# PARTITION 研究

**Oracle 售前顾问团队**

# 什么是分区表和分区索引

今天的企业经常运行多大几百个 GB，某些时候多达几个 TB 数据的任务关键的数据库。这些企业面临需要支持和维护非常大的数据库（VLDB）的挑战，并要谋划出对付那些挑战的方法。

满足 VLDB 要求的一个方法是创建和使用分区表和分区索引。分区表允许将数据分成被称为分区甚至子分区的更小的更好管理的块。索引也可以这么分区。每个分区可以被单独管理，可以不依赖于其他分区而单独发挥作用，因此提供了一个更有利于可用小和性能的结构。

如果打算使用并行执行，则分区提供了另一种并行的方法。通过给表和索引的不同分区分配不同的并行执行服务器，就可以并行执行对分区表和分区索引的操作。

表或索引的分区和子分区都共享相同的逻辑属性。例如，表的所有分区（或子分区）共享相同的列和约束定义，一个索引的分区（或子分区）共享相同的索引选项。然而，它们可以具有不同的物理属性（如表空间）。

尽管不需要将表或索引的每个分区（或子分区）放在不同的表空间，但这样做更好。将分区存储到不同的表空间能够：

减少数据在多个分区中冲突的可能小

可以单独备份和恢复每个分区

控制分区与磁盘驱动器之间的映射（对平衡 I / O 负载是重要的）

改善可管理小，可用小和性能

分区操作对现存的应用和运行在分区表上的标准 DML 语句说是透明的。但是，可以通过 DML 中使用分区扩展表或索引的名字来对应用编程，使其利用分区的优点。

可以使用 SQL*Loader，Import 和 Export 工具来装载或卸载分区表中的数据。这些工具都是支持分区和子分区的。

# 分区的方法

ORACLE 提供了如下几种分区方法：

范围分区

散列分区

列表分区

组合分区

可对索引和表分区。全局索引只能按范围分区，但可以将其定义在任何类型的分区或非分区表上。通常全局索引比局部索引需要更多的维护。

组建局部索引，已便反映其基础表的结构。它与基础表是等同分区的，即，它与基础表在同样的列上分区，创建同样数据的分区或子分区，设置与基础表相对应的同样的分区边界。对局部索引而言，当维护活动影响分区时，会自动维护索引分区。这保证了索引与基础表之间的等同分区。

## 何时使用范围分区方法

要想将行映射到基于列值范围的分区，就使用范围分区方法。当数据可以被划分成逻辑范围时，如年度中的用份，这种类型的分区就有用了。当数据在整个范围中能被均等地划分时性能最好。如果开范围的分区会由于不均等的划分而导致分区在大小上明显不同时，就需要考虑其他分区方法。

创建范围分区时，必须指定：

分区方法：范围

分区列

标识分区边界的分区描述

如下列子创建一个 4 个分区的表，一个分区对应一个季度的销售情况，列 sale_year, sale_month 和 sale_day 是分区列，即它们的值组成一个特殊的行的分区键。VALUESLESS THAN 子加决定分区边界：分区键值比在子句中指定的，已排序的值的列表还要小的行，将被存储在分区中。每个分区都有名字（sales_q1,sales_q2,…），每个分区都在单独的表空间（tsa,tsb,…）。

```
CREATE TABLE sales
( invoce_no    NUMBER,
sale_year      INT NOT NULL,
sale_month     INT NOT NULL,
sale_day       INT NOT NULL)
PARTITION  BY  RANGE  (sale_year,sale_month,sale_day)
(PARTITION  sales_q1  VALUES LESS THAN (1999,04,01)
TABLESPACE     tsa,
PARTITION   sales_q2  VALUES LESS THAN (1999,07,01)
TABLESPACE     tsb,
PARTITION   sales_q3  VALUES LESS THAN (1999,10,01)
TABLESPACE     tsc,
PARTITION   sales_q4  VALUES LESS THAN (2000,01,01)
TABLESPACE     tsd);
```

具有 sale_year=1999,sale_month=8,sale_day=1 的行的分区键是（1999，8，1）将被存储在分区 sales_q3 中。范围分区表的每个分区都被存储在单独的段中。

注意：如果已经或将要使用具有不同字符集的数据库，那么在字符列上分区时就需要注意，因为字符的分类序列在不同字符集中是不同的。

## 何时使用散列分区方法

如果数据不那么容易进行范围分区，但为了性能和管理的原因又想分区时，就使用散列分区方法。散列分区提供 了一种在指定数量的分区中均等地划分数据的方法。基于分区键的散列值将行映射到分区中。创建和使用散列分区会给你提供了一种很灵活的放置数据的方法，因为可以通过在 I / O 驱动丘之间播撒（摘掉）这些均等定量的分区，来影响可用性和性能。

为了创建散列分区，需要指定：

分区方法：散列

分区列

分区数量或单独的分区描述

如创建一个散列分区表。分区列是 id，创建 4 个分区并赋予系统产生的名字，并且且放到 4 个命名表空间（gear1,gear2，…）。

```
CREATE TABLE scubagear
( id   NUMBER,
name       VARCHAR2(60))
PARTITION BY HASH (id)
PARTITIONS 4
STOR IN (gear1,gear2,gear3,gear4);
```

散列分区表的每个分区都被存储在单独的段中。

## 何时使用列表分区方法

当需要明确控制如何将行映射到分区时，就使用列表分区方法。可以以在每个分区的描述中为该分区列指定一列离散值。这不同于范围分区，在那里一个范围与一个分区相关，这也不同于散列分区，在那里用户不能控制如何将行映射到分区。

列表分区方法是特意为遵从离散值的模块化数据划分而设计的，范围分区或散列分区不能么容易做到这一点，因为：

范围分区为分区列假书了一个值的自然范围。这就不可能将该值的范围以外的分区组织到一起

散列分区不允许对数据的划分进行控制，因为是用系统的散列函数将数据划分到个种分区的。此外，这就不可能从逻辑上将用于分区列的离散值按分区组织到一起。

进一步说，列表分区可以非常自然地将无序的和不相关的数据集进行分组和组织到一起。

与范围分区和散列分区所不同，列表分区不支持多列分区。如果要将表按列分区，那么分区键就只能由表的一个单独的列组成。然而，可以用范围分区或散列分区方法进行分区的所有的列，都可用用列表分区方法进行分区。

当创建列表分区时，必须指定：

分区方法：列表

分区列

分区描述，每个描述指定一串文字值（值的列表），它们是分区列（它们限定将被包括在分区中的行）的离散值。

如下例子创建一个列表分区表。创建一个按地区进行分区的表 sales_by_region；即，按地理位置将州分组到一起：

```
CREATE TABLE sales_by_region
(deptno number,
deptname varchar2(20),
quarterly_sales number(10,2),
state varchar2(2))
PARTITION BY LIST(state)
(PARTITION q1_northwest VALUES(‘OR’,’WA’),
```

PARTITION q1_southwest VALUES(‘AZ’,’UT’,’NM’),
PARTITION q1_northeast VALUES(‘NY’,’VM’,’NJ’),
PARTITION q1_southeast VALUES(‘FL’,’GA’),
PARTITION q1_northcentral VALUES(‘SD’,’WI’),
PARTITION q1_southcentral VALUES(‘OK’,’TX’));

通过检查一个行的分区列的值是否属于描述该分区的值的集合，就可以将该行映射到该分区。

例如，下面的行按如下方式插入：

（10,’accounting’,100,’WA’）映射到分区 q1_northwest

（20,’R&D’,150,’OR’）映射到分区 q1_northwest

（30,’sales’,100,’FL’） 映射到分区 q1_southeast

(40,’HR’,10,’TX’) 映射到分区 q1_southwest

(50,’systems engineering’,10,’CA’)不会映射到表中的任何分区。

有关列表分区的一件有趣的事是，各个分区之间没有明显的排序（不象范围分区）。

## 何时使用组合分区方法

组合分区方法是在分区中使用范围分区方法分区数据，而在子分区中则使用散列分区方法。组合分区很适合于历史数据和条块数据两者，它改善了范围分区及其数据放置的管理性，并提供了散列分区的并行机制的优点。

当创建组合时，要指定：

分区方法：范围

分区列

标识分区边界的分区描述

子分区方法：散列

子分区列

每个分区的子分区数量，或子分区的描述

如下语句创建一个组合分区表。在本列中，创建了 3 个范围分区，每个范围分区包含 8 个子分区。因为没有给子分区命名，所以系统为其命名，但 STORE IN 子句将其划分到 4 个指定的表空间中（ts1,…,ts4）。

CREATE TABLE scubagear（equipno NUMBER, equipname VARCHAR(32),price NUMBER）

PARTITION BY RANGE（equipno）SUBPARTITION BY HASH（equipname）

SUBPARTITIONS 8 STORE IN（ts1,ts2,ts3,ts4）

（PARTITION p1 VALUES LESS THAN（1000），

PARTITION p2 VALUES LESS THAN（2000），

PARTITION p3 VALUES LESS THAN（MAXVALUE））；

组合分区表的每个子分区都被存储在自己的段中。因为它们的数据被存储在它们的子分区的段中，所以组合分区表的分区仅仅是一个逻辑结构。因为有了分区，所以这些子分区才共享相同的逻辑属性。与范围分区表中的范围分区不同，子分区不能具有与其所属分区所不同的物理属性，尽管他们不需要被存储在相同的表空间中。

# 创建分区表

除了包括一个分区方法的子句之外，创建分区表或分区索引很像创建非分区表或非分区索引。所包括的分区子句，和下级子句，取决于进行的分区的类型。

可对普通表（堆积结构表）和索引结构表两重，包括那些具有 LOB 列的表，进行分区。可以在分区表上创建非分区的全局索引、范围分区的全局索引，和局部索引。

创建（或更改）分区表时，可以指定行移行子句，即 ENABLE ROW MOVEMENT 或者 DISABLE ROW MOVEMENT。该子句启用或停用将行迁移到一个新的分区（当其键被更改时）。默认值为 DISABLE ROW MOVEMENT。

## 创建范围分区表

CREATE TABLE 语句的 PARTITION BY RANGE 子句表示将表进行范围分区。PARTITION 子句标识各个分区范围，可选的 PARTITION 子句的下级子句可以指定特别用于该分区的物理的和其他的属性。如果在分区层次上没有重载，则分区继承其基础表的属性。

如下例子比前面用于范围分区表的例子复杂一些。在表层次上指定了存储参数和 LOGGING 属性。这些属性代替了相应的从表空间层次继承的默认值（可用于表自身，并可被范围分区继承）。当然，因为第一季度没有什么业务，所以分区 sales_q1 的存储属性安排得小些，ENABLE ROW MOVEMENT 子句表示，如果键值被更改了（它会将行放置到不同的分区），就允许将行迁移到新分区。

```
CREATE TABLE sales
(invoice_no NUMBER,
sale_year INT NOT NULL,
sale_month INT NOT NULL,
sal_day INT NOT NULL)
STORAGE(INITIAL 100K NEXT 50K)LOGGING
PARTITION BY RANGE(sale_year,sale_month,sale_day)
(PARTITION sales_q1 VALUES LESS THAN(1999,04,01)
TABLESPACE tsa STORAGE (INITIAL 20K,NEXT 10K),
PARTITION sales_q2 VALUES LESS THAN(1999,07,01)
TABLESPACE tsb,
PARTITION sales_q3 VALUES LESS THAN(1999,10,01)
TABLESPACE tsc,
PARTITION sales_q4 VALUES LESS THAN(2000,01,01)
TABLESPACE tsd)
ENABLE ROW MOVEMENT;
```

创建范围分区全局索引的规则相似于创建范围分区表。下面是一个在上面的表的 sales_month 列上，创建一个范围分区全局索引的例子。每个索引分区都被命名，但存储在索引的默认表空间中。

```
CREATE INDEX month_ix ON sales(sale_month)
GLOBAL PARTITION BY RANGE(sale_month)
```

(PARTITION pm1_ix VALUES LESS THAN(2),
PARTITION pm2_ix VALUES LESS THAN(3),
PARTITION pm3_ix VALUES LESS THAN(4),
PARTITION pm4_ix VALUES LESS THAN(5),
PARTITION pm5_ix VALUES LESS THAN(6),
PARTITION pm6_ix VALUES LESS THAN(7),
PARTITION pm7_ix VALUES LESS THAN(8),
PARTITION pm8_ix VALUES LESS THAN(9),
PARTITION pm9_ix VALUES LESS THAN(10),
PARTITION pm10_ix VALUES LESS THAN(11),
PARTITION pm11_ix VALUES LESS THAN(12),
PARTITION pm12_ix VALUES LESS THAN(MAXVALUE));

## 创建散列分区表

CREATE TABLE 语句的 PARTITION BY HASH 子句表示将进行散列分区。PARTITIONS 子句可用于指定要创建的分区的数量，可选地，以及用于存储这些分区的表空间。换句话说，可以使用 PARTITION 子句来命名各个分区及其表空间。

可以为散列分区指定的属性只有 TABLESPACE。一个表的所有散列分区都必须共享从表层次上继承来的相同的属性（除 TABLESPACE 之外）。

下面的例子说明了创建名为 DEPT 的散列分区表的 2 种方法。在第一个例子中指定了分区的数量，但是由系统给它们命名，并将它们存储在该表的默认表空间中。

CREATE TABLE dept(deptno NUMBER,deptname VARCHAR(32))
PARTITION BY HASH(deptno) PARTITONS 16;

在这第二个例子中，指定了各个分区的名字及其所属的表空间。在表层次上还明确地为每个散列分区（段）指定了初始盘区的大小，并且所有分区都继承了这个属性。

CREATE TABLE dept(deptno NUMBER,deptname VARCHAR(32))
STORE(INITIAL 10K)
PARTITION BY HASH(deptno)
(PARTITION p1 TABLESPACE ts1,
PARTITION p2 TABLESPACE ts2,
PARTITION p3 TABLESPACE ts3,
PARTITION p4 TABLESPACE ts4);

如果为上面的表创建局部索引，则 ORACLE 会创建一个与基础表等同分区的索引。ORACLE 还保证，当对基础表进行维护时，该索引也被自动维护。下面是一个在表 DEPT 上创建局部索引的例子。CREATE INDEX locd_dept_ix ON dept(deptno) LOCAL

可选地，可以对散列分区和存储局部索引分区的表空间进行命名。如果没有做这件事，ORACLE 就将对应的基础分区的名字作为索引分区的名字，并将索引分区存储在表分区的相同的表空间中。

## 创建列表分区表

创建列表分区的过程很像创建范围分区，但创建列表分区时，在 CREATE TABLE 语句中是指定 PARTITION BY LIST 子句，并且该 PARTITION 子句会指定一串文字值，它会是分区列的离散值（限定在分区中包括的行）。与范围分区相同 ，PARTITION 子句的，可选的下级子句可以为分区段指定特殊的物理的和其他的属性。如果在分区层次上没有重载，则分区继承其基础表的属性。

在如下例子中创建表 sales_by_region，并用列表分区方法将其分区。前面 2 个 PARTITION 子句指定物理属性，它们重载了表层次的默认专科肆 PARTITION 子句没有指定属性，那些分区就从表层次上继承它们的默认的物理属性。

```
CREATE TABLE sales_by_region(item# INTEGER, qty INTEGER,
Store_name VARCHAR(30), state_code VARCHAR(2),
Sale_date DATE)
STORAGE(INITIAL 10K NEXT 20K) TABLESPACE tbs5
PARTITION BY LIST( state_code)
(
PARTITION region_east
VALUES('MA','NY','CT','NH','ME','MD','VA','PA','NJ')
STORAGE(INITIAL 20K NEXT 40K PCTINCREASE 50)
TABLESPACE tbs8,
PARTITION region_west
VALUES('CA','AZ','NM','OR','WA','UT','NV','CO')
PCTINCREASE 20 NOLOGGING,
PARTITION region_south
VALUES('TX','KY','TN','LA','MS','AR','AL','GA'),
PARTITION region_central
VALUES('OH','ND','SD','MO','IL','MI',null,'IA')
);
```

## 创建组合分区表

为了创建组合分区表，先用 CREATE TABLE 语句的 PARTITION BY RANGE 子句，然后指定一个与 PARTITION BY HASH 语句遵从相似语法和规则的 SUBPARTITON BY HASH 子句。各个 PARTITON 后面紧跟 SUBPARTITON 或 SUBPARTITONS 子句。

为（范围）分区指定的属性适用于那个分区的所有子分区。可以为每个（范围）分区指定不同的属性，并且，如果分区的子分区要分布的表空间与其他分区不同的话，还可以在分区层次上指定 STORE IN 子句。下面的例子对此作解释。

```
CREATE TABLE emp (deptno NUMBER, empname VARCHAR(32),grade NUMBER)
PARTITION BY RANGE(deptno) SUBPARTITION BY HASH(empname)
SUBPARTITIONS 8 STORE IN(ts1,ts3,ts5,ts7)
(PARTITION p1 VALUES LESS THAN（1000） PCTFREE 40,
PARTITION p2 VALUES LESS THAN（2000） STORE IN(ts2,ts4,ts6,ts8),
```

P PARTITION p3 VALUES LESS THAN（MAXVALUE）
(SUBPARTITON p3_s1 TABLESPACE ts4,
SUBPARTITON p3_s2 TABLESPACE ts5);
如下语句在 emp 表上创建一个局部索引，该表的索引段分布在表空间 ts7,ts8 和 ts9。
CREATE INDEX emp_ix ON emp(deptno)
LOCAL STORE IN (ts7,ts8,ts9)；
该局部索引在如下方面与基础表是等同分区的：

分区数量与基础表相同。

每个索引分区的子分区数量与相应的基础表分区相同。

在给定的基础表的子分区中的行的索引项，被存储在该索引的相应的子分区中。

## 创建分区索引结构表

可以对索引结构表使用范围分区或散列分区方法。但只有范围分区索引结构表才能包含 LOB 数据类型的列。创建范围分区或散列分区索引结构表的过程与创建普通表相似，区别在于：

创建该表时要指定 ORGANIZATION INDEX 子句，需要时还要指定 INCLUDING 和 OVERFLOW 子句。

PARTITION 或 PARTITION 子句可以有 OVERFLOW 下级子句，它允许你在分区层次上指定溢出段的属性。

指定 OVERFLOW 子句使得溢出数据段本身是与主键索引段等同分区的。因此，对具有溢出数据段的分区索引结构表而言，每个分区具有一个索引段和一个溢出数据段。

索引结构表的分区列的集合必须是主键列的子集。因为索引结构表的行是按表的主键索引存储的，所以分区方法的标准对可用性有作用。通过将分区键选成主键的子集，插入操作就只需要校验在单个分区中的主键的唯一性，因此，对分区的维护就不互不依赖了。

对索引结构表的辅助索引的支持相似于对普通表的支持，但是，某些维护操作不会将全局索引标记成 UNUSABLE，这不同于普通表。

## 创建范围分区索引结构表

可以用范围分区方法对索引结构表，及其辅助索引进行分区。如下例子创建一个范围分区索引结构表 sales。INCLUDING 子句指定将 week_no 列之后的所有列都存储溢出段中。每个分区有一个溢出段，都存储在相同的表空间（overflow_here）中。可选地，可以在各个分区层次上指定 OVERFLOW TABLESPACE，这样，某个或所有溢出段可以有不同的 TABLESPACE 属性。

CREATE TABLE sales(acct_no NUMBER(5),
acct_name CHAR(30),
amout_of _sale NUMBER(6),
week_no INTEGER,
sale_details VARCHAR2(1000),
PRIMARY KEY(acct_no,acct_name,week_no))

```
ORGANIZATION INDEX
INCLUDING week_no
OVERFLOW TABLESPACE overflow_here
PARTITION BY RANGE(week_no)
(PARTITON VALUES LESS THAN(5)
TABLESPACE ts1,
PARTITION VALUES LESS THAN(9)
TABLESPACE ts2 OVERFLOW TABLESPACE overflow_ts2,
…
PARTITION VALUES LESS THAN (MAXVALUE)
TABLESPACE ts13);
```

## 创建散列分区索引结构表

对索引结构表进行分区的其他做法是使用散列分区方法。如下例子用散列分区方法对索引结构本 sales 进行分区。

```
CREATE TABLE sales(acct_no NUMBER(5),
acct_name CHAR(30),
amount_of_sale NUMBER(6),
week_no INTEGER,
sale_details VARCHAR2(1000),
PRIMARY KEY(acct_no,acct_name,week_no))
ORGANIZATION INDEX
INCLUDING week_no
OVERFLOW
PARTITION BY HASH(week_no)
PARTITONS 16
STORE IN(ts1,ts2,ts3,ts4)
OVERFLOW STORE IN(ts3,ts6,ts9);
```

注意：因为一个设计成熟的散列函数被认为会将行在各个分区中作很好的平衡分布，所以，更改一个行的主键列就很可能将该行移动到不同的分区。因此，建议在创建一个具有可变的分区键的散列分区索引结构表时，明确指定 ROW MOVEMENT ENABLE 子句。默认时该特征是停用的。

# 用于多个数据块大小的分区限制

当在具有多个数据块大小的表空间的数据库中创建分区对象时要留意。将分区对象对象存储到这些表空间中时，会受到某些限制。特别的，如下物化的所有分区，必须存储在具有相同数据块大小的表空间中：

常规表

索引

索引结构表的主键索引段

索引结构表的溢出段

在外存储的 LOB 列

因此

每个常规表的所有分区，必须被存储在具有相同数据块大小的表空间中。

每个索引结构表的所有主键索引分区，必须被存储在具有相同 数据块大小的表空间中，并且，那个表的所有溢出分区也必须被存储在具有相同数据块大小的表空间中。当然，索引分区和溢出分区可以存储在具有不同数据块大小的表空间中。

每个索引(全局或局部)的每个分区必须被存储在具有相同数据块大小的表空间中。当然，定义在同一个对象上的不同的索引的分区可以被存储在具有不同数据块大小的表空间中。

每个 LOB 列的每个分区必须被存储在具有相同 数据块大小的表空间中。当然，不同的 LOB 列可以被存储在具有 不同数据块大小的表空间中。

当创建或更改分区表或分区索引时，为每个物化的分区和子分区明确指定的所有表空间，都必须具有相同的数据块大小。如果没有为一个物化明确指定存储表空间，那么，ORACLE 默认使用的表空间，就必需具有相同的数据块大小。因此，必须在分区对象的每个层次上留意默认表空间。

# 维护分区表

下表列出了可以对表的分区（或子分区）进行的维护操作，并且，对每种分区类型，或出了用于执行维护操作的 ALTER TABLE 语句的特殊子句。

| 维护操作 | 范围 | 散列 | 列表 | 组合 |
|---|---|---|---|---|
| 增加分区 | ADD PARTITION | ADD PARTITION | ADD PARTITION | ADD PARTITION MODIFY PARTITION … SUBPARTITION |
| 合并分区 | N/A | CONALESC PARTITION | N/A | MODIFY PARTITION … COALESCE SUBPARTITION |
| 删除分区 | DROP PARTITION | N/A | DROP PARTITION | DROP PARTITION |
| 交换分区 | EXCHANGE PARTITION | EXCHANGE PARTITION | EXCHANGE PARTITION | EXCHANGE PARTITION EXCHANGE SUBPARTITION |
| 融合分区 | MERGE PARTITIONS | N/A | MERGE PARTITIONS | MERGE PARTITIONS |
| 修改分区 增加值 | N/A | N/A | MODIFY PARTITION … ADD VALUES | N/A |
| 修改分区 删除值 | N/A | N/A | MODIFY PARTITION … DROP VALUES | N/A |

| | | | | |
|---|---|---|---|---|
| 修改默认属性 | MODIFY DEFAULT ATTRIBUTES | MODIFY DEFAULT ATTRIBUTES | MODIFY DEFAULT ATTRIBUTES | MODIFY DEFAULT ATTRIBUTES MODIFY DEFAULT ATTRIBUTES FOR PARTITION |
| 修改分区的真实属性 | MODIFY PARTITION | MODIFY PARTITION | MODIFY PARTITION | MODIFY PARTITION MODIFY SUBPARTITION |
| 移动分区 | MOVE PARTITION | MOVE PARTITION | MOVE PARTITION | MOVE PARTITION |
| 更名分区 | RENAME PARTITION | RENAME PARTITION | RENAME PARTITION | RENAME PARTITION |
| 分割分区 | SPLIT PARTITION | N/A | SPLIT PARTITION | SPLIT PARTITION |
| 舍弃分区 | TRUNCATE PARTITION | TRUNCATE PARTITION | TRUNCATE PARTITION | TRUNCATE PARTITION TRUNCATE SUBPARTITION |

下表列出了可以对索引分区进行的维护操作，并且指明了可以对何种练习索引（全局或局部）执行这些维护操作。显示了可用于维护操作的 ALTER INDEX 子句。

| 维护操作 | 索引类型 | 索引分区的类型 | | |
|---|---|---|---|---|
| | | 范围 | 散列 / 列表 | 组合 |
| 删除索引分区 | 全局 | DROP PARTITION | | |
| | 局部 | N / A | N / A | N / A |
| 修改索引分区的默认属性 | 全部 | MODIFY DEFAULT ATTRIBUTES | | |
| | 局部 | MODIFY DEFAULT ATTRIBUTES | MODIFY DEFAULT ATTRIBUTES | MODIFY DEFAULT ATTRIBUTESMODIFY DEFAULT ATTRIBUTES FOR PARTITION |
| 修改索引分区的真实属性 | 全局 | MODIFY PARTITION | | |
| | 局部 | MODIFY PARTITION | MODIFY PARTITION | MODIFY PARTITION MODIFY SUBPARTITION |
| 重建索引分区 | 全局 | REBUILD PARTITION | | |
| | 局部 | REBUILD PARTITION | REBUILD PARTITION | REBUILD SUBPARTITION |
| 更名索引分区 | 全局 | RENAME PARTITION | | |
| | 局部 | RENAME PARTITION | RENAME PARTITION | RENAME PARTITION RENAME SUBPARTITION |
| 分割索引分区 | 全局 | SPLIT PARTITION | | |
| | 局部 | N / A | N / A | N / A |

全局索引不反映基础表的结构，如果要分区，就只能对其进行范围分区。范围分区索引共享某些（并非全部）可以在范围分区表上进行的分区维护操作。

因为局部索引反映基础表的结构，因此当对表的分区和子分区进行维护活动时，会自动地对局部索引的分区进行 维护。因此不太需要对局部分区进行维护，选项也很少。

注意：只有非空的索引和索引分区可以被标记成 UNUSABLE。如果是空的，则 USABLE / UNUSABLE 就处于释放状态。

只有处于 USABLE 状态的索引或索引分区可以被后续 DML 语句更改。

# 自动更新全局索引

在讨论对分区表和分区索引的各种维护操作之前，最好先 4 讨论一下可以在 ALTER TABLE 语句中指定的 UPDATE GLOBAL INDEX,子句的作用。

默认时，对分区表的许多表维护操作会使全局索引不可用（标记成 UNUSABLE）。那么，就必须重建整个全局索引或其全部分区（如果已被分区）。Oracle 允许在用于维护操作的 ALTER TABLE 语句中，指定 UPDATE GLOBAL INDEXES，来重载这个默认特性。指定这个子句也就告诉 Oracle，当它执行维护操作的 DDL 语句时，更新全局索引。这提供了以下好处：

• 在操作基层表的同时更新全局索引。这就不需要后来单独地重建全局索引。

• 因为没有被标记成 UNUSALBE，所以全局索引的可用性更高了。甚至在执行分区的 DDL 语句时，仍然可用索引来访问表中的其他分区。

• 避免了查询所有失效的全局索引的名字，以八重建它们。

另外，在指定 UPDATE GLOBAL INDEXES 之前，还要考虑我下性能因素：

• 因为要更新事先被标记成 UNUSABLE 的索引，所以分区的 DDL 语句要执行更长时间。当然，这要与先不要更新索引而执行 DDL，然后再重建索引所花的时间做个比较。一个适用的规则是，如果分区的大小小于表的大小的 5%，则更新索引更快点。

• DROP、TRUNCATE 和 EXCHANGE 操作也不那么快了。同样，这必须与先执行 DDL，然后主重建所有全局索引所花的时间做个比较。

• 要登记对索引的更新，并产生重做记录和撤消记录。重建整个索引时，可选择 NOLOGGING。

• 重建整个索引产生一个更有效的索引，因为这更利用使用空间。再者，重建索引时允许修改存储选项。

注意：分区索引结构表不 UPDATE 支持 GLOBAL INDEXES 子句。

如下操作 UPDATE 支持 GLOBAL INDEXES 子句：

- ADD PARTITION|SUBPARTITION（仅限于散列分区）
- COALESCE PARTITION|SUBPARTITION
- DROP PARTITION
- EXCHANGE PARTITION|SUBPARTITION
- MERGE PARTITION
- MOVE PARTITION|SUBPARTITION
- SPLIT PARTITION
- TRUNCATE PARTITION|SUBPARTITION

# 增加分区

## 给范围分区表增加分区

用 ALTER TABLE…ADD PARTITION 语句将新分区增加到“高”端（最后一个现存分区的后面）。要想在表的开始或中间增加分区，就要使用 SPLIT PARTITION 子句。

例如，考虑表 sales，它包括当前月份和前 12 个月份的数据。在 1999 年 1 月 1 日，你为 1 月份增加了一个分区，并将其存储在表空间 tsx 中。

```
ALTER TABLE sales
ADD PARTITION jan96 VALUES LESS THAN (’01-FEB-1999’)
TABLESPACE tsx;
```

与范围份区表相关的局部索引和全局索引仍然可用。

## 给散列分区表增加分区

当给散列分区表增加一个新分区时，Oracle 会在新分区中放入，来自一个现存分区（由 oracle 选 定的）的，由散列函数决定的，被重新做过散列操作后的行。

下面的例子显示了给表 scubagear 增加一个散列分区的两种方法。第一个语句增加一个新散列分区，其分区名是由系统产生的，并被放置在表的默认表空间中。第二个语句也增加一个新散列分区，但分区被明确命名为 p_namedi 达并被创建到表空间 gear5 中。

```
ALTER TABLE scubagear ADD PARTITION;
ALTER TABLE scubagear
ADD PARTITION p_named TABLESPACE gear5;
```

按下表所解释的，索引可能被标记成 UNUSABLE；

| 表类型 | 索引特性 |
| --- | --- |
| 普通的（堆积的）<br><br>索引结构的 | • 用于新分区的和用于现存分区的，重新划分行的局部索引被标记成 UNUSABLE，并必须被重建。<br>• 除非指定 UPDATE GLOBAL INDEXES，否则所有全局索引或分区全局索引的所有分区，被标记成 UNUSABLE，并必须被重建<br>• 局部索引与普通表一样<br>• 所有全局索引保持可用。 |

## 给列表分区表增加分区

如下语句解释，给列表分区表增加一个新分区。本例中，给将增加的分区指定了物理属性和 NOLOGGING。

```
ALTER TABLE sales_by_region
ADD PARTITION region_nonmainland VALUES (‘HI’,’PR’)
STORAGE (INITIAL 20K NEXT 20K) TABLESPACE tbs_3 NOLOGGIN;
```

描述将要增加的分区的文字值的集合中任何值，都不能在该表的任何其他分区中存在。
与列表分区表相关的局部索引和全局索引仍然可用。

## 给组合分区表增加分区

可以在范围分区层次和散列子分区层次两个层次上增加分区。

增加分区　　给组合分区表增加一个范围分区与前面在“给范围分区表增加分区”中描述的一样。然而，可以使用 SUBPARTITIONS 子句，来增加指定数量的子分区，或使用 SUBPARTITION 子句来给特殊子分区命名。如无指定 SUBPARTITIONS 或 SUBPARTITION 子句，则该分区继承用于子分区的表层次的默认值。

本例给表 sales 增加一个范围分区 q1_2000，用于放置 2000 年第一季度的数据。在表空间 tbs5 中有 8 个子分区。

ALTER TABLE sales ADD PARTITION q1_2000

VALUES LESS THAN (2000,04,01) SUBPARTITIONS 8 STORGE IN tbs5;

增加子分区

可以用 ALTER TABLE 语句的 MODIFY PARTITON…ADD SUBPARTITION 子句，给组合分区表增加散列子分区。在新增加的子分区中，放入来自相同分区的其他子分区的，由散列函数决定的，被重新做过散列操作后的行。

在如下例子中，存储在表空间 usl 中的，新散列子分区 us_loc5i 达被增加到表 diving 中的范围分区 locations_us 中。

ALTER TABLE diving MODIFY PARTITON locations_us

ADD SUBPARTITION us_locs5 TABLESPACE us1;

必须重建对应于已增加的，和重新做过散列操作的子分区的，局部索引子分区。除非指定 UPDATE GLOBAL INDEXES，否则，所有全局索引，或分区全局索引的所有分区，将被标记成 UNUSABLE，并必须被重建。

# 增加索引分区

不能明确地给局部索引增加分区。替代方法是，只有当为基础表增加分区时，才能给局部索引增加了一个新分区。特别是，当表中定义了一个局部索引 ，并过布 ALTER TABLE 语句来增加分区时，也就给该局部索引增加了一个匹配的分区。因为 ORACLE 会给新的索引分区分配名字和默认物理存储属性，所以在完成了 ADD 操作之后，可能想要给它们重命名或更改它们。

因为最高分区总是具有 MAXVALUE 分区比较，所以不能给全局索引增加分区。如果要想增加一个新的最高分区，就要使用 LATER INDEX…SPLIT PARTITION 语句了。

# 合并分区

合并分区是一种减少散列分区表中的分区数量，或组合分区表中的子分区数量的一种方法。合并散列分区时，由散列函数决定将其内容重新划分到一个或多个剩余的分区中。由 ORACLE 选择要被合并的特殊的分区，并在重新划分其内容后将其删除。

按下表所解释的那样，索引可能被标记成 UNUSABLE：

| 表类型 | 索引特性 |
| --- | --- |
| 普通的（堆积的） | ■ 对应于被选中分区的任何局部索引分区也被删除。对应于一个或多个兼并的分区的局部索引分区被标记成 UNUSABLE，并必须被重建。<br>■ 除非指定 UPDATE GLOBAL INDEXES，否则，所有全局索引，或分区全局索引的所有分区，都被标记成 UNUSABLE，并必须被重建 |
| 索引结构的 | ■ 行上面所注，某些局部索引被标记成 UNUSABLE。<br>■ 所有全局索引仍然可用。 |

## 合并散列分区表中的分区

用 LATER TABLE …COALESCE PARTITION 语句，来合并散列分区表中的分区。如下语句靠合并一个分区来减少该表的一个分区。

```
ALTER TABLE ouu1
COALESCE PARTITION;
```

## 合并组合分区表中的分区

如下语句将分区 us_locationsg 的一个子分区的内容，划分到相同分区的一个或多个子分区中（由散列函数决定）。基本上说，该操作是 MODIFY PARTITION…ADD SUBPARTITION 子句的逆操作。

```
ALTER TABLE diving MODIFY PARTITION us_locations
COALESCE SUBPARTITION;
```

# 删除分区

可以删除范围，组合或列表分区表中的分区。对散列分区表，或组合分区表的散列子分区来说，必须用执行合并操作来替代删除操作。

## 删除表分区

使用 ALTER TABLE …DROP PARTITION 语句来删除表分区。如果想保留分区中的数据，则用 MERGE PARTITION 语句来代替 DROP PARTITION 语句。

如果给表定义过局部，这个语句还会从局部索引中删除匹配的分区或子分区，所有的全局索引，或分区全局索引的所有分区都将被标记成 UNUSABLE，除非满足如下条件之一：

- 指定了 UPDATE GLOBAL INDEXES（不能对索引结构表做这个指定）
- 要删除的分区或其子分区是空的

注意：不能删除表中唯一的分区，那么，就删除该表。

如下几节包含删除表分区的某些方案。

## 从包含数据和全局索引的表中删除分区

如果在分区中包含数据和在表上定义了一个或多个全局索引，就用下面的方法之一来删除该表的分区。

## 方法 1：

在发布 ALTER TABLE…DROP PARTITION 语句期间先不管全局索引。然后，必须重建任何全局索引（无论是否分区），因为索引（或索引分区）可能已被标记成了 UNUSABLE。下面的语句先从 sales 表中删除分区 dec98，然后重建其全局非分区索引。

```
ALTER TABLE sales DROP PARTITION dec98
ALTER INDEX sales_area_ix REBUILD;
```

假如索引 sales_area_ix 是一个范围分区全局索引，那么，就需要重建该索引的所有分区。进一步说，不可能在一条语句中重建一个索引的所有分区。必须为索引中的每个分区写一条单独的 REBUILD 语句。下面的语句重建索引分区 jan99_ix,feb99_ix,mar99_ix,…idh.99_ix。

```
ALTER    INDEX sales_area_ix REBUILD PARTITION jan99_ix;
ALTER    INDEX sales_area_ix REBUILD PARTITION feb99_ix;
ALTER    INDEX sales_area_ix REBUILD PARTITION mar99_ix;
    …
ALTER    INDEX sales_area_ix REBUILD PARTITION nov99_ix;
```

这种方法最适合要删除的分区中包含该表中很多数据的大表。

## 方法 2：

在发布 ALTER TABLE…DROP PARTITION 语句之前，先发布 DELETE 语句来删除分区中的所有行。DELETE 语句更改全局索引，并启动触发器和产生重做及撤消日志。

例如，要删除分区边界为 10000 的第一个分区，就发布如下语句：

```
DELETE FROM sales WHERE TRANSID <10000;
ALTER TABLE sales DROP PARTITION dec98;
```

这种方法最适合于小表，或者要删除的分区只包含该表中较少数据的大表。

## 方法 3：

在 ALTER TABLE 语句中指定 UPDATE GLOBAL INDEXES，这导致在删除分区的同时更改全局索引。

```
ALTER TABLE sales DROP PARTITION dec98
UPDATE GLOBAL INDEXES;
```

# 删除包含数据和参照完整小约束的分区

如果分区包含数据，并且该表具有参照完整性约束，那就选择下面的方法之一 来删除该表的分区。该表只有局部索引，所以不需要建任何索引。

## 方法 1：

停用完整性约束，发布 ALTER TABLE …DROP PARTITION 语句，然后再启用完整性约束：

```
ALTER TABLE sales
DISABLE CONSTRAINT dname_sales1;
ALTER TABLE sales DROP PARTITION dec98;
ALTER TABLE sales
ENALBE CONSTRAINT dname_sales1;
```

这种方法最适合于要删除的分区中包含该表中很多数据的大表。

## 方法 2：

在发布 ALTER TABLE …DROP PARTITION 语句之前，先发布 DELETE 语句来删除分区中的所有行。DELETE 语句参照完整性约束，并启动触发器 和产生重做及撤消日志。

```
DELETE FROM sales WHERE TANSI<10000;
ALTER TABLE sales DROP PARTITION dec94;
```

这种方法最适合于小表，或者要删除的分区只包含该表中较少数据的大表。

# 删除索引分区

可以明确地删除局部索引的分区。要么，只有当删除基础表的分区时才删除局部索引分区。

如果全局索引是空的，就可以发布 ALTER INDEX …DROP PARITITON 语句，来明确地删除它，但是，如果全局索引分区中包含数据，那么，删除该分区就将导致下一个最高分区被标记成 UNUSABLE，例如，如想删除索引分区 P1，并且 P2 是下一个最高分区，就必

须发布如下语句：

ALTER INDEX npr DROP PARTITION p1;

ALTER INDEX npr REBUILD PARTITION p2；

注意，不能删除全局索引中的最高分区。

# 交换分区

可以通过交换数据段的方法，将一个分区（或子分区）转换成一个非分区表，或将一个非分区表转换成一个分区表的分区（或子分区）。还可以将一个散列分区表转换成一个组合分区表的一个分区，或将组合分区表的分区指挥成一个散列分区表。

当有一个使用非分区表的应用，又想将该非分区表转换成一个分区表的分区时，交换表分区就非常有用了。例如，你可能已经有了打算将其融合到分区表中的分区视图了。

当与可传输表空间一起使用时，交换分区还有助于高速数据装载。

交换分区时会保留登记属性。可以选择指定是否也交换局部索引，是否为适当的映射而对行进行校验。除非指定 UPDATE GLOBAL INDEXES（不能用于索引结构表），否则，ORACLE 会将正在交换分区的表上的全局索引、或所有全局索引分区，标记成 UNUSABLE。

正在交换的表上的任何全局索引或全局索引分区都被标记成为 UNUSABLE。

## 交换范围、散列、列表分区

要想将范围、散列、列表分区表的分区转换成非分区表，或反之，就要使用 ALTER TABLE…EXCHANGE PARTITION 语句。下面是一个将一个分区转换成一个非分区表的例子。本例中，表 stocks 可以是范围、散列或列表分区表。

ALTER TABLE stocks

EXCHANGE PARTITION p3 WITH stock_table_3;

## 将散列分区表转换成组合分区

本例要将整个散列分区表，及其所有子分区，转换成组合分区表的范围分区及其所有散列子分区。如下例所示。

首先，创建一个散列分区表

CREATE TABLE t1 （i NUMBER, j NUMBER）

PARTITION BY HASH（i）

（PARTITION p1,PARTITION p2）;

填充该表，然后按下面所示创建组合分区表：

CREATE TABLE T2 (i NUMBER ,j NUMBER)

PARTITION BY RANGE(j)

SUBPARTITION BY HASH(i)

(PARTITION p1 VALUES LESS THAN(10)

```
SUBPARTITION t2_pls1
SUBPARTITION t2_pls2,
PARTITION p2 VALUES LESS THAN(20)
SUBPARTITION t2_p2s1
SUBPARTITION t2_p2s2));
```

重要的事是，表 t1 的分区键与表 t2 的子分区键相同。

为了将数据从 t1 迁移到 t2 并校验行，要使用如下语句：

```
ALTER TABLE t1 EXCHANGE PARTITION p1 WITH TABLE t2
WITH VALIDATION;
```

## 交换组合分区表的子分区

要想将组合分区表的散列分区转换成非分区表，或反之，就要使用 ALTER TABLE …EXCHANGE SUBPARTITION 语句。如下例子将表 sales 的子分区 q3_1999_s1 转换成非分区表 q3_1999。局部索引分区被转换成 q3_1999 上对应的索引。

```
ALTER TABLE sales EXCHANGE SUBPARTITIONS q3_1999_s1
WITH TABLE q3_1999 INCLUDING INDEXES;
```

# 融合分区

要想将两个分区的内容融合到一个个分区，就要使用 ALTER TABLE …MERGER PARTITION 语句。两个原始分区及其任何对应的局部索引都将被删除。

不能将该语句，用于散列分区表或组合分区表的散列子分区。

除非涉及到的分区或子分区是空的，否则，按下表所解释的那样，索引可能被标记成 UNUSABLE：

| 表类型 | 索引特性 |
|---|---|
| 普通的（堆积的） | ■ ORACLE 将所有结果所对应的局部索引分区或子分区都标记成 UNUSALBLE<br>■ 除非指定 UPDATE GLOBAL INDEXES，否则，所有全局索引，或分区全局索引的所有分区都被标记成 UNUSABLE，并必须被重建。 |
| 索引结构的 | ■ ORACLE 将所有结果所对应的局部索引分区或子分区都标记成 UNUSABLE。<br>■ 所有全局索引仍然可用。 |

## 融合范围分区

允许将两个相邻的范围分区中的内容融合到一个分区。不相邻的范围分区不能融合，结果分区继承两个被融合分区的较高的上边界。

融合范围分区的一个原因是希望联机地将历史数据保留在一个大分区中。例如，有日分区，以及将最早的日分区累积到月分区，等等。

如下脚本是融合范围分区的一个例子。

首先，创建一个分区表并创建局部索引。

```
--Create a table with four partitions each on its own tablespace
--Parttitioned by range on the data column
--
CREATE TABLE four_seasons
(
one DATE,
two VARCHAR2(60),
three NUMBER
)
PARTITION BY RANGE (one)
(
PARTITION quarter_one
VALUES LESS THAN ( TO_DATE ('01-apr-1998','dd-mon-yyyy'))
TABLESPACE quarter_one,
PARTITION quarter_two
VALUES LESS THAN ( TO_DATE ('01-jul-1998','dd-mon-yyyy'))
TABLESPACE quarter_two,
PARTITION quarter_three
VALUES LESS THAN ( TO_DATE ('01-oc-1998','dd-mon-yyyy'))
TABLESPACE quarter_three,
PARTITION quarter_four
VALUES LESS THAN ( TO_DATE ('01-jan-1999','dd-mon-yyyy))
TABLESPACE quarter_four,
)
/
--
--Create local PREFIXED index on Four_Seasons
--Prefiexed because the leftmost columns of the index match the partition key
--
CREATE INDEX I_four_seasons_1 ON four_seasons (one, two)
LOCAL (
PARTITION I_quarter_one TABLESPACE I_quarter_one,
PARTITION I_quarter_two TABLESPACE I_quarter_two,
PARTITION I_quarter_three TABLESPACE I_quarter_three,
PARTITION I_quarter_four TABLESPACE I_quarter_four
)
/
```

下一步，融合分区。

```
--Merge the first two partitions
--
```

```
ALTER TABLE four_seasons
MERGE PARTITIONS quarter_one,quarter_two INTO PARTITION quarter_two
/
```

然后，为受影响的分区重建局部索引。

```
--Rebuild index for quarter_two, which has been marked unusable
--because it has not had all of the data from Q1 added to it.
--Rebuilding the index will correct this.
--
ALTER TALBE four_seasons MODIFY PARTITION
Quarter_two REBUILD UNUSABLE LOCAL INDEXES
/
```

## 融合列表分区

融合列表分区时，要融合的分区可以是任何两个分区。因为列表分区方法并不假设分区的任何排序，所以这两个分区不需要像范围分区中那样是相邻的。结果分区由两个原始分区中的所有数据组成。

如下语句用列表分区方法，将一个列表分区表的两个分区融合成一个分区，该分区继承了它们两者在表层次上的所有默认属性，除了在语句中指定的 PCTFREE 和 MAXEXTENTS 之外。

```
ALTER TALBE sales_by_region
MERGER PARTITIONS sales_northwest,sales_southwest
INTO PARTITION sales_west
PCTFREE 50 STORAGE (MAXEXTENTS 20);
```

两个原始分区的值列表被指定为：

```
PARTITION sales_northwest VALUES (‘WA’,’OR’,’WY’,’MT’)
PARTITION sales_ southwest VALUES (‘AZ’,’NM’,’CO’)
```

而结果 sales_west 分区的值列表是由这两个分区的值列表的并集组成，即：

```
(‘WA’,’OR’,’WY’,’MT’, ‘AZ’,’NM’,’CO’)
```

## 融合范围组合分区

融合范围组合分区时，子分区被重做散列操作，形成或者在 SUBPARTITIOSNS 或 SUBPARTITION 子句中指定的子分区数量，或者（如没有包含这些子句时）就用表层次上默认的子分区数量。

要注意，分割范围组合分区与融合两个范围组合分区两者，对属性的继承是有区别的。分割分区时，因为仅有一个双亲，所以新的分区可以从原始分区中继承属性。然而，当融合分区时，因为有两个双亲，所以必须继承表层次上的默认属性，即，新分区不能继承其一而损害其二的利益。

如下例子融合两个范围组合分区：

```
ALTER TABLE all_seasons
```

MERGE PARTITIONS quarter_1,quarter_2 INTO PARTITION quarter_2
SUBPARTITION 8;

# 修改默认属性

可以修改表，或组合分区表的分区的默认属性。修改默认属性时，新属性只影响以后创建的分区，或子分区。创建新分区或子分区时还可以特意重载默认值。

修改表的默认属性

可以用 ALTER TABLE 语句的 MODIFY DEFAULT ATTRIBUTES 子句来修改会被范围分区、列表分区或散列分区继承的默认属性。如下例子修改表 emp 的 PCTFREE 属性的默认值，以便创建任何新分区。

ALTER TABLE emp
MODIFY DEFAULT ATTRIBUTES PCTFREE 25;

对散列分区表而言，只能修改 TABLESPACE 属性。

## 修改分区的默认属性

为了修改在创建子分区时要继承的默认属性，就使用 ALTER TABLE…MODIFY DEFAULT ATTRIBUTES FOR PARTITION 语句。如下语句修改 TABLESPACE 属性，该表空间将存储组合分区表 emp 中的分区 pl 的以后的子分区。

ATLER TABEL emp
MODIFY DEFAULT ATTRIBUTES FOR PARTITION p1 TABLESPACE ts1;

因为所有子分区必须共享除 TABLESPACE 之外的相同的属性，所以它是仅能被修改的属性。

## 修改索引分区的默认属性

与表分区相似，可以修改会被范围分区全局索引的分区、或范围分区表、散列分区表或组合分区表的局部索引分区继承的默认属性，可以用 ALTER INDEX …MODIFY DEFAULT ATTRIBUTES 语句来做此事。如果要修改被组合分区表的子分区继承的默认属性，就用 ATLER INDEX …MODIFY DEFAULT ATTRIBUTES FOR PARTITION 语句来做此事。

## 修改分区的真实属性

可以修改表或索引的现存分区的属性。

不能修改 TABLESPACE 属性。要将分区或子分区移动到新的表空间，就使用 ALTER TABLESPACE…MOVE PARTITION / SUBPARTITION 语句。

修改范围分区或列表分区的真实属性

使用 ALTER TABLE…MODIFY PARTITION 语句来修改范围分区的现存属性。可以修改段属性（除 TABLESPACE 之外），可以分配和回收盘区，将局部索引分区标记成 UNUSABLE，或重建被标记成 UNUSABLE 的局部索引。

如果组合分区表具有范围分区，就要注意：

- 如果分配或回收盘区，该动作将在指定分区的每个子分区上执行。
- 而且，修改任何其他属性都将导致修改那个分区的所有子分区的对应的那些属性。还会修改分区层次的默认属性。为了避免修改现存子分区的属性，就要使用 MODIFY DEFAULT ATTRIBUTES 语句的 FOR PARTITION 子句。

下面是修改分区的真实属性的某些例子。

本例修改表 sales 的范围分区 sales_q1 的 MAXEXTENTS 存储属性：

ALTER TABLE sales MODIFY PARTITION sales_Q1 STORAGE （MAXEXTENTS 10）

下例会将组合分区表 scubagear 的分区 ts1 中所有局部索引分区都标记成 UNUSABLE：

ALTER TABLE scubagear MPDIFY PARTITION ts1 UNUSABLE LOCAL INDEXES;

## 修改散列分区的真实属性

仍然使用 ALTER TABLE…MODIFY PARTITION 语句来修改散列分区的属性，然而，因为各个散列分区的物理属性都必须相同（除 TABLESPACE 之外），所以，可以做的事情被限制到：

- 分配新盘区
- 回收未被使用的盘区
- 将局部索引子分区标记成 UNUSABLE
- 重建被标记成 UNUSABLE 的子分区

如下例子重建所有与表 dept 的散列分区 P1 相关的，不可用的局部索引分区：

ALTER TABLE dept MODIFY PARTITION p1

REBUILD UNUSABLE LOCAL INDEXES;

## 修改子分区的真实属性

可以用 ALTER TABLE 语句的 MODIFY SUBPARTITION 子句，来执行前面针对散列分区的那些动作，只是现在是在特别的组合分区表的子分区层次上进行操作。例如：

ALTER TABLE emp MODIFY SUBPARTITION p3_s1

REBUILD UNUSABLE LOCAL INDEXES;

## 修改索引分区的真实属性

可用 ALTER INDEX 语句的 MODIFY PARTITION 子句，来修改索引分区或它的子分区

的真实属性。其规则非常像用于表分区的规则，但不像 ALTER TABLE 语句的 MODIFY PARTITION 子句，因为这里没有重建不可用索引分区的下级子句，但这里有合并索引分区或它的子分区的下级子句。此就是说，合并意味着融合那些可以被释放以便被重新使用的索引数据块。

还可以用 MODIFY SUBPARTITION 子句，来分配或回收用于局部索引的子分区的存储体，或将它标记 UNUSABLE。

# 修改列表分区：增加或删除值

列表分区方法允许你选择增加或删除定义的值列表中的文字值。

## 修改分区：增加值

为了扩展现存分区的值列表，使用 ALTER TABLE 语句的 MODIFY PARTITION…ADD VALUES 子句。要增加的文字值必须没有被包含在任何其他分区的值列表中。任何对应的局部索引分区的分区值列表也被对应地扩展，并且，任何全局索引，或全局索引分区或局部索引分区，仍然可用。

如下语句将新的州代码集合（’OK’,’KS’）增加到一个现存分区的列表中。

```
ALTER TABLE sales_by_region
MODIFY PARTITION region_east
ADD VALUES （’OK’,’KS’）;
```

## 修改分区：删除值

为了从现存分区的值列表中删除文字值，使用 ALTER TABLE 语句的 MODIFY PARTITION…DROP VALUES 子句。该语句总是随着校验一起执行，即，它检查是否有任何行存在于要被删除的值的集合所对应的分区中。如果发现了这种行，那么 ORACLE 就返回一条错误消息并且操作失败。需要时，在打算删除文字之前，先用 DELETE 语句从表中删除对应的行。

注意：不能删除描述分区的值列表中的所有的文字值。那么，就必须使用 ALTER TABLE…DROP PARTITION 语句。

任何对应的局部索引分区的值列表会反映该新的值列表，并且，任何全局索引，或全局索引分区或局部索引分区，仍然可用。

如下语句从一个现存分区的值列表中删除州代码集合（’OK’和’KS’）。

```
ALTER TABLE sales_by_region
MODIFY PARTITION region_east
DROP VALUES （’OK’,’KS’）
```

注意：因为为了检查对应于要被删除的文字值的分区中的行的存在性，需要执行查询，所以最好在该表上创建一个局部前缀索引。这会加速查询和所有操作的执行速度。

# 移动分区

使用 ALTER TABLE 语句的 MOVE PARTITION 子句来：

- 重新分簇数据并减少碎片
- 将一个分区移动到另一个表空间
- 修改创建时的属性

典型地，可以使用 ALTER TABLE / INDEX…MODIFY PARTITION 语句，只用一步，就可以修改一个分区 的物理存储属性。但是，某些物理属性，如 TABLESPACE，不能用 MODIFY PARTITION 语句来修改。这时，就要用 MOVE PARTITION 子句了。

除非要被移动的分区不包含任何数据，否则，根据下表，索引可能被标记成 UNUSABLE：

| 表类型 | 索引特性 |
| --- | --- |
| 普通的（堆积的）<br>索引结构的 | ➢ 每个局部索引中的匹配分区被标记成 UNUSABLE。发布 MOVE PARTITION 之后必须重建这些索引<br>➢ 除非指定 UPDATE GLOBAL INDEXES，否则，任何全局索引，或分区全局索引的所有分区被标记成 UNUSABLE。<br>在要被移动的分区上定义的任何局部索引或全局索引仍然可用，因为它们是基于逻辑行标识（ROWIDS）的主键。然而对这些行标识的推测信息就不正确了 |

## 移动表分区

使用 MOVE PARTITION 子句来移动分区。例如，为了将最活动的分区移动到它自己的磁盘上的表空间中（为了平衡 I / O）并且不登记该动作，就发布如下语句：

```
ALTER TABLE parts MOVE PARTITION depot2
TABLESPACE ts094 NOLOGGING;
```

该语句总是删除该分区的旧段并创建一个新段，甚至于在没有指定新的表空间时也这么做。

## 移动子分区

如下语句说明如何移动一个表的子分区中的数据。本例还指定了一个 PARALLEL 子句。

```
ALTER TABLE scuba_gear MOVE SUBPARTITION bcd_types
TABLESPACE tbs23 PARALLEL（DEGREE2）;
```

## 移动索引分区

对于普通表，ALTER TABLE…MOVE PARTITION 语句将全局索引的所有分区都标记成 UNUSALBE。可以用 ALTER INDEX …REBUILD PARTITION 语句来逐个重建每个分区，以便重建整个索引。可以同时进行这些重建工作。

还可以简单地删除索引并重建该索引。

# 重建索引分区

重建索引分区的某些原因如下：

- 为了恢复空间和改善性能
- 为了修复由于媒体故障而导致损坏的索引分区
- 为了在用 Import 或 SQL*Loader 装载完基础表分区之后重建局部索引分区
- 为了重建被标记成 UNUSABLE 的索引分区

# 重建全局索引分区

可以用两种方法来重建全局索引分区

① 发布 ALTER INDEX …REBUILD PARTITION 语句，重建每个分区（可以同时运行重建）。

② 删除整个全局索引，重建该全局索引。

注意：第二种方法更有效，因为只扫描一次表。

对于在具有全局索引的分区表上进行的大多数维护操作而言，可以选择通过在 DDL 语句中指定 UPDATE GLOBAL INDEXES 来避免重建全局索引的工作。

# 重建局部索引分区

如下所示，可以用 ALTER INDEX 或者 ALTER TABLE 语句来重建局部索引：

- ALTER INDEX…REBUILD PARTITION / SUBPARTITION，该语句无条件地重建索引分区或子分区
- ALTER TABLE…MODIFY PARTITION / SUBPARTITION…REBUILD UNUSABLE LOCAL INDEXES 该语句在给定表分区或子分区中寻找所有不可用的索引并重建这些索引。该语句只重建被标记成 UNUSABLE 的索引分区。

# 用更改索引来重建分区

ALTER INDEX…REBUILD PARTITION 语句重建一个索引的一个分区。它不能用于组合分区表。重建索引时，还可以选择将该分区移动到一个新的表空间中或修改属性。

对于组合分区表，使用 ALTER INDEX …REBUILD SUBPARTITION 语句来重建一个索引的一个子分区。可以将该子分区移动到另一个表空间中或指定一个并行子句。如下语句重建一个表上的一个局部索引的一个子分区，并将该索引单子分区移到另一个表空间中。

```
ALTER INDEX scuba
REBUILD SUBPARTITION bcd_types
TABLESPACE tbs23 PARALLEL（DEGREE 2）;
```

## 用更改表来重建索引分区

ALTER TABLE…MODIFY PARTITION 语句的 REBUILD UNUSABLE LOCAL INDEXES 子句不允许革被重建的索引分区指定任何新的属性。如下例子在表 scubagear 的分区 p1 中查找和重建任何不可用的局部索引分区。

ALTER TABLE scubagear

MODIFY PARTITION p1 REBUILD UNUSABLE LOCAL INDEXES;

为了重建不可用的局部索引子分区，可以使用对应的 ALTER TABLE…MODIFY SUBPARTITION 子句。

# 重命名分区

可以重命名表和索引两者的分区和子分区的名字。给分区重命名的一个原因是为了起一个有意义的名字，用以替换在另外的维护操作中赋给该分区的默认的系统名字。

## 重命名表分区

用 ALTER TABLE…RENAME PARTITION 语句来重命名范围分区、散列分区、或列表分区的名字。例如：

ALTER TABLE scubagear RENAME PARTITION sys_p636 TO tanks;

## 重命名表子分区

相似的，可以给表的子分区起一个新名字。这时，使用

ALTER TABLE…RENAME SUBPARTITION 语法。

## 重命名索引分区

可用用相似的方法重命名索引分区和索引子分区的名字，但使用 ALTER INDEX 语法。

## 重命名一个索引分区

用 ALTER INDEX…RENAME PARTITION 语句来更改一个索引分区的名字。

## 重命名一个索引子分区

如下语句简单地表示了如何更改一个子分区的名字，该子分区具有一个系统产生的名

字，它是在基础表上增加一个分区时产生的：

ALTER INDEX scuba RENAME SUBPARTITION sys_subp3254 TO bcd_types;

# 分割分区

ALTER TABLE 或 ALTER INDEX 语句的 SPLIT PARTITION 子句被用于将一分区中的内容重新划分成两个新的分区。当一个分区变得太大以至于要用很长时间才能完成备份、恢复、或维护操作时，就应考虑分割分区的工作。还可以用 SPLIT PARTITION 子句来重新划分 I / O 负载。

这个语句不能用于散列分区或子分区。

除非要分割的分区不包含任何内容，否则，根据下表，索引可能被标记成 UNUSABLES：

| 表类型 | 索引特性 |
| --- | --- |
| 普通的（堆积的）索引结构的 | ➢ ORACLE 将每个局部索引中的新分区（有两个）标记成 UNUSABLE。<br>➢ 除非指定 UPDATE GLOBAL INDEXES，否则，任何全局索引，或分区全局索引的所有分区被标记成 UNUSABLE。并必须重建。<br>➢ ORACLE 将每个局部索引中的新分区（有两个）标记成 UNUSABLE。<br>➢ 所有全局索引仍然可用。 |

## 分割范围分区表的分区

用 ALTER TABLE …SPLIT PARTITION 语句来分割范围分区。可选地，可以给分割产生的两个分区指定新的属性。如果在该表上定义了局部索引，这条语句也分割在每个局部索引中匹配的分区。

在如下例子中，fee_katy 是表 vet_cats 的一个分区，它有一个局部索引 jaf1。在该上还有一个全局索引 vet。Vet 包含两个分区，vet_parta 和 vet_partb。

为了分割分区 fee_katy，并重建索引分区，发布如下语句：

ALTER TABLE vet_cates SPLIT PARTITION

fee_katy at（100） INTO（ PARTITION

fee_katy1…,PARTITION fee_katy2…）;

ALTER INDEX JAF1 REBUILD PARTITION Fee_katy1;

ALTER INDEX JAF1 REBUILD PARTITION Fee_katy2;

ALTER INDEX VET REBUILD PARTITION vet_parta;

ALTER INDEX VET REBUILD PARTITION vet_partb;

注意：如果没有指定新分区名字，则 ORACLE 赋予的名字的形式为 SYS_pn。可检查数据字典来找到赋给该新局部索引分区的名字，也许想给它们更名。没有指定的任何属性都从原始分区中继承。

## 分割列表分区表的分区

用 ALTER TABLE…SPLIT PARTITION 语句来分割列表分区。SPLIT PARTITION 子句允许你为那些具有对应的分区键值，被插入到第一个新分区中的行，指定一个文字值的值列表。原始分区中剩余的修补 插入到第二个分区中。

可以选择为分割所产生的两个分区指定新的属性。

如下语句将分区 regiion_east 分割成 2 个分区：

```
ALTER TABLE sales_by_region
SPLIT PARTITION region_east VALUES（’CT’,’VA’,’MD’）
INTO
（PARTITION region_east_1
PCTFREE 25 TABLESPACE tbs2,
PARTITION region_east_2
STORAGE （NEXT 2M PCTINCREASE 25））
PARALLEL 5;
```

原始 region_east 分区的文字值列表被指定为：

```
PARTITION region_east VALUES（’CT’,’VA’,’MD’,’NY’,’NH’,’ME’,’VA’,’PA’,’NJ’）
```

两个新分区的是：

- region_east_1 的文字值列是（’CT’,’VA’,’MD’）
- region_east_2 继承剩余的文字值列是，是（’NY’,’NH’,’ME’,’VA’,’PA’,’NJ’）各个分区具有在分区层次上指定的新的物理属性，该操作用程度 5 的并行机制执行。

## 分割范围组合分区

这与融合范围组合分区相反。分割范围组合分区时，新的子分区被重做散列操作，形成在 SUBPARTITIONS 子句或者在 SUBPARTITION 子句中指定的子分区的数量。或者，如果没有包含这些子句，新的分区就继承而被分割的分区中的子分区（和表空间）的数量。

注意，分割范围组合分区和融合两个范围组合分区，在属性的继承上是有区别的。分割分区时，因为只有一个双亲，所以新的分区可以继承原始分区的属性。然而，融合分区时，属性必须继承表层次的默认值，因为此时有两个双亲，新的分区不能继承其一而损害其而的利益。

如下例子分割一个范围组合分区：

```
ALTER TABLE all_seasons SPLIT PARTITION quarter_1
AT（TO_DATE （’16-dec-1997’,’dd-mon-yyyy’））
INTO（PARTITION q1_1997_1 SUBPARTITIONS 4 STORE IN（ts1,ts3）,
PARTITION q1_1997_2）;
```

## 分割索引分区

不能明确地分割局部索引中的分区。只有当分割基础表的分区时，才能分割局部索引分

区。然而，可以像下面一样分割全局索引分区：

ALTER INDEX quon1 SPLIT

PARTITION Canada AT VALUES LESS THAN（100）INTO

（PARTITION canada1…, PARTITION canada2…）

ALTER INDEX quon1 REBUILD PARTITION canada1；

ALTER INDEX quon1 REBUILD PARTITION canada2；

正被分割的索引可以包含索引数据，并且不需要重建所产生的分区，除非原始分区事先已被标记成了 UNUSABLE。

# 舍弃分区

用 ALTER TABLE…TRUNCATE PARTITION 语句，来移出表分区中的所有的行。舍弃分区有些像删除分区，区别在于这是清空分区中的数据而不是物理地将其删除以外。

不能舍弃索引分区。但是，如果为该表定义了局部索引，则 ALTER TABLE TRUNCATE PARTITION 语句也舍弃在每个局部索引中匹配的分区。除非指定 UPDATE GLOBAL INDEXES（不能用于索引结构表），否则，全局索引，或分区全局索引的所有分区，都被标记成 UNUSABLE，并必须被重建。

## 舍弃一个表分区

用 ALTER TABLE…TRUNCATE PARTITION 语句，来移出表空间中的所有的行，伴随或不伴随回收空间。

舍弃包含数据和全局索引的表分区

如果分区包含数据和全局索引，就用如下方法之一来舍弃表分区。

方法 1：

在执行 ALTER TABLE TRUNCATE PARTITION 语句期间，不管全局索引。本例中，表 sales 有一个全局索引 sales_area_ix，该索引被重建。

ALTER TABLE sales TRUNCATE PARTITION dec98;

ALTER INDEX sales_area_ix REBUILD;

这个方法最适用于要舍弃的分区包含了表中大部分数据的大表。

方法 2：

在发布 ALTER TABLE…TRUNCATE PARTITION 语句之前，先发布 DELETE 语句，删除分区中的所有的行。DELETE 语句更改全局索引，并启动触发器和产生重做与撤消日志。

例如，为舍弃分区边界为 1000 的第一个分区，发布如下语句：

DELETE FROM sales WHERE TRANSID < 10000;

ALTER TABLE sales TRUNCATE PARTITION dec98;

这个方法最适用于小表，或要舍弃的分区包含了表中大数据的大表。

方法 3：

在 ALTER TABLE 语句中指定 UPDATE GLOBAL INDEXES。这使得在舍弃分区的同时也舍弃了全局索引。

ALTER TABLE sales TRUNCATE PARTITION dec98 UPDATE GLOBAL INDEXES;

舍弃包含数据和参照完整性约束的分区

如果分区包含数据和具有参照完整性约束，就用如下方法之一来舍弃该表分区。

方法 1：

停用完整性约束，发布 ALTER TABLE…TRUNCATE PARTITION 语句，然后，重新起用完整性约束：

ALTER TABLE sales

DISABLE CONSTRAINT dname_sales1;

ALTER TABLE sales TRUNCATE PARTITION dec94;

ALTER TABLE sales

ENABLE CONSTRAINT dname_sales1;

这种方法最适用于要舍弃的分区包含了表中很多数据的大表。

方法 2：

在发被 ALTER TABLE …TRUNCATE PARTITION 语句之前，先发布 DELETE 语句删除分区中的所有的行。DELETE 语句实施参照完整性约束，并启动触发器和产生重做与撤消日志。

注意：可以通过在删除分区中的所有行之前，给分区设置 NOLOGGING 属性（用 ALTER TABLE …MODIFY PARTITION …NOLOGGING），来充分地减少登记的数量。

DELTE FROM sales WHERE TRANSID <10000;

ALTER TABLE sales TRUNCATE PARTITION dec94;

这种方法最适用于小表，或要舍弃的分区包含了表中很少数据的大表。

## 舍弃一个子分区

用 ALTER TABLE …TRUNCATE SUBPARTITION 语句，来移出组合分区表的子分区中所有的行，还舍弃对应的局部索引子分区。

如下语句显示如何舍弃表的子分区中的数据。本例中，被删除的行所占据的空间就可以被表空间中其他模式的对象使用了。

ALTER TABLE diving TRUNCATE SUBPARTITION us_locations DROP STORAGE;

# 分区表和索引的例子

## 移动历史表中的时间窗口

历史表描述一个企业在一段时间间隔之中的业务事务。历史表可以是基础表，它包含基础信息，如销售、支票和订单。历史表还可以是积累表，它是用诸如 GROUP BY 、AVERAGE 或 COUNT 等操作，从基础信息中提取的汇总信息。

历史表中的时间间隔常常是一个滚动窗口。DBA 周期性地删除描述最早的事务的行的集合然后为描述最近的事务的行的集合分配空间。例如，1995 年 4 月 31 日业务结束时，DBA

删除描述 1994 年 4 月以来的事务的行（及其支持的索引项），并为 1995 年 4 月的事务分配空间。

现在考虑一个特例，有一个表 order，它包含 13 个月的事务：1 年的历史数据和本月的订单，每个月有一个分区。这些按用的分区被命名成 order_yymm,t 亦即它们所在的表空间。

Order 表含有两个局部索引，order_ix_onums 一个订单号码上的局部的、前缀的、唯一索引，order_iex_supps 一个供应商号码上的局部的、非前缀索引。局部索引分区用匹配基础表的后缀命名。还有一个用于客户姓名的全局唯一索引 order_ix_cust。order_ix_cust 包含 3 个分区，每一个分区用于 1 / 3 的字母。1994 年 10 月 31 日，按如下方法修改时间间隔：

① 备份最早的时间间隔的数据。

ALTER TABLESPACE order_9310 BEGIN BACKUP;

…

ALTER TABLESPACE order_9310 END BACKUP;

② 删除最早的时间间隔的分区

ALTER TABLE order DROP PARTITION order_9310;

③ 增加最近的时间间隔的分区

ALTER TABLE order ADD PARTITION order_9411;

④ 重建全局索引分区

ALTER INDEX order_ix_cust REBUILD PARTITION order_ix_cust_AH;

ALTER INDEX order_ix_cust REBUILD PARTITION order_ix_cust_IP;

ALTER INDEX order_ix_cust REBUILD PARTITION order_ix_cust_QZ;

通常，ORACLE 获得足够的锁以保证没有什么操作（DML，DDL 或工具）会干扰单个的 DDL 语句，如 ALTER TABLE…DROP PARTITION。然而，如果分区维护操作需要几步，则 DQB 就要负责保证应用（或其他维护操作）不会干扰多步操作的进行。为此，某些针对性的方法是：

在一个被定义好的批处理窗口期间，关闭所有用户层次的应用。

通过撤消所有应用使用的角色的访问权限，来保证没有谁可以访问表 order。

## 将分区视图转换成分区表

本方案描述如何将一个分区视图（也称为“手动分区”）转换成一个分区表。分区视图按如下定义：

CREATE VIEW accounts AS

SELECT * FROM accounts_jan98

UNION ALL

SELECT * FROM accounts_feb98

UNION ALL

…

SELECT * FROM accounts_dec98

为了增量式地将分区视图迁移到一个分区表，采用如下步骤：

① 首先，通过创建分区表，仅仅两个最近的分区（account_nov98 和 accounts_dec98）会被从该视图迁移到该分区表中。每个分区获得两数据块的一个段（作为一个占位器）

CREATE TABLE accounts_new（…）

TABLESPACE ts_temp STORAGE（INITIAL 2）
PARTITION BY RANGE（opening_date）
（PARTITION jan98 VALUES LESS THAN（'01-FEB-1998'），
PARTITION dec98 VALUES LESS THAN（'01-JAN-1999'））；

② 用 EXCHANGE PARTITION 语句，将滚表迁移到对应的分区。

ALTER TABLE account_new
EXCHANGE PARTITION nov98 WITH TABLE
accounts_nov98 WITH VALIDATION;
ALTER TABLE accounts_new
EXCHANGE PARTITION dec98 WITH TABLE
Accounts_dec98 WITH VALIDATION;

这样，与 nov98 和 dec98 分区相关占位器数据段就同与 accounts_nov98 和 accounts_dec98 表相关的数据段作了交换。

③ 重新定义 accounts 视图

CREATE OR REPLACE VIEW accounts AS
SELECT * FROM accounts_jan98
UNION ALL
SELECT * FROM accounts_feb_98
UNION ALL
…
UNION ALL
SELECT * FROM accounts_new PARTITION（nov98）
UNION ALL
SELECT * FROM accounts_new PARTITION（dec98）

④ 删除 accounts_nov98 和 accounts_dec98b 表，它们拥有最初附属于 nov98 和 dec98 分区表的占位器数据段。

⑤ 在将 UNION ALL 视图中的所有的表都转换到分区中之后，删除该视图，并将该分区表更名成被删除视图的名字。

DROP VIEW accounts;
RENAME accounts_new TO accounts;

## 查看有关分区表和索引的信息

| 视图 | 说明 |
| --- | --- |
| DBA_PART_TABLES | DBA 视图显示数据库中所有分区表分区信息，ALL 视图显示可被用户访问的所有分区表的分区信息。USER 视图局限于显示用户所拥有分区表分区信息 |
| ALL_PART_TABLES | |
| USER_PART_TABLES | |
| DBA_TAB_PARTITIONS | 显示分区层次的分区信息、分区存储参数、和由 ANALYZW 语句决定的用于分区的分区统计数据。 |
| ALL_TAB_PARTITIONS | |
| USER_TAB_PARTITIONS | |
| DBA_TAB_SUBPARTITIONS | 显示子分区层次的分区信息、子分区存储参数、和由 ANALYZE 操作决定的用于子分区统计数据 |
| ALL_TAB_SUBPARTITIONS | |
| USER_TAB_SUBPARTITIONS | |

| | |
|---|---|
| DBA_PART_KEY_COLUMNS<br>ALL_PART_KEY_COLUMNS<br>USER_PART_KEY_COLUMNS | 显示分区表的分区键列 |
| DBA_SUBPART_KEY_COLUMNS<br>ALL_SUBPART_KEY_COLUMNS<br>USER_SUBPART_KEY_COLUMNS | 显示综合分区表（和组合分区表上的局部索引）的子分区键列 |
| DBA_PART_COL_STATISTICS<br>ALL_PART_COL_STATISTICS<br>USER_PART_COL_STATISTICS | 显示表的分区的，列统计数据和直放图信息 |
| DBA_SUBPART_COL_STATISTICS<br>ALL_SUBPART_COL_STATISTICS<br>USER_SUBPART_COL_STATISTICS | 显示表的子分区的，列统计数据和直放图信息 |
| DBA_PART_HISTOGRAMS<br>ALL_PART_HISTOGRAMS<br>USER_PART_HISTOGRAMS | 显示表的分区上的直方图的直方图数据（每个直方图的端点） |
| DBA_SUBPART_HISTOGRAMS<br>ALL_SUBPART_HISTOGRAMS<br>USER_SUBPART_HISTOGRAMS | 显示表的子分区上的直方图的直方图数据（每个直方图的端点） |
| DBA_PART_INDEXES<br>ALL_PART_INDEXES<br>USER_PART_INDEXES | 显示分区索引的分区信息 |
| DBA_IND_PARTITIONS<br>ALL_IND_PARTITIONS<br>USER_IND_PARTITIONS | 显示索引分区的如下信息：分区层次的分区信息、分区的存储参数、ANALYZE 语句采集的统计数据 |
| DBA_IND_SUBPARTITIONS<br>ALL_IND_SUBPARTITIONS<br>USER_IND_SUBPARTITIONS | 显示索引子分区的如下信息：分区层次的分区信息、分区的存储参数、ANALYZE 语句采集的统计数据 |