术分析
- Oracle Sharding 技术分析
- 昆仑分布式数据库

泽拓科技KunlunDB

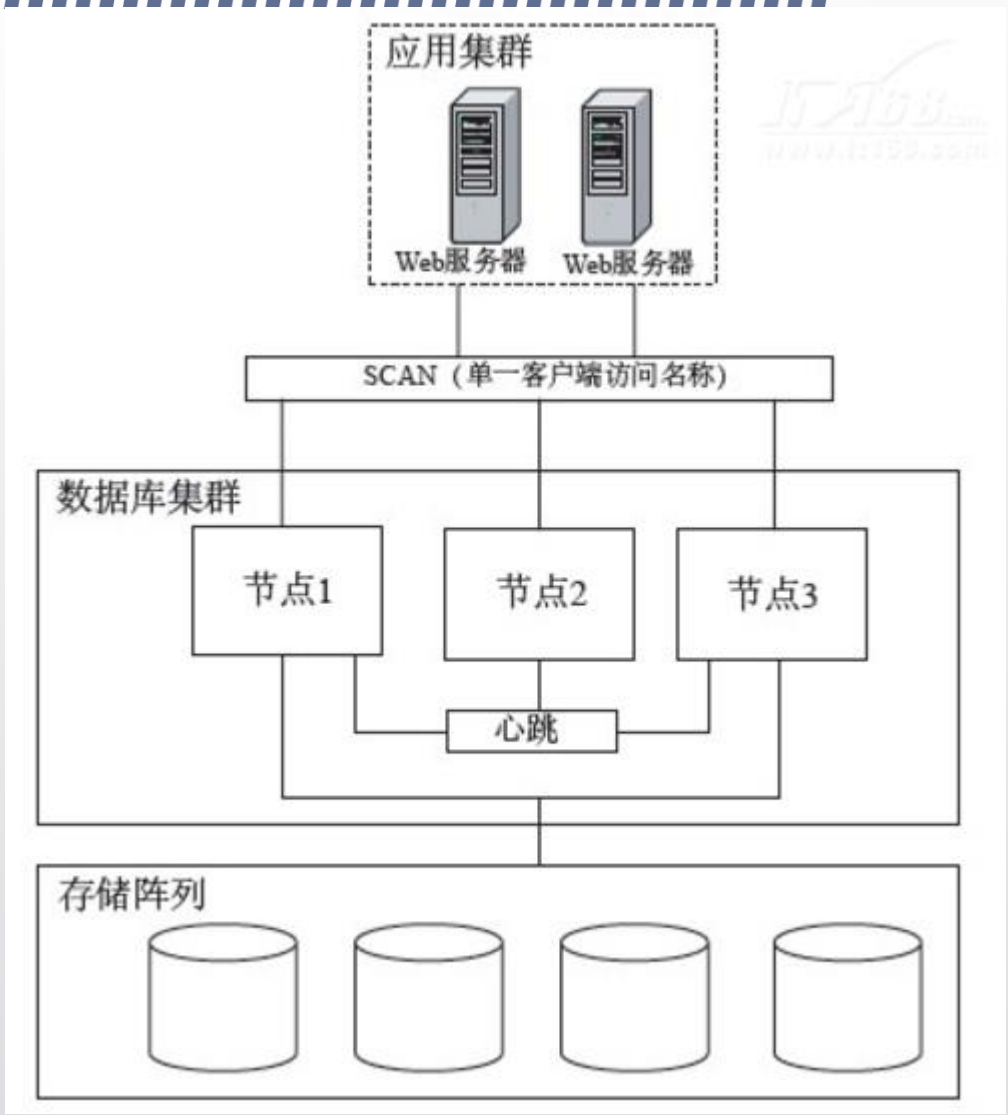
双节点Oracle RAC技术架构

RAC 的使命：

1， 提高可靠性，负载均衡，避免单节点故障

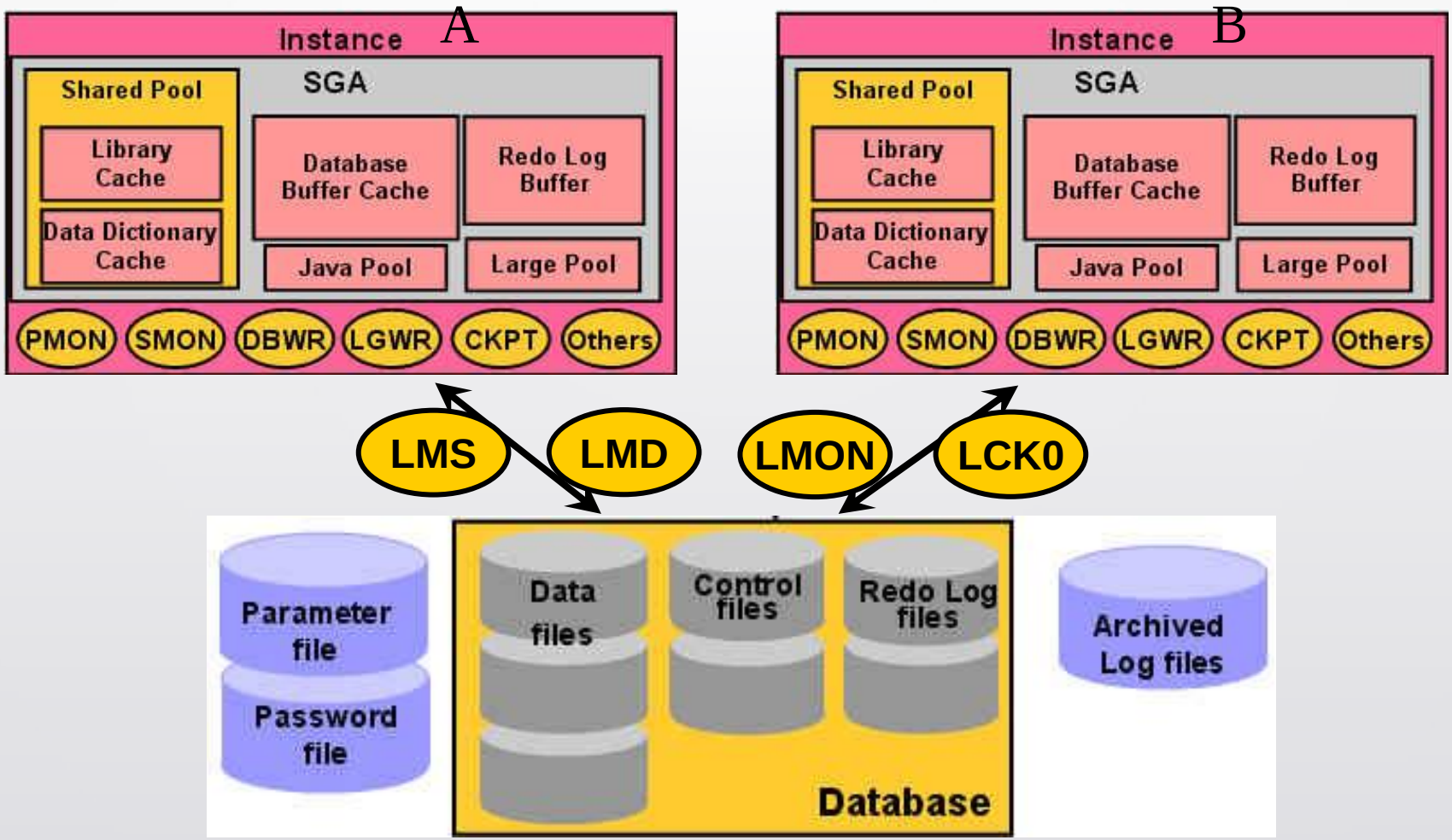
2，解决单节点处理能力瓶颈

3，小型机迁移到X86 ，降低成本



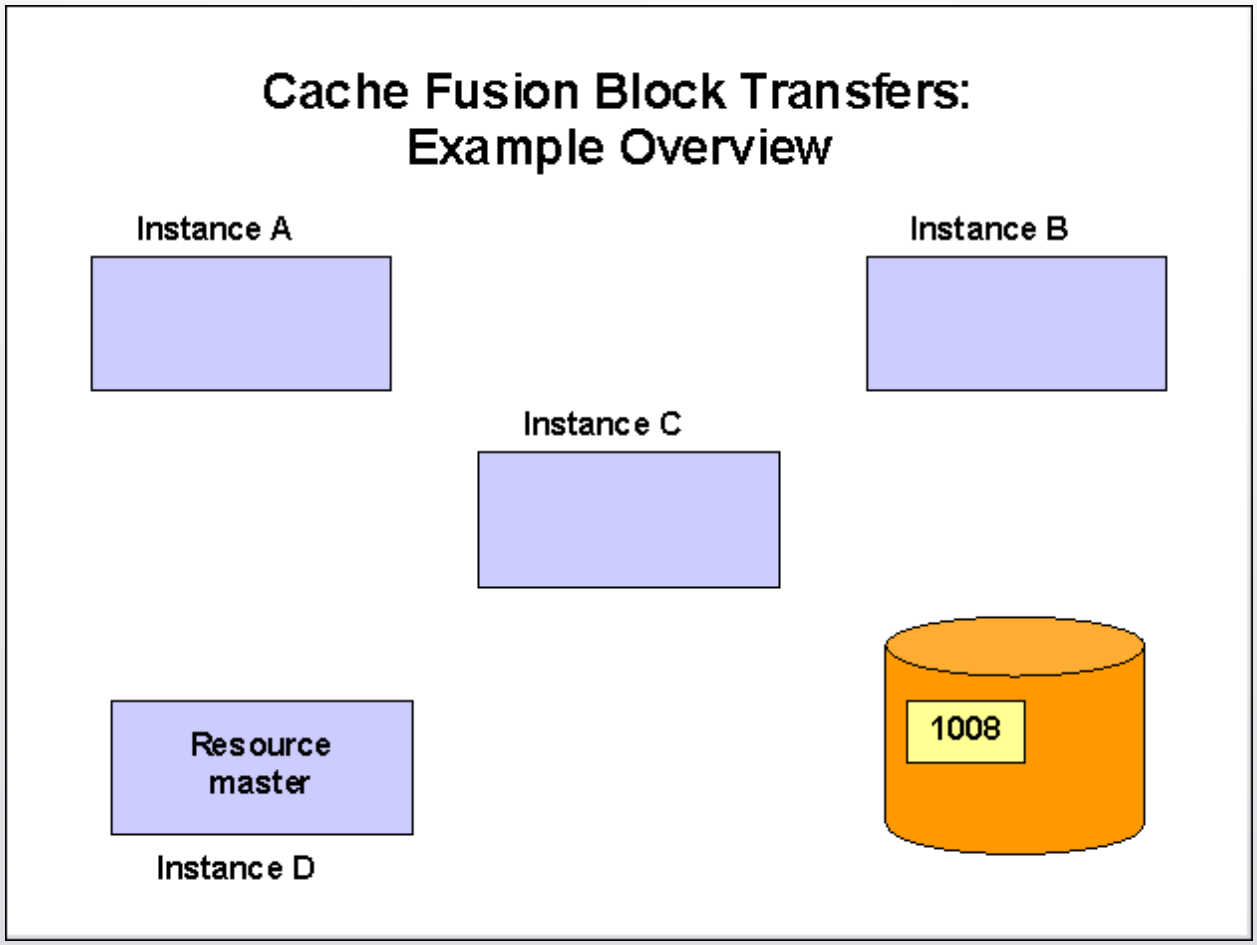
泽拓科技KunlunDB

双节点Oracle RAC技术架构



泽拓科技KunlunDB

Cache Fusion 示例



数据块属性

状态：

PI,CR,SCUR,XCUR

访问模式：

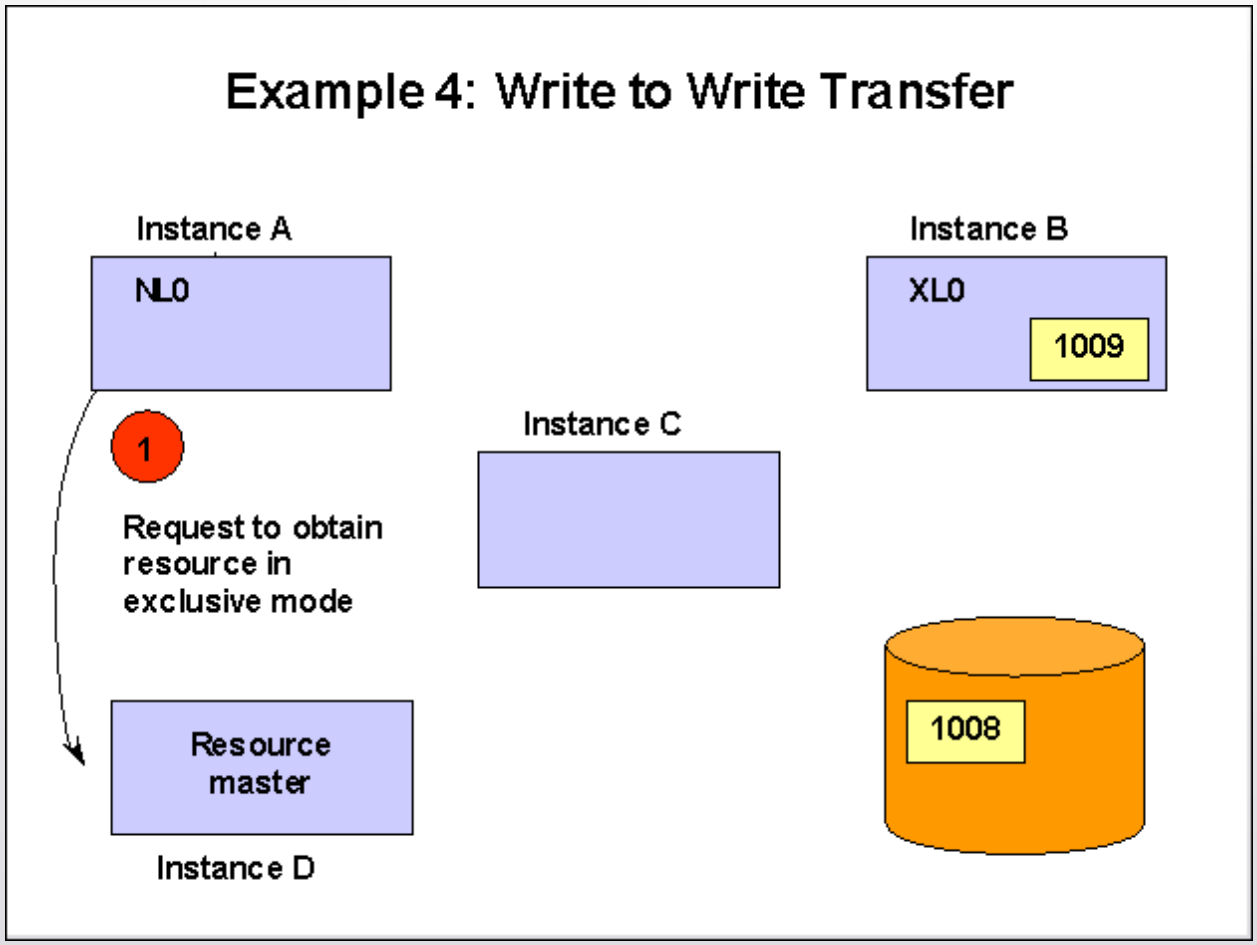
NULL,S,X

角色：

Local，Global

泽拓科技KunlunDB

Cache Fusion 示例



数据块属性

状态：

PI,CR,SCUR,XCUR

访问模式：

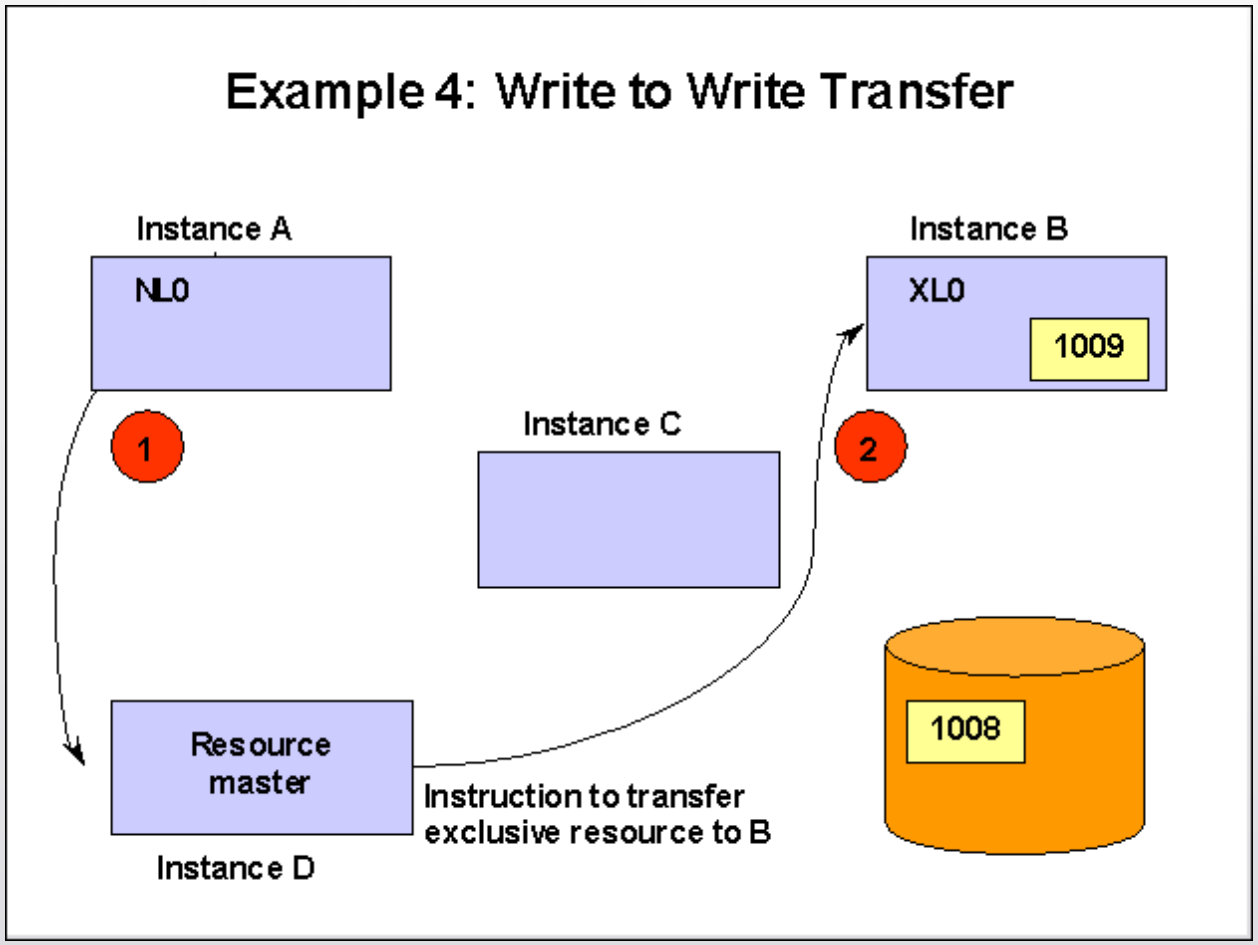
NULL,S,X

角色：

Local，Global

泽拓科技KunlunDB

Cache Fusion 示例



数据块属性

状态：

PI,CR,SCUR,XCUR

访问模式：

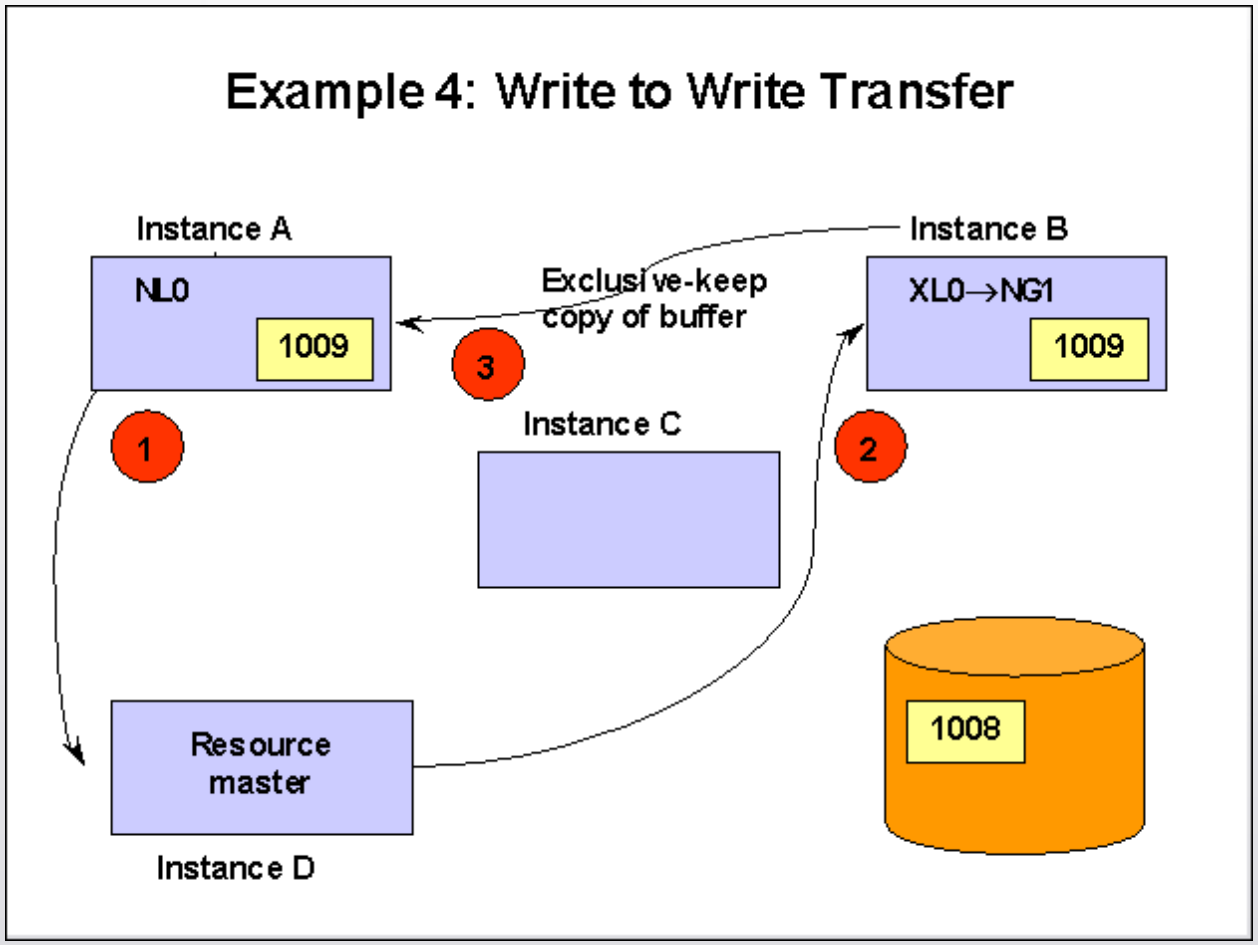
NULL,S,X

角色：

Local，Global

泽拓科技KunlunDB

Cache Fusion 示例



数据块属性

状态：

PI,CR,SCUR,XCUR

访问模式：

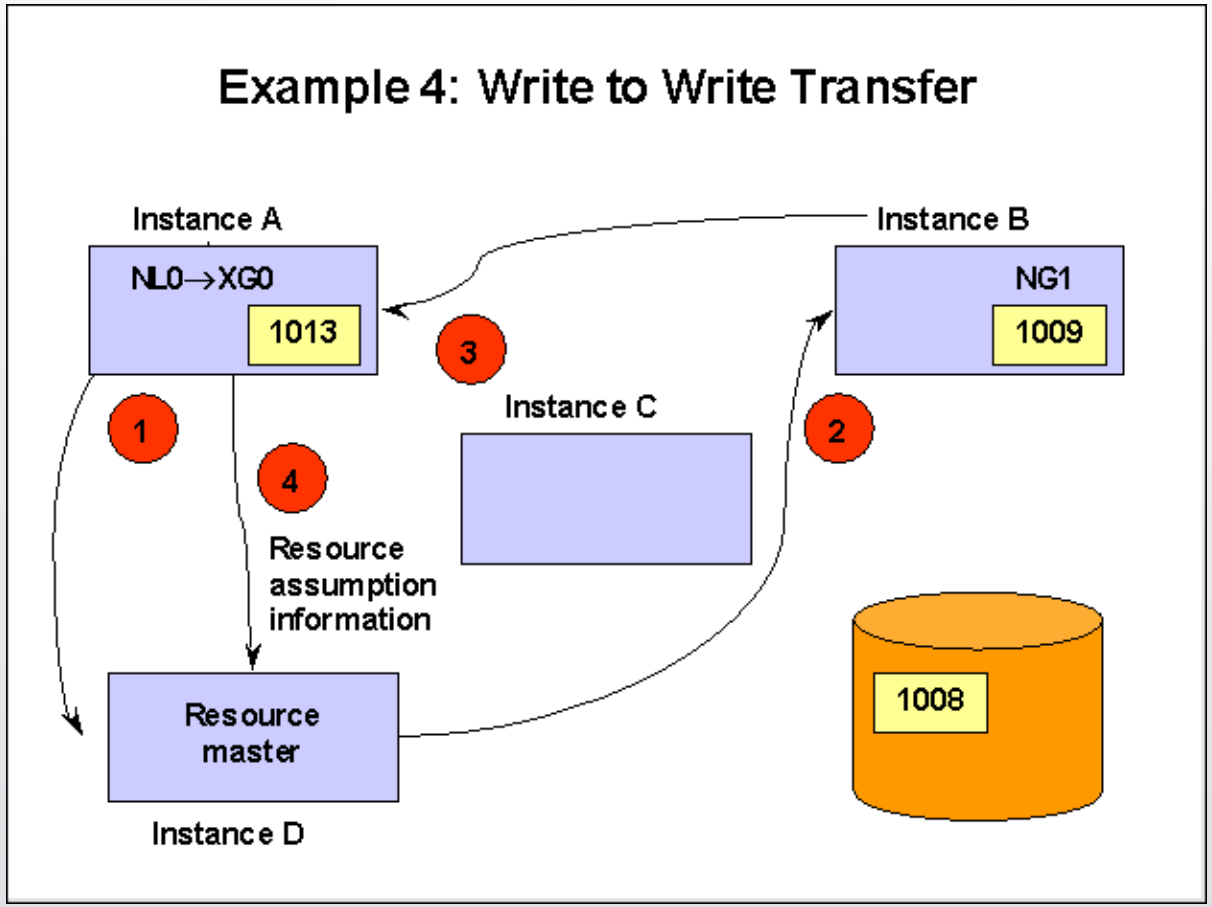
NULL,S,X

角色：

Local，Global

泽拓科技KunlunDB

集群环境下数据处理流程示例



数据块属性

状态：

PI,CR,SCUR,XCUR

访问模式：

NULL,S,X

角色：

Local，Global

泽拓科技KunlunDB

Oracle RAC 技术特性

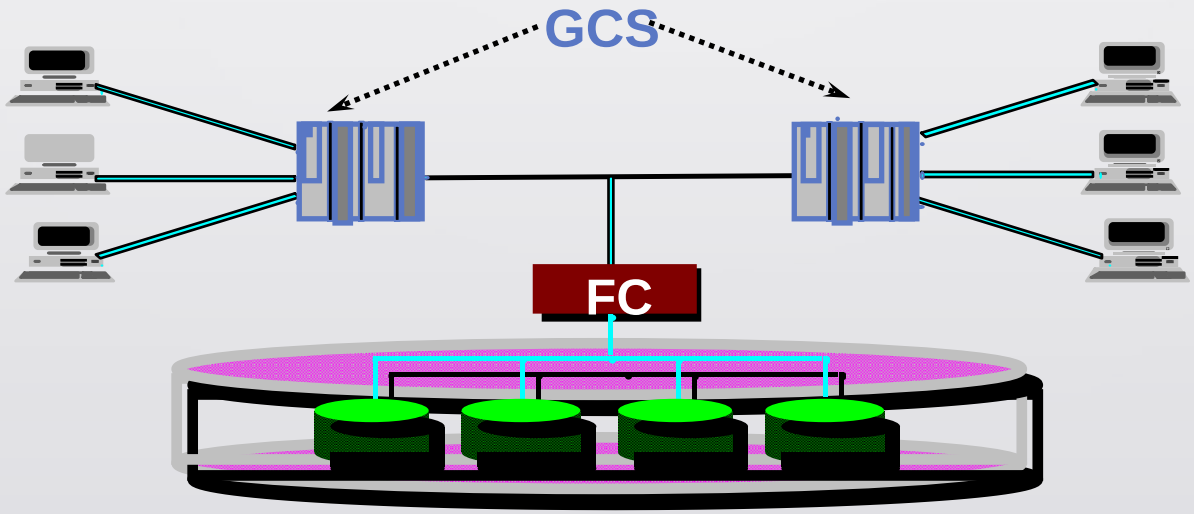
1.高性能：并行处理

2.高可用性&可靠性 ： 避免单点故障，提供负载均衡

3.高扩展性：容量动态扩展

4.成本：X86 架构

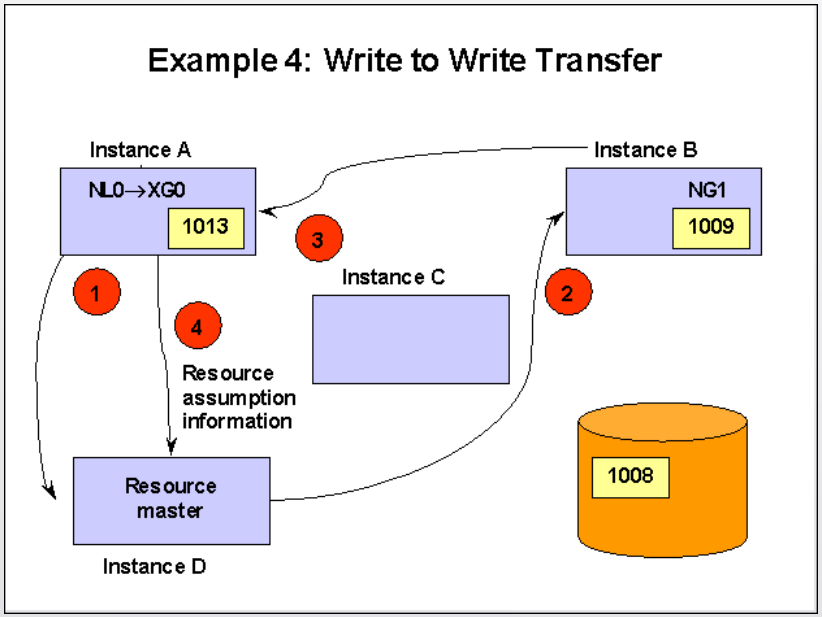
5.可管理性：统一的集群管理系统



泽拓科技KunlunDB

Oracle集群在实际应用中面对的挑战

1.性能方面：分布式一致性缓存受限于网络带宽&延迟，横向扩展过程中带来的跨节点数据融合带来性能问题，无法满足业务负载的增长需求

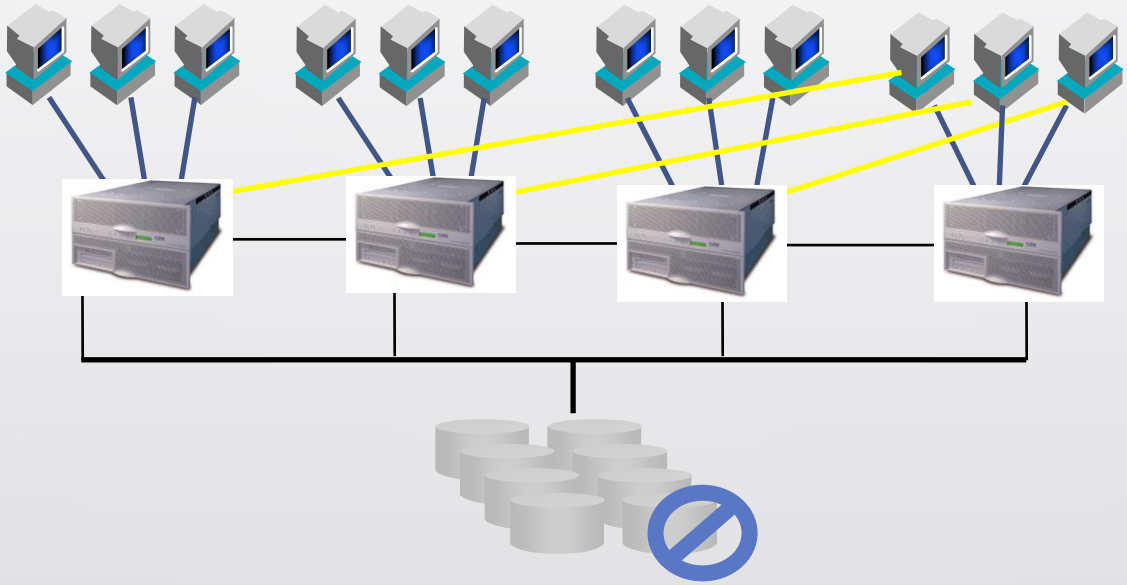


技术	访问延迟
CPU寄存器	<1ns
CPU Cache	1-50ns
本地内存	100ns
NUMA架构非本地内存	200-300ns
万兆以太网(UDP)	30-100us
千兆以太网(UDP)	100us-200us

泽拓科技KunlunDB

Oracle集群在实际应用中面对的挑战

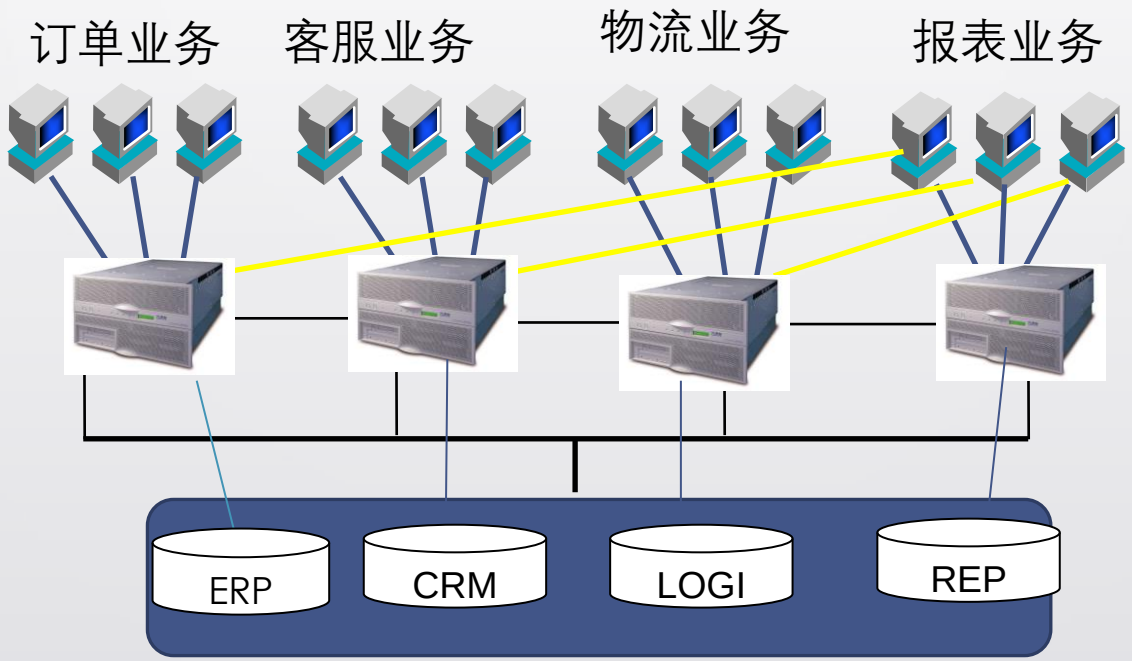
2.可用性&可靠性方面：单纯RAC 共享存储架构依然存在存储失效的单点故障，需要配合dataguard 等容灾方案提高可靠性



泽拓科技KunlunDB

Oracle集群在实际应用中面对的挑战

3.扩展性：实际应用中只有少量4个节点以上案例，高并发多节点场景也需要通过业务分区实例绑定方案避免各种问题&隐患



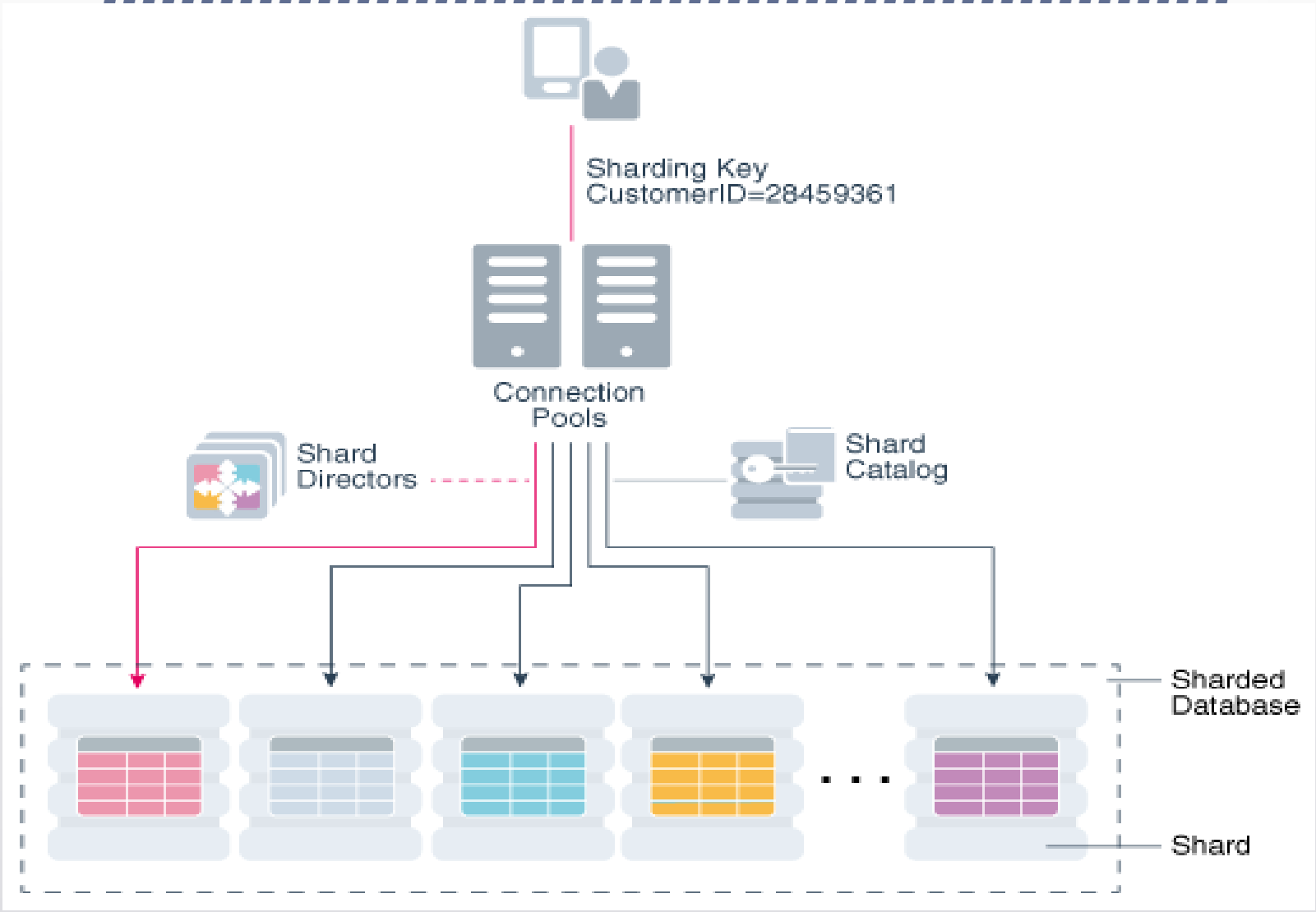
泽拓科技KunlunDB

纲要

- Oracle 集群技术分析
- Oracle Sharding 技术分析
- 昆仑分布式数据库

泽拓科技KunlunDB

Oracle Sharding 技术架构

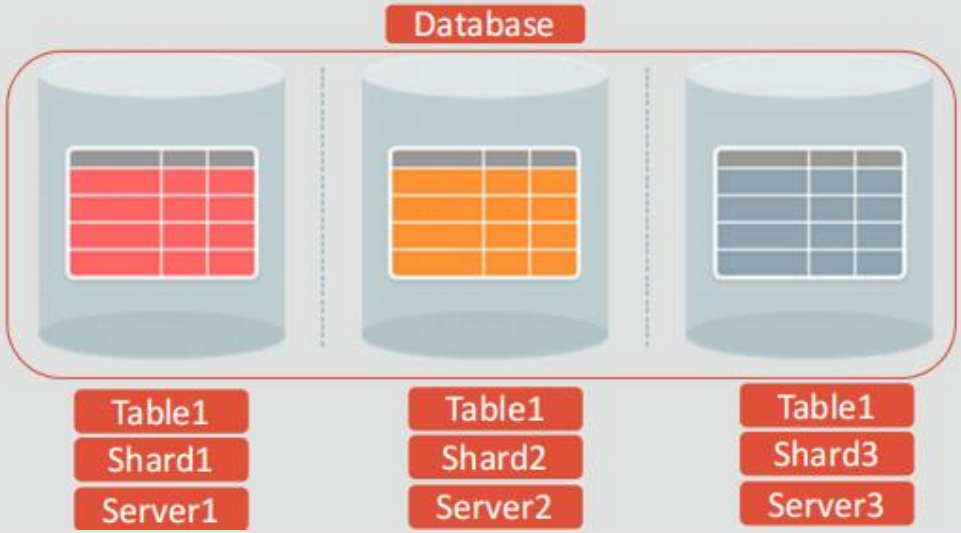


泽拓科技KunlunDB

Oracle Sharding 技术架构

将单个逻辑数据库分片到N个物理数据库中

- 跨独立数据库的数据水平分区(分片)
  每个分片保存数据的一个子集
  - 可以是单实例数据库, RAC 或者 PDB
  - 数据复制以实现高可用性
- 无共享架构(Shared-nothing):
  - 分片不共享任何硬件(CPU, 内存, 磁盘)或软件(群集软件)



泽拓科技KunlunDB

Oracle Sharding 技术架构

主要组件：

✓ 数据存放在 Sharded Database

多个shard 节点组成 一个 share-nothing 数据存储群

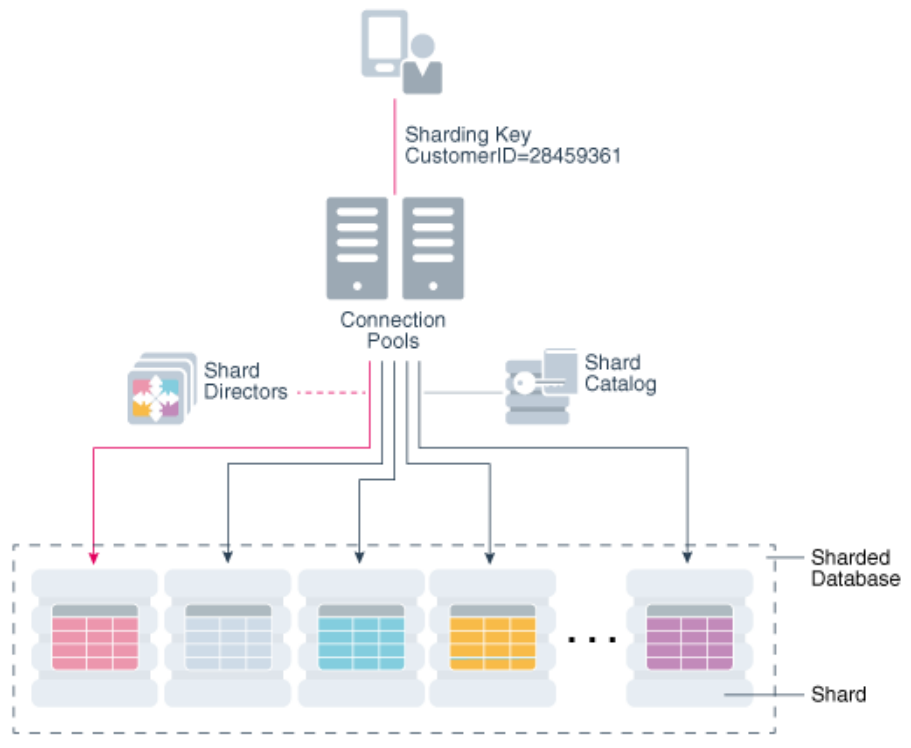
✓ Shard catalog

集中存储和管理SDB配置信息。添加/删除shard等配置变化都记录在Shard catalog。应用跨多个shard查询，由Shard catalog统一协调。Shard catalog可配置为HA高可用模式

Shard Director:

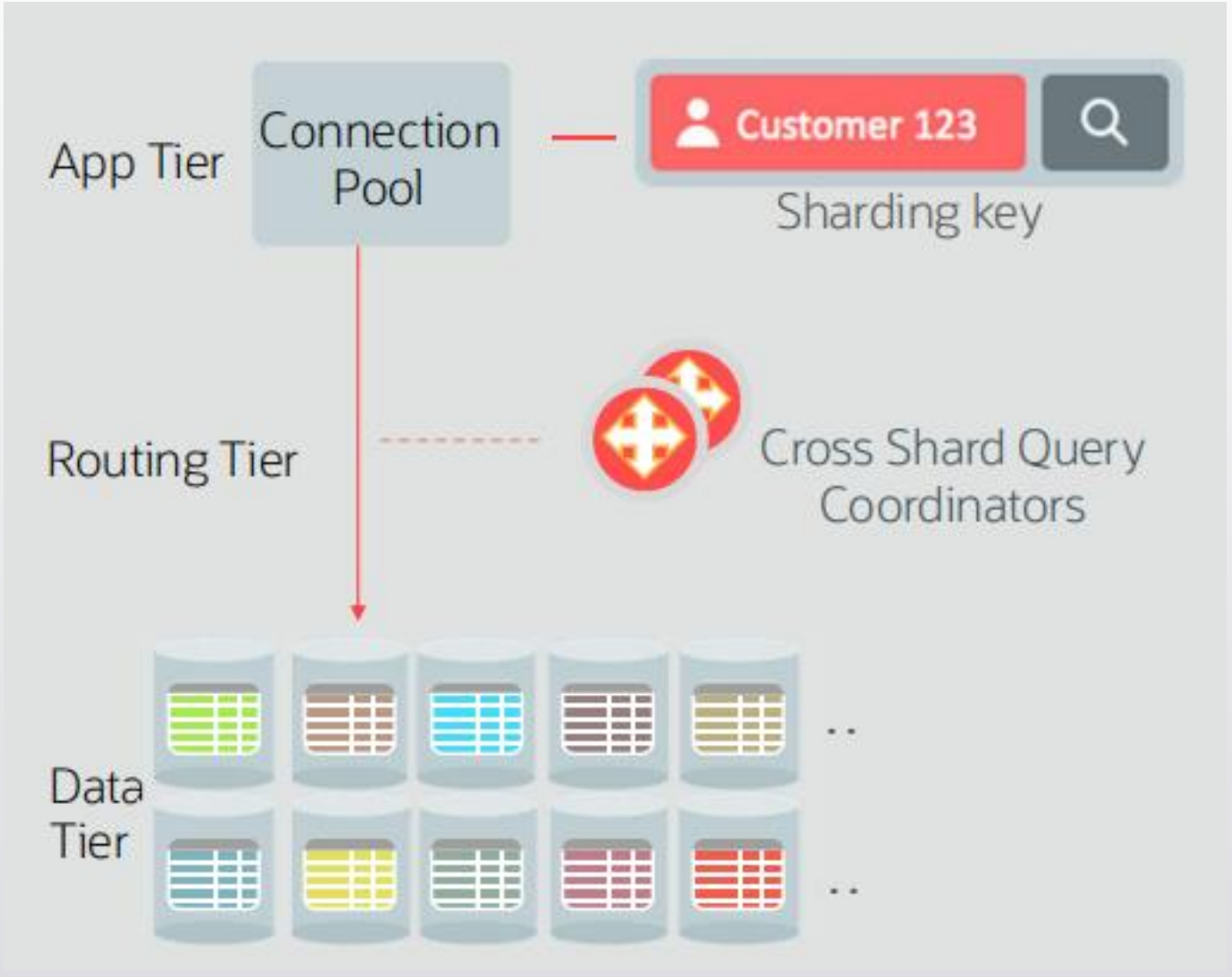
全局服务管理器，用于将连接请求直接路由到分片， 通过ONS发布运行时分片数据库拓扑图，负载平衡咨询，FAN事件。

可以部署多份提高可用性



泽拓科技KunlunDB

Oracle Sharding 技术架构



泽拓科技KunlunDB

Oracle Sharding 技术优点

线性可伸缩性

在线添加分片以增加数据库大小和吞吐量。在线拆分和重新平衡数据。

极高的可用性

无共享架构。一个分片的故障对其他分片没有影响。

地理分布

用户定义的数据放置，以提高性能，可用性，灾难恢复或满足法规要求。

泽拓科技KunlunDB

Oracle Sharding 示例

创建分片表家族

```
CREATE TABLESPACE SET  tbs1 ;

CREATE SHARDED TABLE Customers
( CustId    VARCHAR2(60) NOT NULL,
  FirstName VARCHAR2(60),
  LastName  VARCHAR2(60),
  …
  CONSTRAINT pk_customers
    PRIMARY KEY(CustId)
)
PARTITION BY CONSISTENT HASH (CustId)
PARTITIONS AUTO
TABLESPACE SET tbs1 ;
```

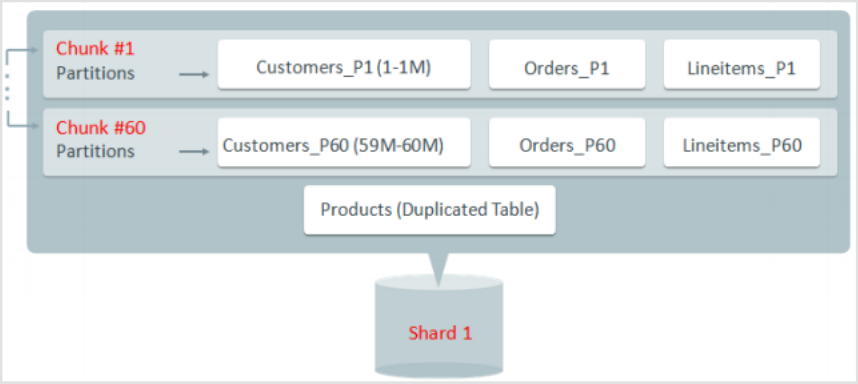
```
CREATE SHARDED TABLE Orders (
  OrderId   INTEGER,
  CustId    VARCHAR2(60),
  OrderDate TIMESTAMP,
  …
  CONSTRAINT pk_orders
    PRIMARY KEY (CustId, OrderId),
  CONSTRAINT fk_orders_parent
    FOREIGN KEY (CustId) REFERENCES
Customers(CustId)
)
PARTITION BY REFERENCE (fk_orders_parent) ;

CREATE DUPLICATED TABLE Products (
ProductId INTEGER PRIMARY KEY,
Name      VARCHAR2(128),
LastPrice NUMBER(19,4),
…
)
TABLESPACE products_tsp ;
```

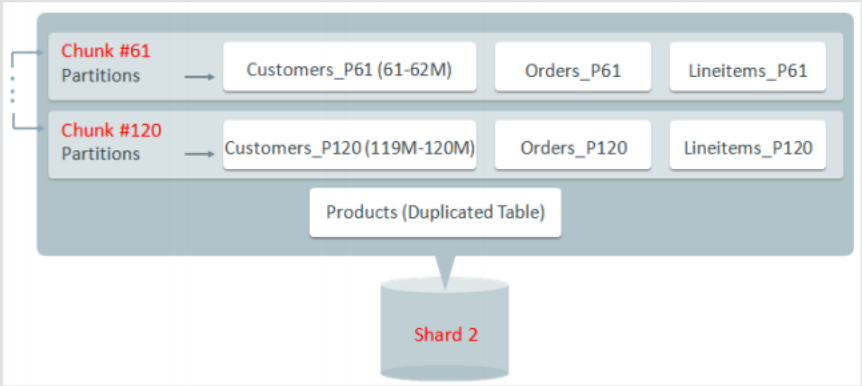

泽拓科技KunlunDB

Oracle Sharding 示例

Chunk

- 表族里所有分片表相应分区所在的表空间的集合
  - 例: Chunk#1 包含 Customers_P1, Orders_P1, LineItems_P1
- 给定分片键值(sharding key)的所有数据都位于一个 chunk
  - 无需访问多个shard数据库
- 分片重平衡的数据移动单元





泽拓科技KunlunDB

Oracle Sharding 技术特性分析

Sharding架构对应用的要求

- 仅在新建数据库时可用
- 适用于OLTP应用：
  - ✓ 具有定义良好的数据模型和数据分布策略
  - ✓ 通常访问与分片键的单个值相关联的数据
  - ✓ 使用Oracle集成连接池（UCP、OCI、ODP.NET和JDBC）连接到分片数据库

"Sharding 应该用于穷尽其他优化有段仍不能满足需求的场景."
--维基百科

35

泽拓科技KunlunDB

Oracle Sharding 技术特性分析

- ✓ 产品特性定位在处理数据规模扩容、可用性、地理分布的业务特性，在大数据处理性能上没有优化， 特别在处理跨 Shard 的数据处理的场景性能下降明显
- ✓ 只支持 sharded Table 和 duplicatied table 的创建，并不能灵活支持对微服务等应用场景

泽拓科技KunlunDB

Oracle Sharding 技术特性分析

DMLs on Sharded table

DML statement that affects only **one shard** is supported

Example:

```
update S1 set col = … where sk = <constant>;
```

Within a transaction, multiple single shard DMLs can be performed on different shards

Example:

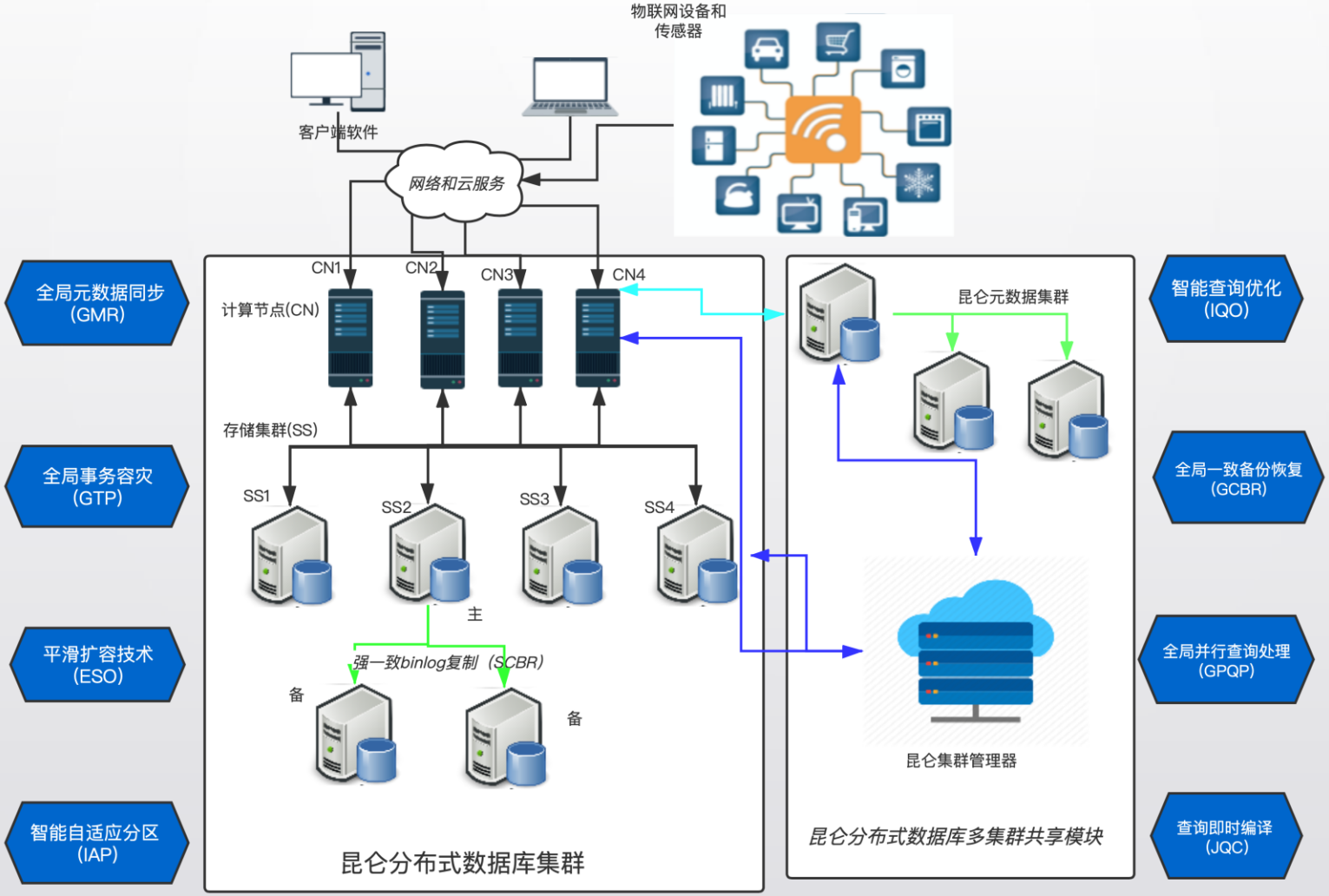
```
insert into S1 values (…);                  /* affects shard #1 */
Update S1 set col = … where sk = <constant>;  /* affects shard #2 */
Delete S1 where sk = <constant>;              /* affects shard #3 */
Commit;
```

Coordinator starts a distributed transaction and performs two phase commit

DML statement that affects more than one shard is not currently supported (12.2)

泽拓科技KunlunDB

昆仑数据库技术架构



泽拓科技KunlunDB

昆仑分布式数据库技术特点

1. 高性能：分布式查询优化，查询计算下推，节点间松偶合，扩容计算节点可以提高系统处理性能

2. 扩展性：复杂环境海量数据（无限制）的支持，按需横向扩展计算节点（CN)
3. 高可用：计算节点互为冗余备份及负载均衡，数据节点一主二备

4. 可靠性：优化的分布式事务处理机制可以保障事务原子性， 数据全局一致性

5. 配置简单， 易于维护， 技术自主，成本可控

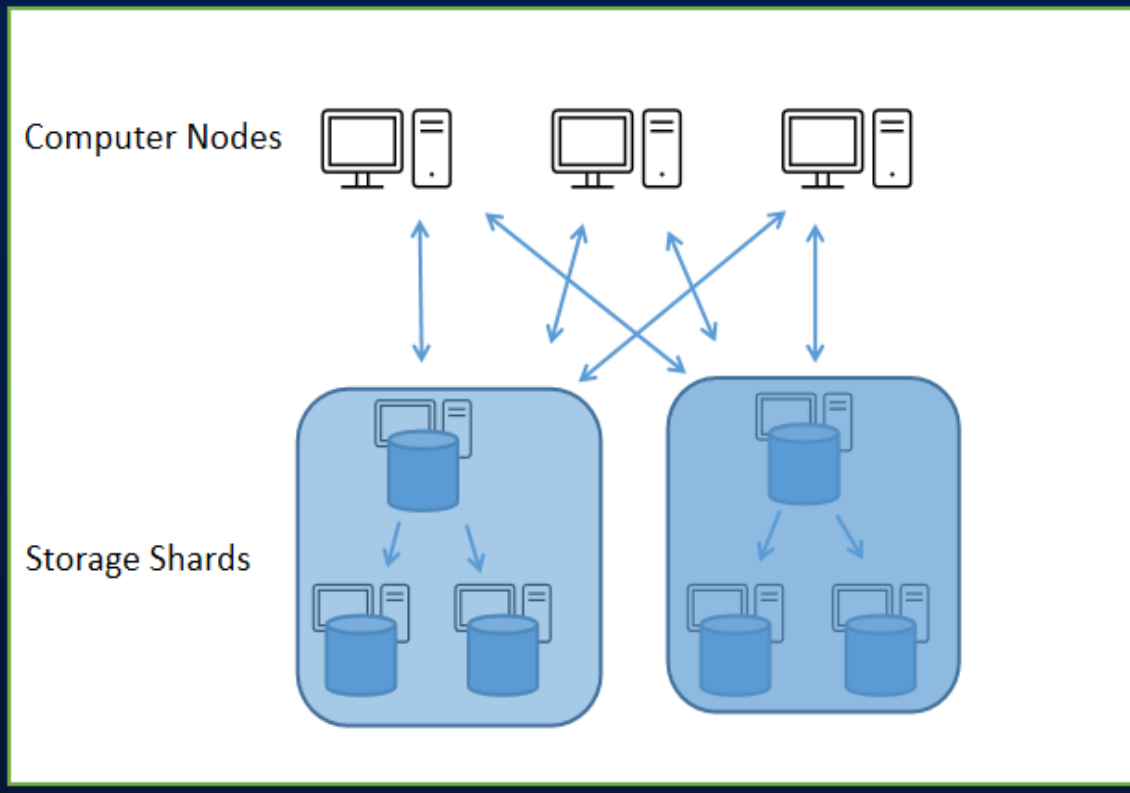
泽拓科技KunlunDB

昆仑分布式数据库分片示例

环境介绍

三个计算节点，构成三个各自独立的计算节点，用户连接任意一台计算节点执行SQL 建表语句。

六个存储节点，构成两个Data Shard, 每个Shard的副本数为2



数据表：

父表

```
CREATE TABLE measurement (
    city_id        int not null,
    logdate        date not null,
    peaktemp       int,
    unitsales      int
) PARTITION BY RANGE (logdate);
```

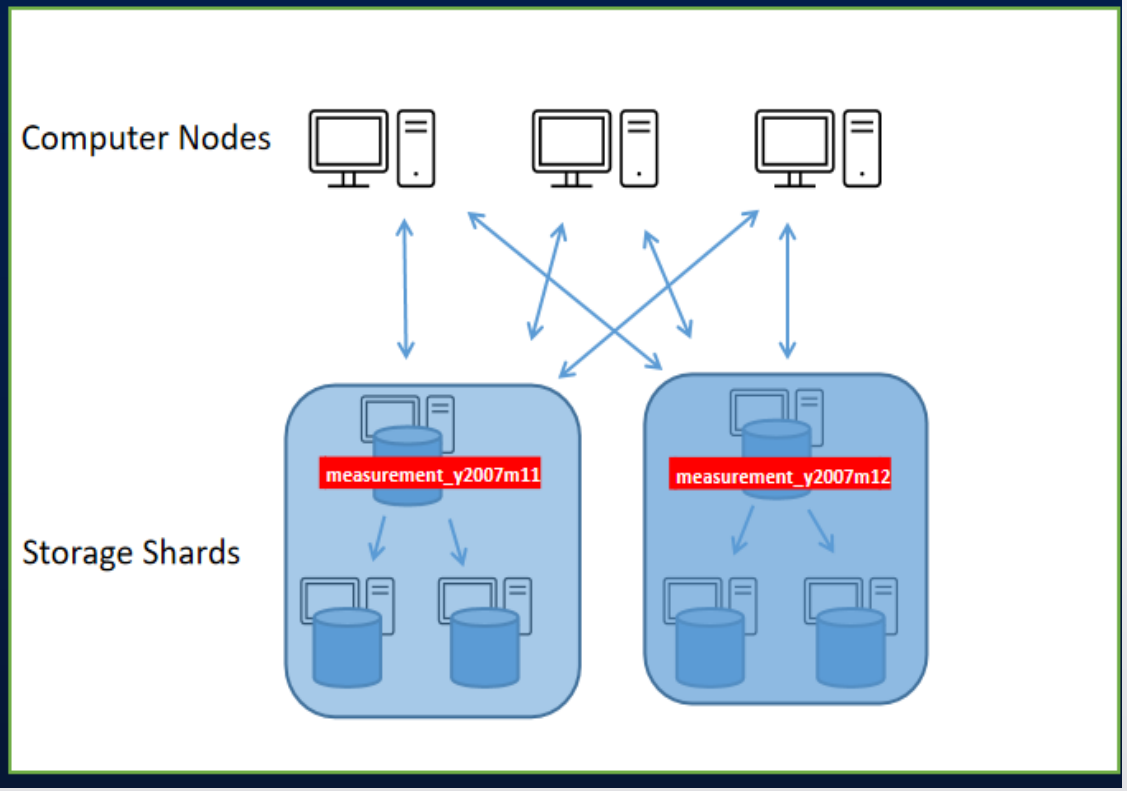
```
alter table measurement add constraint table_pkey primary key (city_id,logdate);
```


泽拓科技KunlunDB

昆仑分布式数据库分片示例

创建表分区

```
CREATE TABLE measurement_y2007m11
PARTITION OF measurement
   FOR VALUES FROM ('2007-11-01') TO ('2007-
12-01');

CREATE TABLE measurement_y2007m12
PARTITION OF measurement
   FOR VALUES FROM ('2007-12-01') TO ('2008-
01-01');
```



泽拓科技KunlunDB

昆仑分布式数据库分片示例

每个分片的数据节点只可以查看到部分区表的信息

```
instance_name 08:44:40>select @@port;
+--------+
| @@port |
+--------+
|   6004 |
+--------+
1 row in set (0.00 sec)
```

```
instance_name 08:45:20>select @@port;
+--------+
| @@port |
+--------+
|   6007 |
+--------+
1 row in set (0.00 sec)
```

```
instance_name 08:45:02>show tables;
+----------------------------+
| Tables_in_runoobdb_$$_public |
+----------------------------+
| measurement_y2007m11       |
+----------------------------+
1 row in set (0.00 sec)
```

```
instance_name 08:46:02>show tables;
+----------------------------+
| Tables_in_runoobdb_$$_public |
+----------------------------+
| measurement_y2007m12       |
+----------------------------+
1 row in set (0.00 sec)
```


泽拓科技KunlunDB

昆仑分布式数据库分片示例

从任意计算节点查询数据

```
explain select city_id from measurement;
```

```
 Append  (cost=0.00..0.01 rows=2 width=4)
   ->  RemoteScan on measurement_y2007m11
(cost=0.00..0.00 rows=1 width=4)
       Shard: 2       Remote SQL: select city_id from
runoobdb_$$_public.measurement_y2007m11
   ->  RemoteScan on measurement_y2007m12
(cost=0.00..0.00 rows=1 width=4)
       Shard: 1       Remote SQL: select city_id from
runoobdb_$$_public.measurement_y2007m12
(5 rows)
```

```
explain select city_id from measurement where
logdate='2007-12-15';
```

```
Append  (cost=0.00..0.01 rows=1 width=4)
   ->  RemoteScan on measurement_y2007m12
(cost=0.00..0.00 rows=1 width=4)
       Filter: (logdate = '2007-12-15'::date)
       Shard: 1       Remote SQL: select city_id from
runoobdb_$$_public.measurement_y2007m12  where
(logdate = '2007-12-15')
(4 rows)
```


泽拓科技KunlunDB

泽拓科技KunlunDB