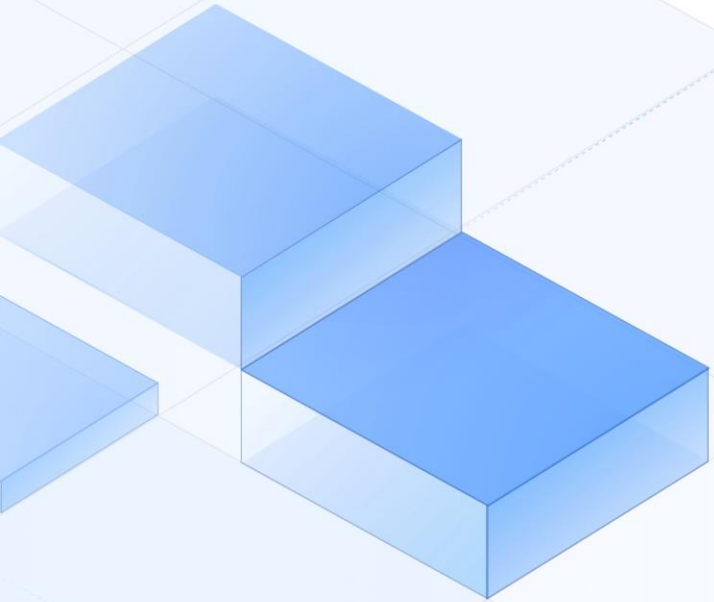
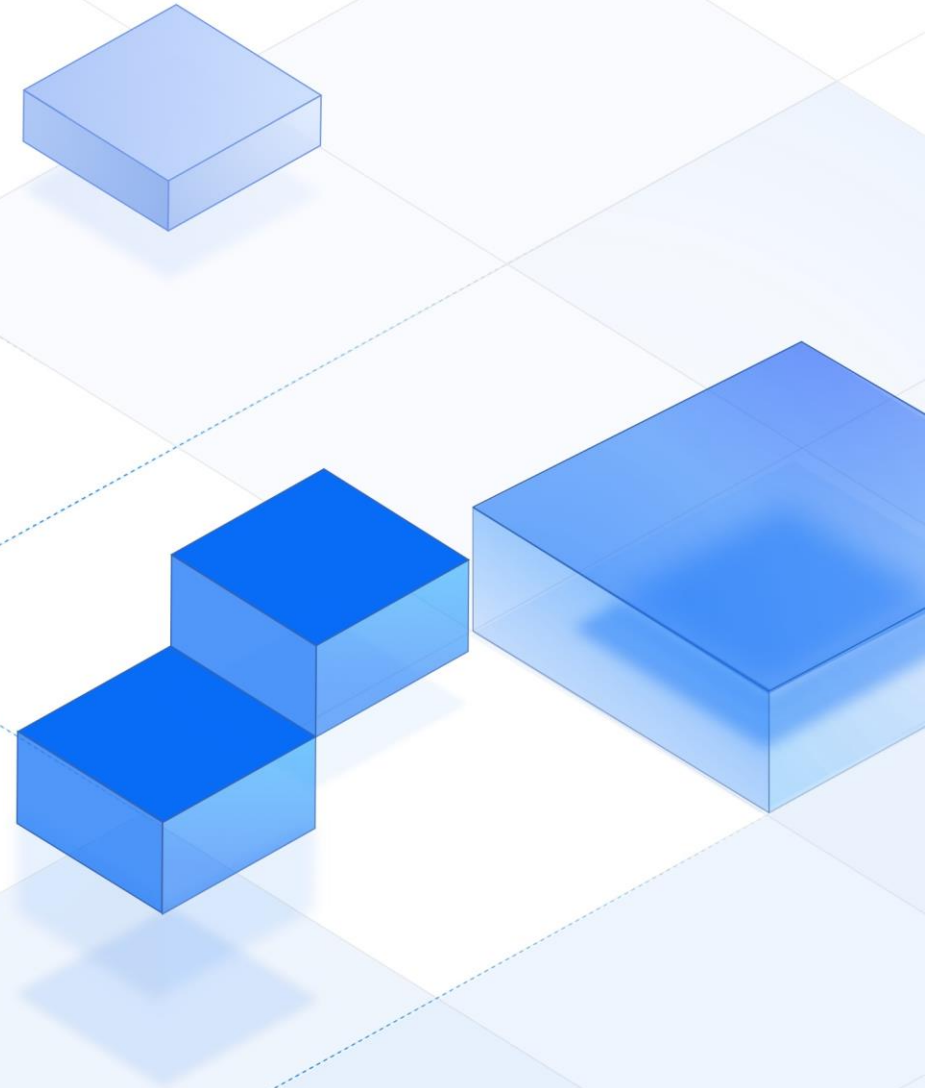
OCEANBASE

OceanBase分布式架构高级技术

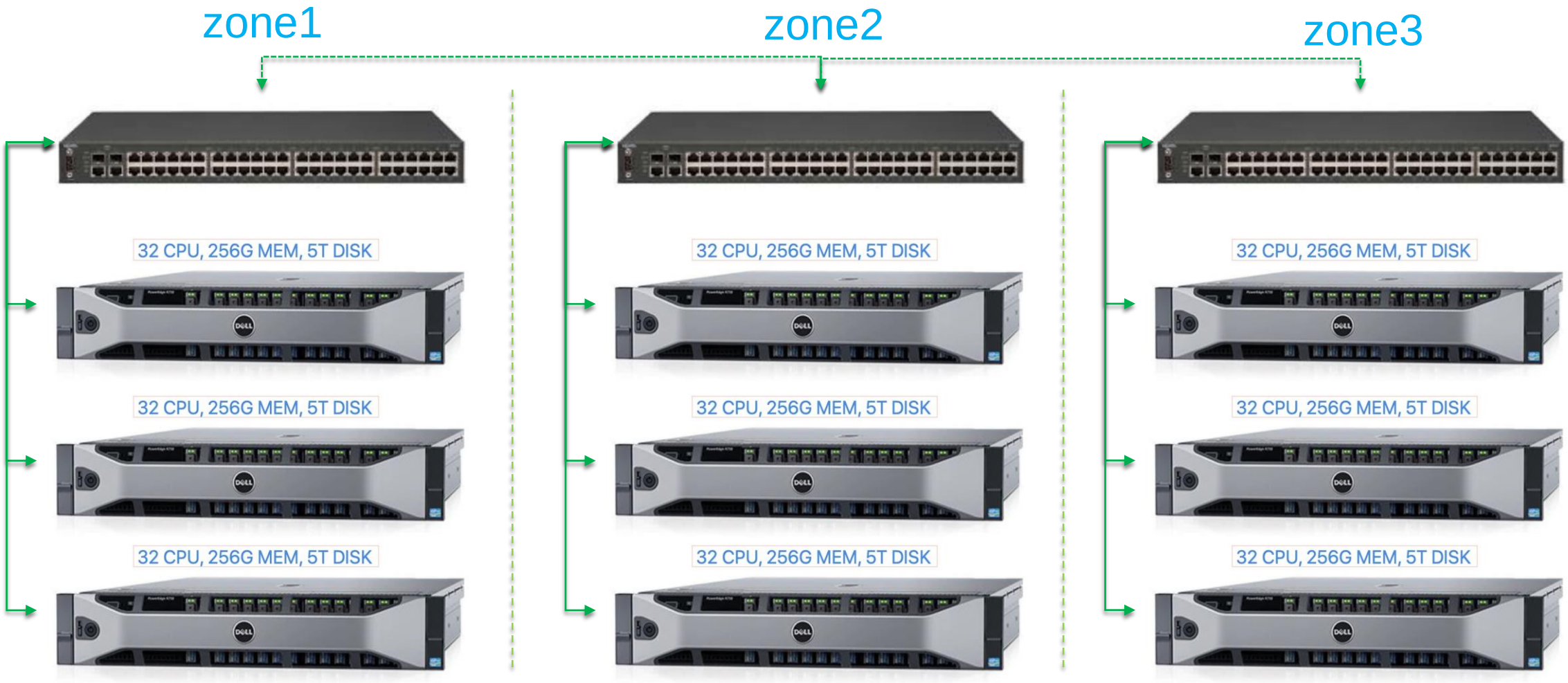
目录



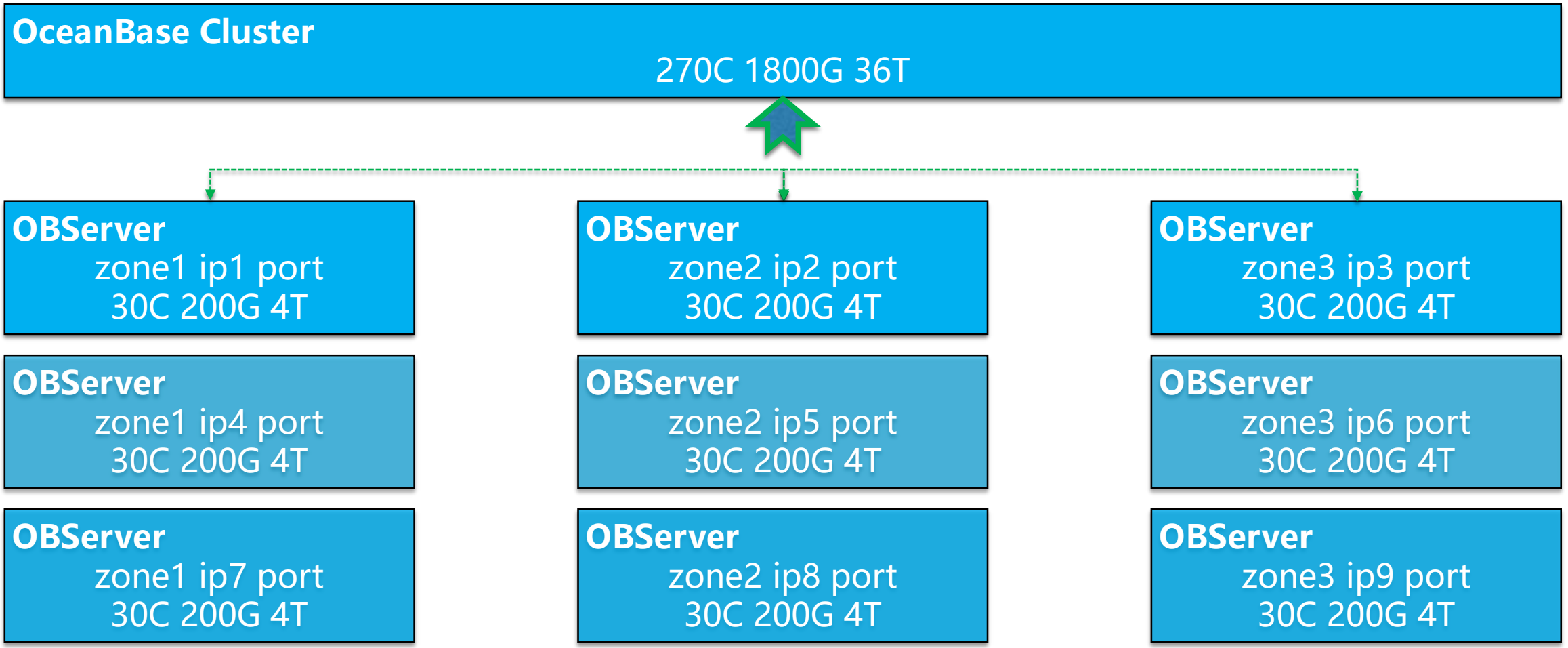
负载均衡策略



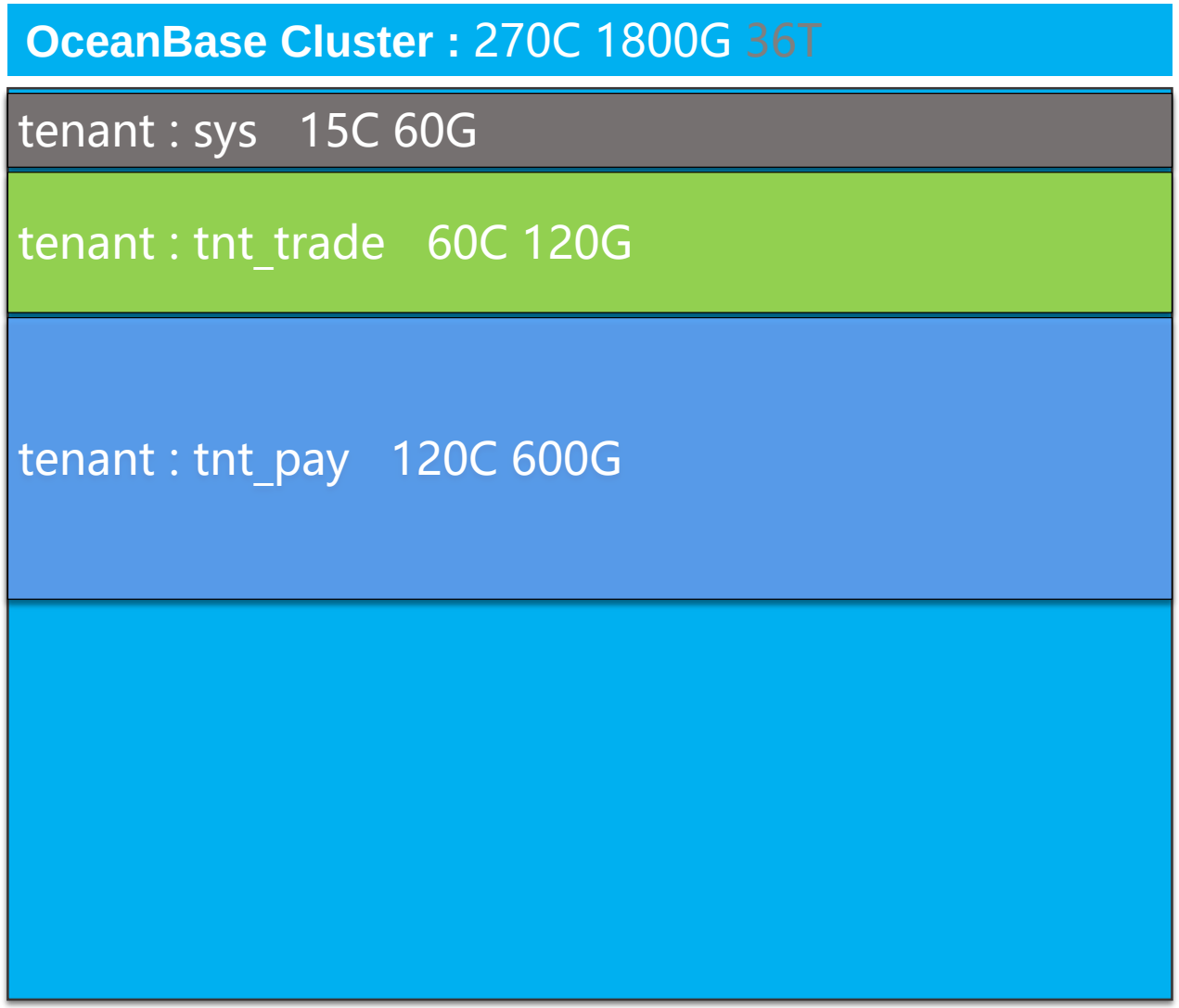
1.1 OB聚合资源的物理表示

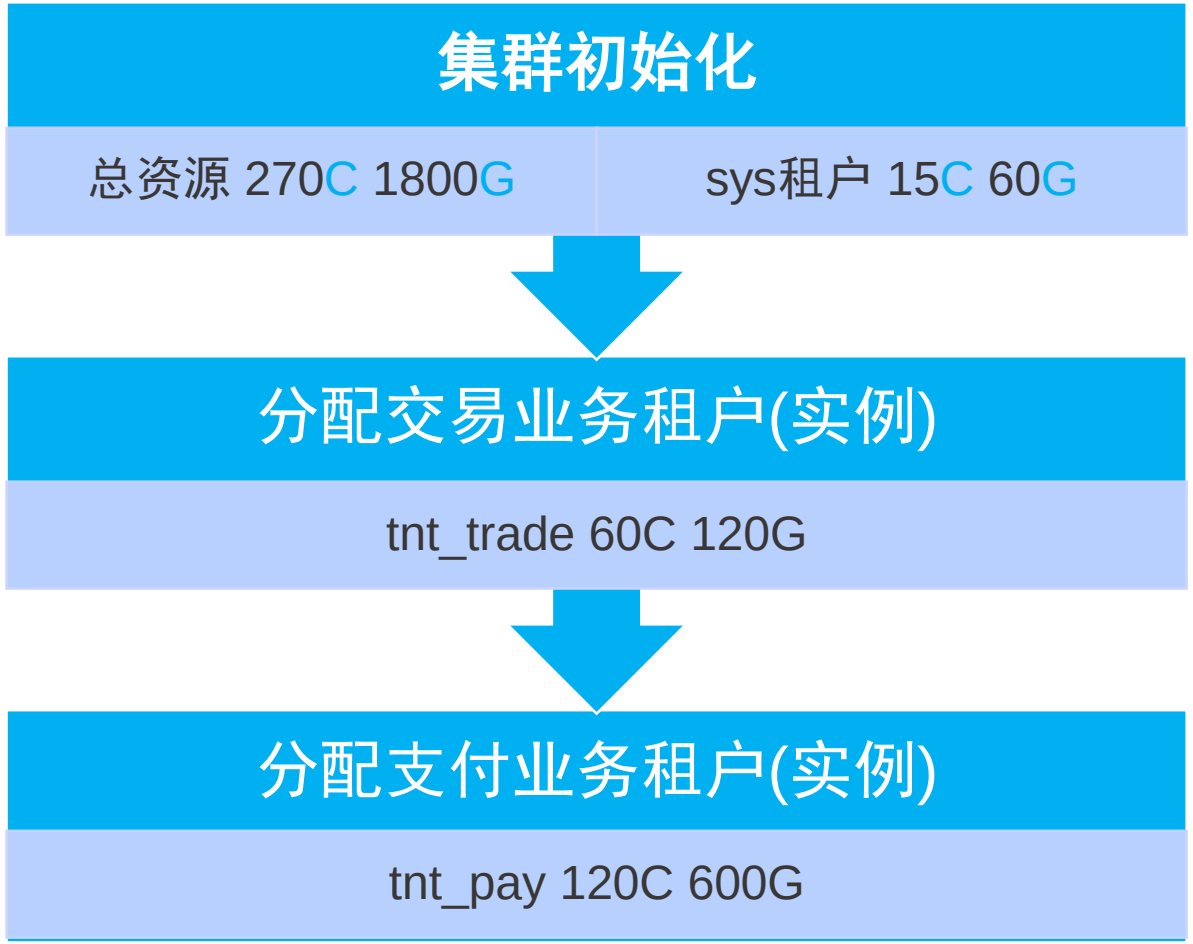


OB聚合资源的逻辑表示



OB资源的分配流程


OceanBase Cluster : 270C 1800G 36T
tenant : sys 15C 60G
tenant : tnt_trade 60C 120G
tenant : tnt_pay 120C 600G


集群初始化
总资源 270C 1800G
sys租户 15C 60G
分配交易业务租户(实例)
tnt_trade 60C 120G
分配支付业务租户(实例)
tnt_pay 120C 600G

OB资源的分配流程

- 查看集群资源由各个节点的聚合情况

```
select zone,concat(svr_ip,':',svr_port) observer,
cpu_capacity,cpu_total,cpu_assigned,cpu_assigned_percent,
mem_capacity,mem_total,mem_assigned,mem_assigned_percent,
unit_Num,round(`load`,2) `load`, round(cpu_weight,2) cpu_weight, round(memory_weight,2) mem_weight,
leader_count
from __all_virtual_server_stat order by zone,svr_ip;
```

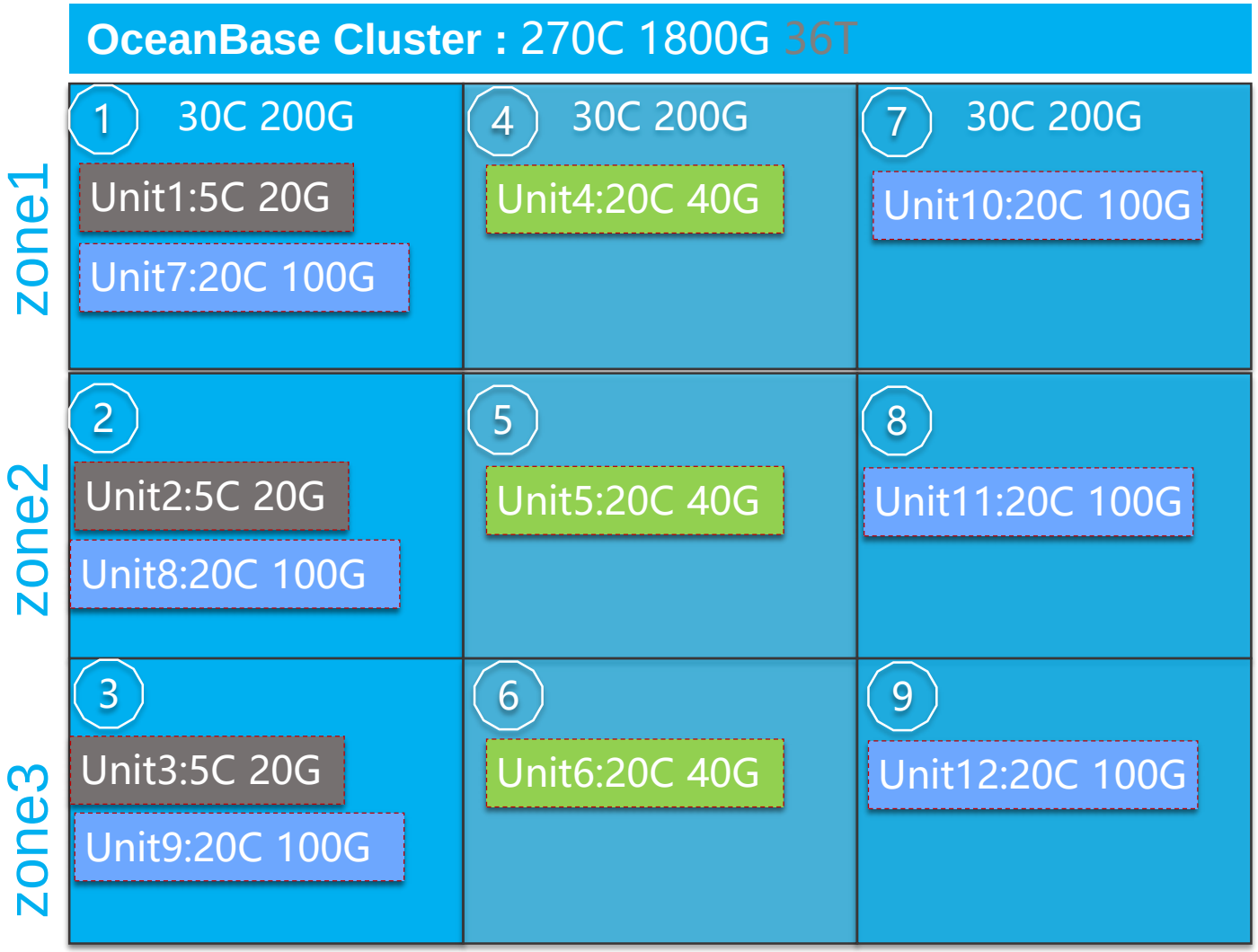
- 定义资源规格

运维人员先定义几个不同的资源规格。每个规格代表了一定的资源(包括CPU、Mem、Disk、session、IOPS）。当然实际上当前版本只对CPU和Mem资源进行限制。

```
create resource unit S2 max_cpu=20, min_cpu=20, max_memory='40G', min_memory='40G', max_iops=10000,
min_iops=1000, max_session_num=1000000, max_disk_size='1024G’;

create resource unit S3 max_cpu=20, min_cpu=20, max_memory='100G', min_memory='100G', max_iops=10000,
min_iops=1000, max_session_num=1000000, max_disk_size='1024G';
;
```

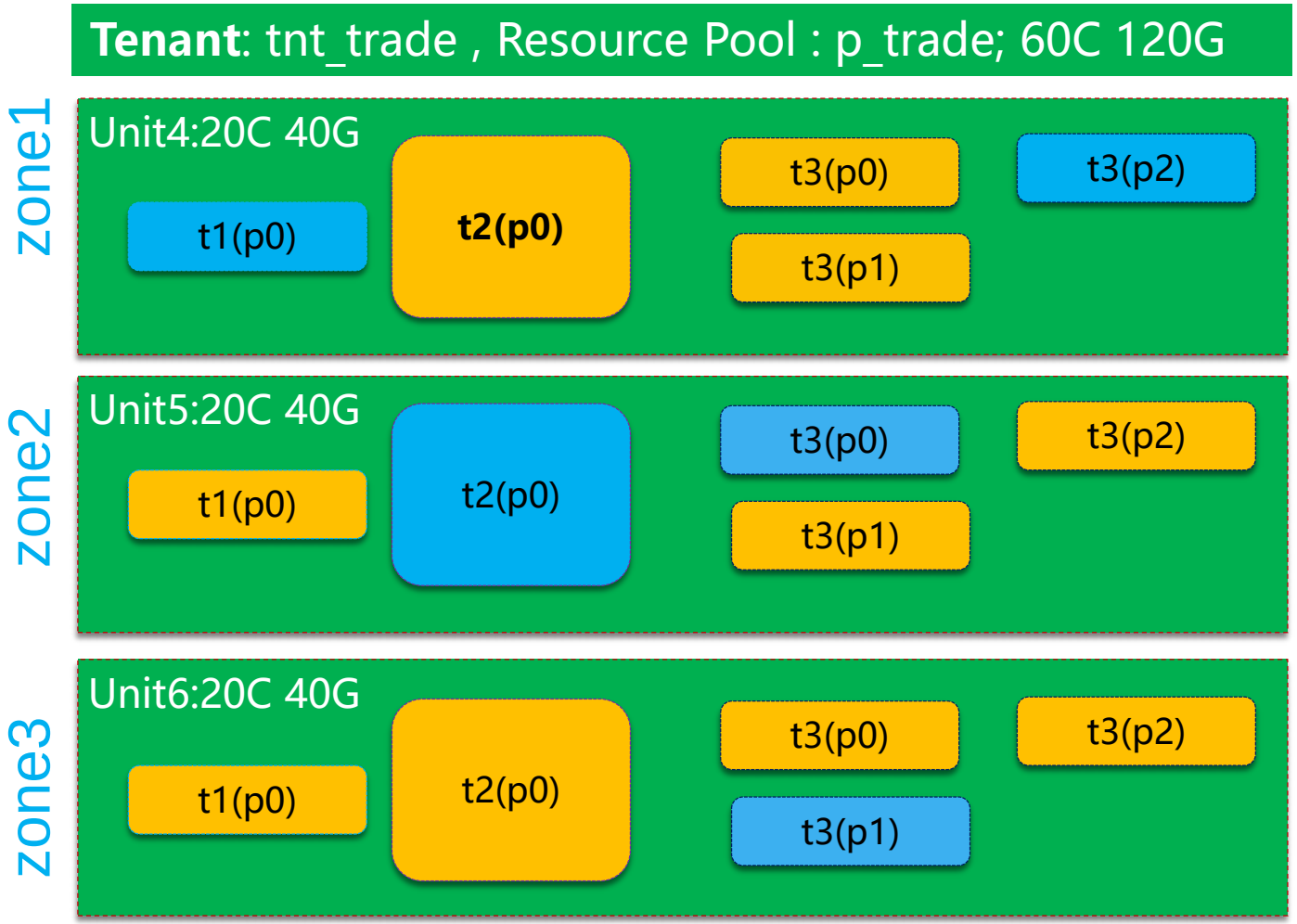
创建租户时的资源分配



- ✓ 集群初始化成功(默认租户：sys)
- ✓ create resource unit S2 max_cpu=20,max_mem=40G,…
- ✓ create resource unit S3 max_cpu=20,max_mem=100G,…
- ✓ create resource pool **p_trade** unit=S2, unit_num=1;
- ✓ create tenant tnt_trade resource pool=p_trade …
- ✓ create resource pool p_pay unit=S3, unit_num=2;
- ✓ create tenant tnt_pay resource pool=p_pay;

- 资源单元(Unit)是资源分配的最小单元，同一个Unit不能跨节点(OBServer)
- 每个租户在一台observer上只能有一个unit
- Unit是数据的容器

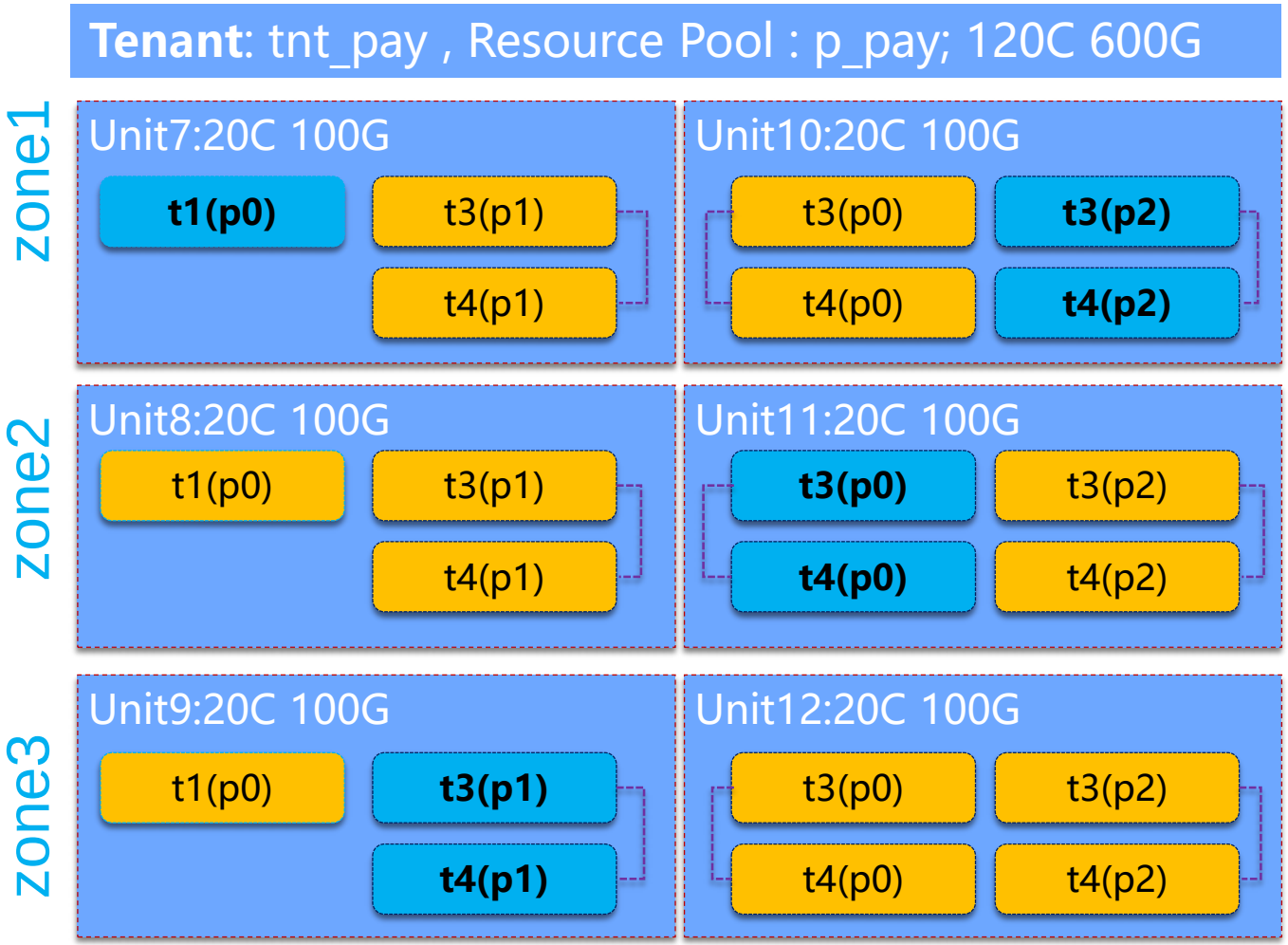
创建租户分区表时的资源分配：租户有 1 个 unit



- ✓ 资源池及租户初始化成功(tnt_trade)
- ✓ create table t1(…);
- ✓ create table t2(…) primary_zone='zone2';
- ✓ create table t3(…) partition by hash（<column name>） partitions 3;

- 每个分区有三个副本，默认leader副本提供读写服务,follower副本不提供服务。
- 每个分区的三副本内容是一样的。

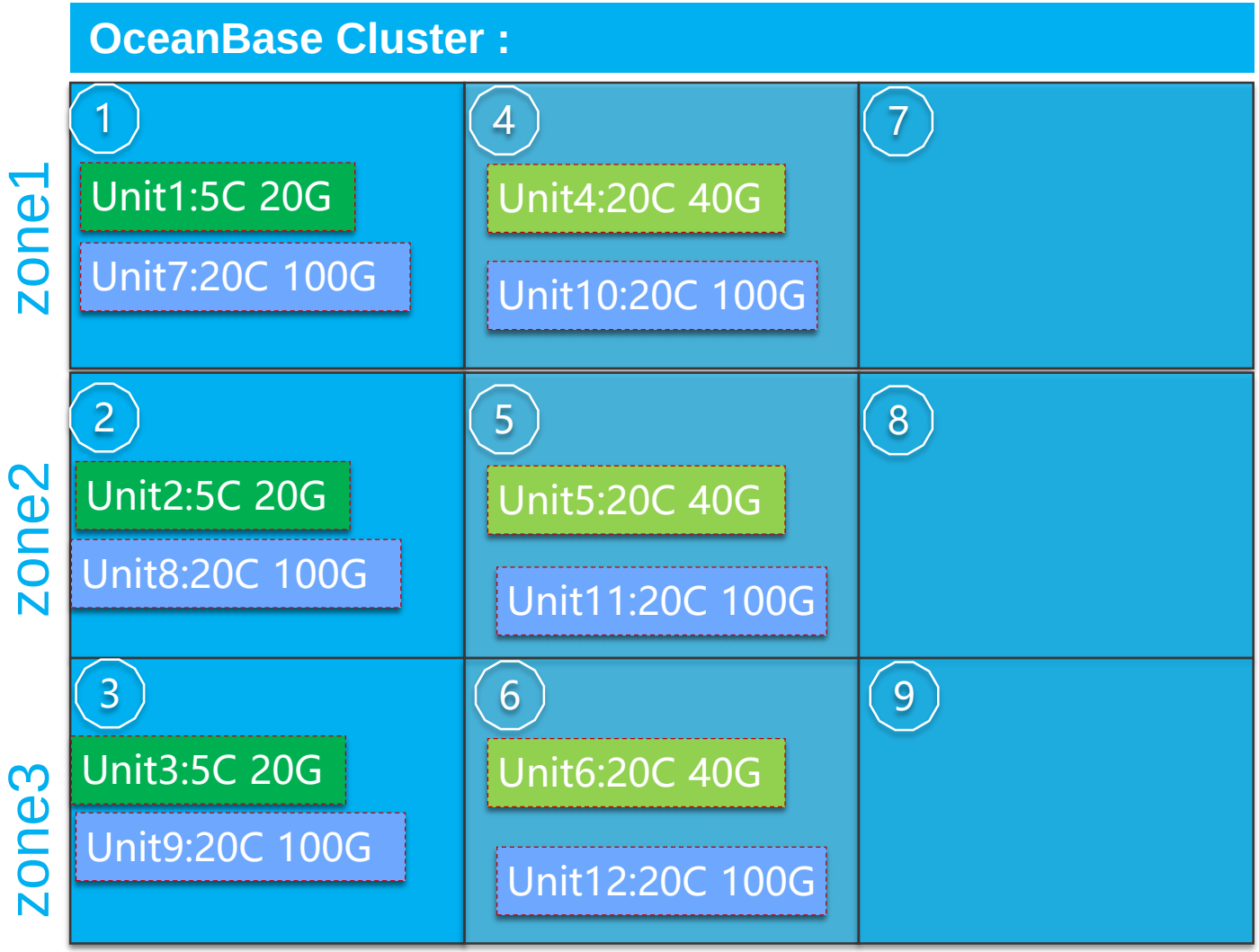
创建租户分区表时的资源分配:租户有多个 Unit



- ✓ 资源池及租户初始化成功(tnt_pay)
- ✓ create tablegroup tgorder partition by hash partitions 3;
- ✓ create table t1(…);
- ✓ create table t3(…) partition by hash(…) partitions 3 tablegroup=tgorder;
- ✓ create table t4(…) partition by hash(…) partitions 3 tablegroup=tgorder;

- 每个分区有三个副本，默认leader副本提供读写服务。
- 同一个分区不能跨Unit，同一个分区表不同分区可以跨Unit
- 同号分区组的分区会聚集在同一个Unit内部。

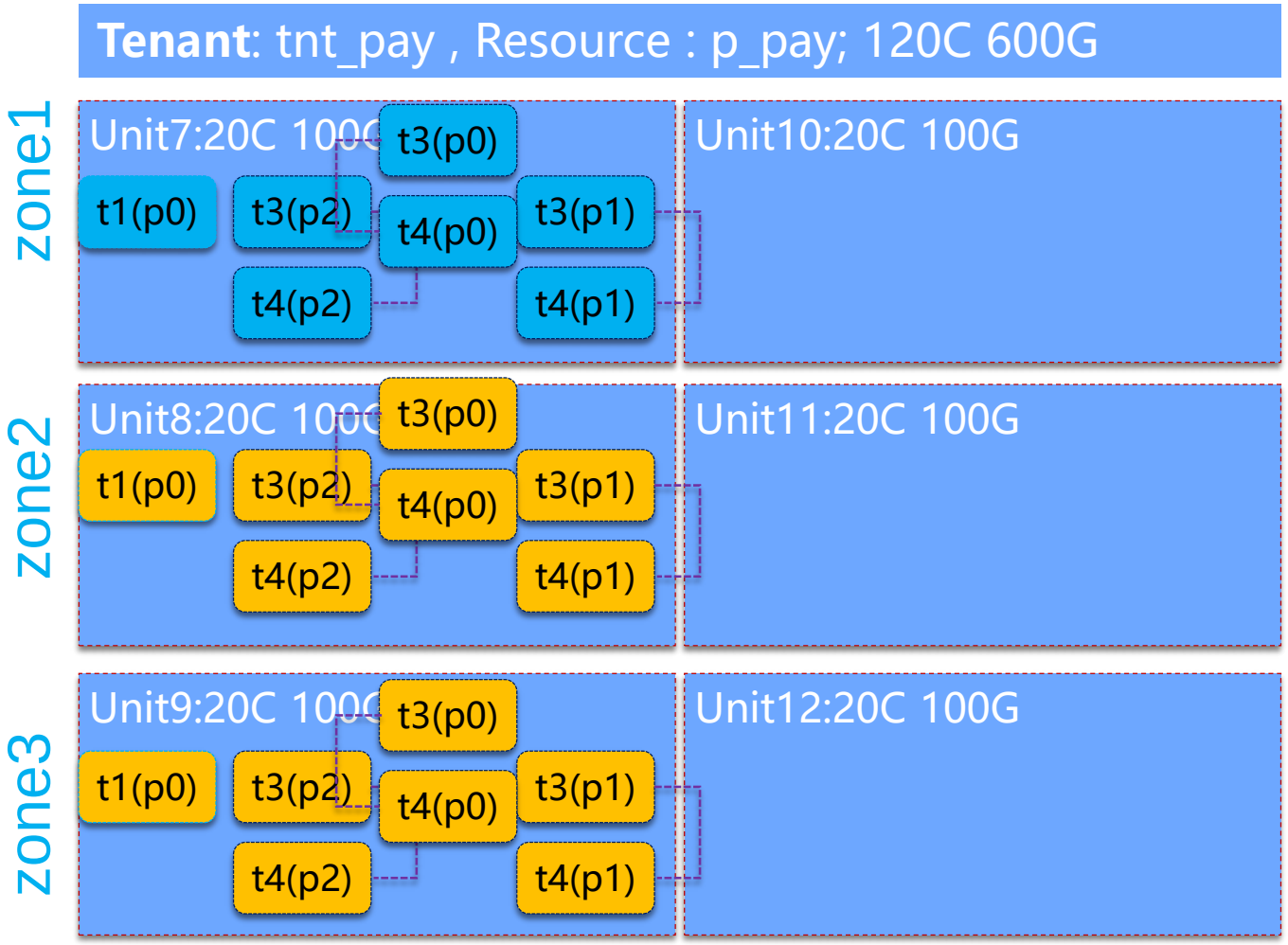
集群扩容



- ✓ 集群初始状态：2-2-2
- ✓ Unit：每个Zone 4个Unit
- ✓ 集群扩容：2-2-2 -> 3-3-3
- ✓ Unit迁移

- Unit 是资源调度的最小单元
- 图中Unit移动是示意图，实际细节是目标端先创建Unit，然后分区复制和切换。
- 参数控制 enable_rebalance 和 enable_auto_leader_switch

租户扩容



租户资源，2种方式：

规格S2升级到S3.

alter resource pool pool_mysql unit='S3';

或者不调整规格，而是增加Unit的数量：

alter resource pool pool_mysql unit_num=2;

- ✓ 租户资源初始状态：unit_num=1
- ✓ 分区分布初始状态：t1, t3, t4
- ✓ 租户资源扩容：unit_num->2
- ✓ 分区搬家，分区组聚合在一起，Leader打散

- 分区是数据迁移的最小单元，同一个分区不能跨Unit，不同分区可以跨Unit
- 同号分区组的分区稳定在同一个Unit内部。
- 分区移动的过程：先复制到另一台机器；分区服务从旧的机器切换到新的机器；再删除旧机器上的分区

OceanBase 的资源弹性伸缩与负载均衡相关参数

自动负载均衡相关参数

这个是通过参数enable_rebalance控制。同时为了控制负载均衡时Partition迁移的速度和影响，可以调整下面几个参数。

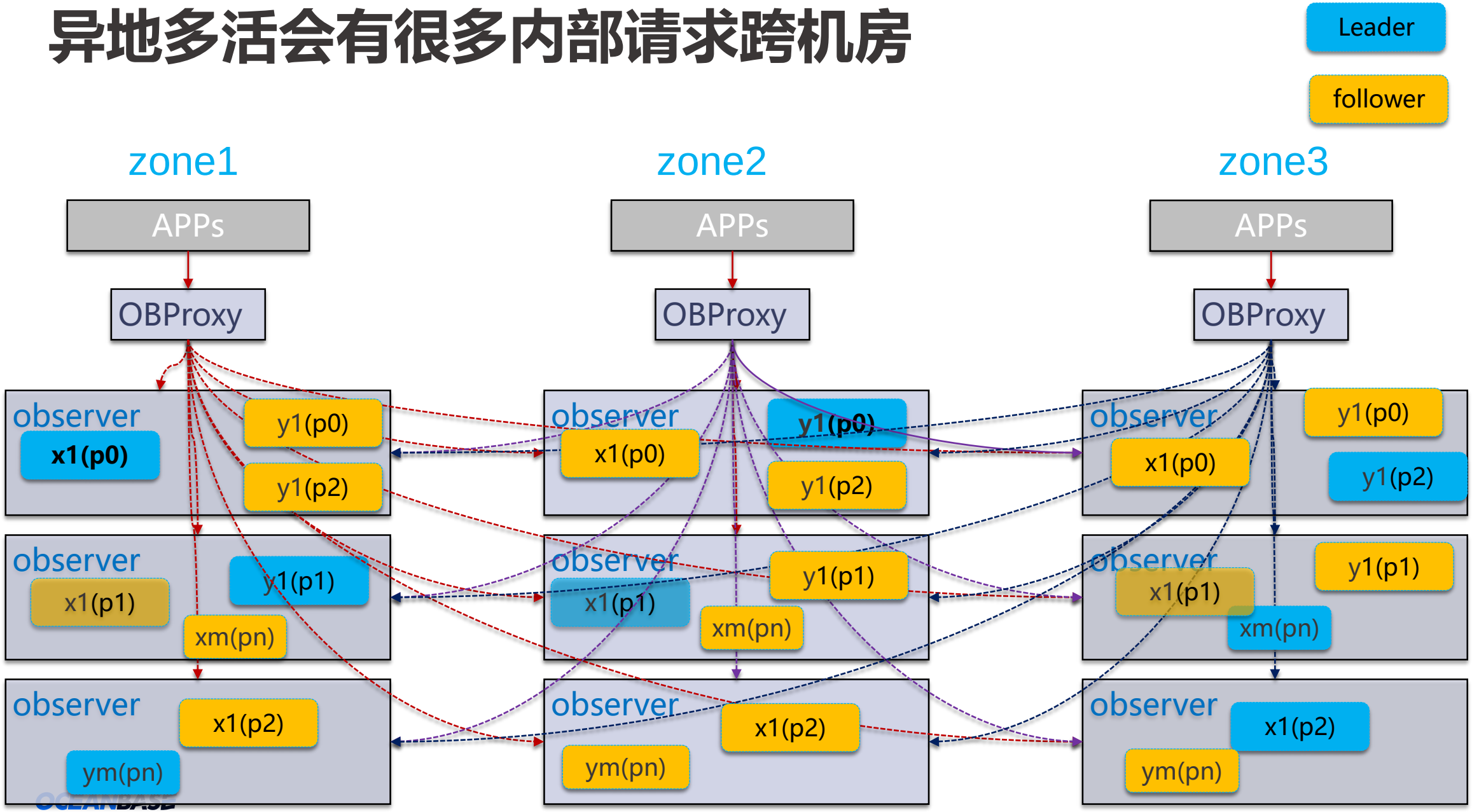
```
show parameters where name in
('enable_rebalance','migrate_concurrency','data_copy_concurrency','server_data_copy_out_concurrency','
server_data_copy_in_concurrency');
```

查看业务租户内部所有leader副本的位置

select t5.tenant_name, t4.database_name,t3.tablegroup_name,t1.table_id,t1.table_name,t2.partition_id, t2.role,
t2.zone, concat(t2.svr_ip,':',t2.svr_port) observer, round(t2.`data_size`/1024/1024) data_size_mb, t2.`row_count`
from __**all_virtual_table** t1 join **gv$partition** t2 on (t1.tenant_id=t2.tenant_id and t1.table_id=t2.table_id)
left join __all_tablegroup t3 on (t1.tenant_id=t3.tenant_id and t1.tablegroup_id=t3.tablegroup_id)
join __all_database t4 on (t1.tenant_Id=t4.tenant_id and t1.database_id=t4.database_id)
join __all_tenant t5 on (t1.tenant_id=t5.tenant_id)
where t5.tenant_id=1020 and **t2.role=1**
order by t5.tenant_name, t4.database_name,t3.tablegroup_name,t2.partition_id;

总控服务的事件日志表(__all_rootservice_event_history)

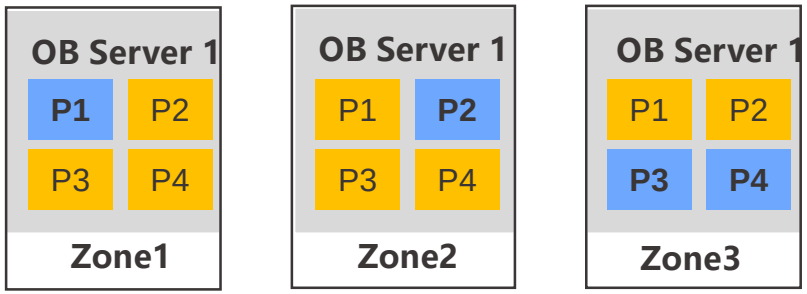
异地多活会有很多内部请求跨机房



通过Primary Zone设置优先级，适配不同业务

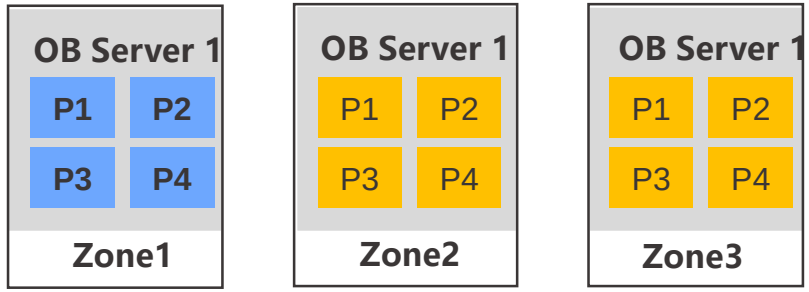
主副本

从副本



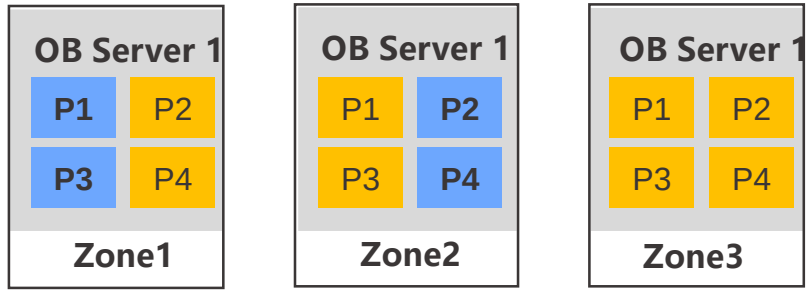
配置1：(z1,z2,z3)　意义：ZONE_1 =ZONE_2 = ZONE_3

- 主副本（Leader）均匀分布到各机器。
- 适合批处理场景，希望尽快跑完，不关注某一个sql的执行时间，期望让整体任务尽快完成。



配置2：(z1;z2;z3)　意义：ZONE_1 >ZONE_2 > ZONE_3

- 主副本（Leader）只在Zone1存在，Zone和Zone3均是从副本；
- 适合对时延敏感的在线处理业务，业务量不大，不超过一台机器的处理能力，可以尽量避免跨服务器访问，从而降低时延。



配置3：(z1,z2;z3)　意义：　(ZONE_1 = ZONE_2)　> ZONE_3

- 主副本（Leader）均匀分布到Zone1和Zone2，Zone3均是从副本
- 适合城市容灾方案，将业务汇聚到距离较近的城市，距离较远的城市只承担从副本的角色（只读）；

通过为不同的租户配置不同的Primary Zone，可以将业务流量集中到若干Zone中，减少跨服务器的操作。

Primary Zone有租户、数据库和表不同的级别

租户

(z1, z2 , z3)

- 主副本均匀分布到3个Zone中
- 满足大部分表的应用场景

优先级 <

数据库

(z1 ; z2 ; z3)

（如没有特殊指定）该数据库下的所有表的主副本将集中到 Zone1内机器中（可能是多台）；

优先级 <

表

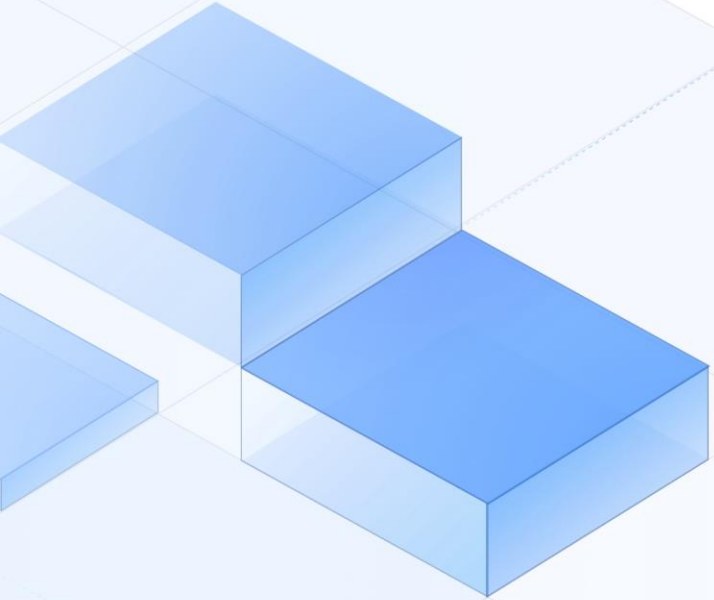
(z1,z2;z3)

主副本将集中到 Zone1和Zone2内机器中；

- **如无特殊指定，自动继承上级对象的primary_zone：database继承租户的primary_zone设置，table继承database的primary_zone设置。**
- **database和table可以指定各自的primary_zone，不必和上一级对象的设置保持一致；提供更加灵活的负载均衡策略。**

小结

1. OB 的资源分配流程是：定义资源规格 -> 创建资源池 -> 分配资源池给租户。
2. Partition自动负载均衡：同一个分区表的不同分区、租户内的所有分区、不同租户间的分区会自动调整，使得分区分布在多个维度上都达到均衡。
3. 管理员可以通过设置primary_zone，影响租户、数据库、表等对象主副本的分布策略。
4. 对于关系密切的表，可以通过表组（tablegroup）干预它们的分区分布，使表组内所有的同号分区在同一个 Unit 内部，避免跨节点请求对性能的影响。
5. Unit负载均衡：集群扩容后或缩容后，Unit自动在不同的observer之间调整，租户的数据自动在Unit之间重新均衡；整个过程在线完成，极大简化运维难度。



感谢学习