管理表

单世民

概述

- 在Oracle系统中，表是数据库中的主要对象，是真正存储信息的对象。一般而言，表具有以下特征：
  - 代表实体
  - 表名在数据库中是**唯一**的
  - 由行和列组成
  - 行的顺序是任意的
  - 列的顺序是任意的
  - 列名在表中是**唯一**的

表属性的特点

- 对于所有类型的表，Oracle都允许开发人员和管理员规定各种表属性，他们会确定如下内容：
  - 哪一个表空间包含表；
  - Oracle怎样将表物理存储在磁盘上；
  - 当从磁盘读取表数据时，Oracle怎样将它们与内存进行映射；
  - Oracle怎样控制表上特定操作的日志。

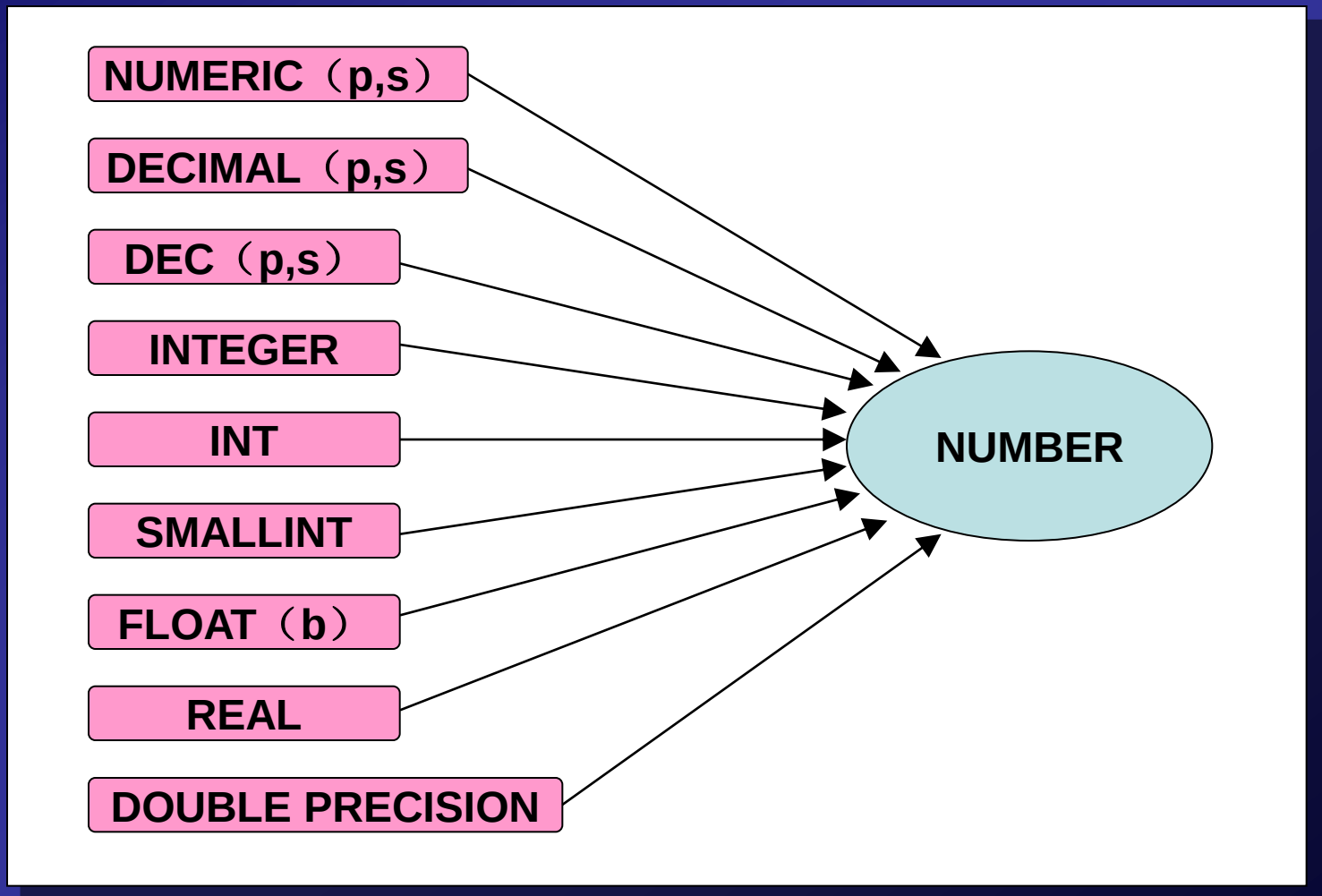
SQL数据类型

- 创建表的第一件事情是，学习Oracle中可以使用的SQL基本数据类型。前面已经提到过，SQL数据类型与PL/SQL数据类型有相同的地方，也有不同的地方。SQL数据类型有时也被称为数据库数据类型。

- Oracle系统支持工业标准的ANSI SQL数据类型。当在Oracle系统中使用ANSI SQL数据类型创建列时，系统会自动将这些数据类型隐式地转换成合适的Oracle内置数据类型。

SQL数据类型

- ANSI数据类型的转换



创建表

- 创建表的最简单、最常见的方法的语法格式如下：

```
CREATE TABLE [schema_name.]<table_name>(
<column_name> <data type>
              [default <expression>] [<constraint>]
[,<column_name> <data type>
              [default <expression>] [<constraint>]]
[,…]
)
ORGANIZATION INDEX
[TABLESPACE tablespace_name PCTTHRESHOLD n]
[INCLUDING column_name]
[OVERFLOW TABLESPACE tablespace_name]
[COMPRESS n];
```

约束

- 约束既可以在create table语句中创建，也可以在alter table语句中添加。在这两种定义方式中，定义约束的语法是类似的。

```
CREATE TABLE PriceHis
(
    PHID NUMBER(38) PRIMARY KEY,
    PHDate DATE NOT NULL,
    VID NUMBER(38) NOT NULL,
    PHNew NUMBER NOT NULL,
    CONSTRAINT fk_PriceHis_Video
    FOREIGN KEY (VID)
        REFERENCES Video(VID)
);
```

约束

- 约束的定义

```
ALTER TABLE Department
MODIFY (DAddr NOT NULL);
```

```
ALTER TABLE Dept_Video
ADD CONSTRAINT chk_DVType
CHECK (DVType IN (1,0));
```

如果用户没有提供约束名，系统将以SYS_Cn格式指定一个名字

约束

- 主键约束 PRIMARY KEY
- 外键约束 FOREIGN KEY
- 非空约束 NOT NULL
- 唯一性约束 UNIQUE
- 检查约束 CHECK

主键, 唯一键最多只能由32个列构成

约束的操作

- 启动和禁止现有约束

```
ALTER TABLE Dept_Video
ENABLE CONSTRAINT chk_DVType;
```

```
ALTER TABLE Dept_Video
DISABLE CONSTRAINT chk_DVType;
```

约束的操作

- 删除现有约束

```
ALTER TABLE Dept_Video
DROP CONSTRAINT chk_DVType;
```

创建表

- 在当前表的基础上创建新表

```
CREATE TABLE new_table_name AS(
        SELECT statement
        );
```

表类型

- 在Oracle系统中经常使用到的表包括堆表、外部表、索引组织表、临时表、分区表、簇表、散列簇表等类型。

表类型

- 堆表
  堆的含义是乱七八糟。在Oracle系统中，最基本的表类型是堆表。这里堆的含义是：数据是在磁盘上随机存储的。
- 一般情况下，Oracle系统在将数据行写入数据块时不会考虑其它行的存储位置。当向堆表中插入数据时，数据库会将该数据写入第一个有足够自由空间的段中。当修改和删除数据行时，系统将为新的插入提供可用的空间。

表类型

- 外部表
  外部表是指**在Oracle数据库之外的文件系统中存储的只读表**。

- 在Oracle 9i以前，使用操作系统上的普通文件存储数据的唯一方式就是通过SQL*Loader工具将其加载入数据库，或者使用insert语句把普通文件中的数据直接插入到数据库的表中，手工建立堆表。但是，这种加载方式的问题是，如果源数据改变了，那么加载过来的数据也必须手工维护，否则这些加载的数据就没有意义了。这种过程相当繁琐。

- 通过使用外部表，无须将数据复制到数据库中并且强制更新，可以让数据依然保留在普通文件中，并且允许数据库对其进行实地读取。在这种方式中，外部应用可以采用它认为合适的方法更新数据，而且不需要调用SQL*Loader工具执行数据加载操作。

表类型

- 外部表的创建
  - 创建目录对象

```
CREATE DIRECTORY ext_data_dir
AS 'D:\';
```

  - 创建外部表

```
CREATE TABLE table_ext(
  column_1 <data_type>,
  ...,
  column_n <data_type>
)
ORGANIZATION EXTERNAL
(
  TYPE oracle_loader
  DEFAULT DIRECTORY directory_name
  ACCESS PARAMETERS(FIELDS TERMINATED BY ',' )
  LOCATION('example.csv')
)
  REJECT LIMIT UNLIMITED;
```

表类型

- 索引组织表
  索引组织表(index organization table，IOT)可以存储索引，以便提高查询性能。
- 索引组织表是以牺牲插入性能和更新性能为代价而提供查询性能的方式。
- 假如，要使用《英汉大词典》查询一些词的含义，那么当用户在词典中搜索词时，用户就要将书打开到这个词的附近位置。用户可以根据词的字母顺序指导找到这个词，然后再基于用户打开书的位置进行前后搜索。
- 对于总是要通过特定索引访问的表（具有不频繁的更新或插入），使用索引组织表来代替堆组织表可以提高性能。

表类型

- 索引组织表的创建

```
CREATE TABLE table_IOT(
first varchar2(15),
second varchar2(15),
last varchar2(2000),
constraint pk_first primary key (first)
)
organization index
tablespace users pctthreshold 20
including second
overflow tablespace others;
```

表类型

- 索引组织表的创建
  - Overflow

    一般情况下，索引要表示表中的一个关键列(或者少量列)，不能在索引中存储类似于大规模的varchar2列或lob列。所以，B树索引将会采用小型的紧密聚集的数据块。然而，在索引组织表中，各个行的大小可能会非常大，将这样的数据存储成索引将会降低所获得的性能。

    overflow是为这种类型的表提供的机制。如果行数据变得太大，它可以将经常要查询的数据放在基本索引块中，而将不经常查询(或较大的数据列)存储在另外的段中，这种段称为**溢出段**。

表类型

- 索引组织表的创建
  - ¤ Including
    当使用including子句时，管理员可以规定表中的一个列。行中从第一列直到including子句所指定列（也包括这一列）的所有列都存储在索引块上，余下的列存储在溢出段中。

表类型

- 索引组织表的创建
  - pctthreshold
    行中的数据量超过块的PCTTHRESHOLD百分比时，行中余下的列将存储在溢出段中。
    当用户拥有变化长度的数据，没有办法判断各行大小时，pctthreshold就会很有用。通过将大量的数据存储在溢出段中，访问关键列的查询性能就会提高，这是因为关键列会存储在主表段中，这些查询可以避免访问更大的溢出段。

表类型

- 临时表（全局临时表）
  临时表是那些只在事务处理和会话期间存在数据的表。数据会在事务处理或会话开始之后插入临时表，当事务处理或会话完成之后该表就会被删除。通过采用这种方式，开发人员可以在需要执行的应用逻辑运行期间访问临时存储区域。

- 对于Oracle的临时表来说，就如同用户建立常规表一样，用户只需建立一次临时表，以后就可以在需要时使用这些临时表。相对于堆表来说，因为其他用户决不会使用这些记录，所以临时表不需要在它们包含的记录上维护锁定。与此同时，它们也不必维护过多的重做日志信息来防止数据库故障，用户不会希望从事务处理或会话的中间继续应用逻辑。

表类型

- 临时表
  为存储临时表数据分配的空间来自于临时表空间，而不是与永久性对象的数据存储争用表空间。

- 为提高应用性能，Oracle采用如下方式处理临时表：
  - 不为临时表创建重做日志
  - 知道在临时表上使用了第一个insert语句之后，才分配数据段
  - Truncate table命令只会为发出这个命令的会话在表中删除数据。使用这个表的其他会话的数据不会受到影响
  - 对于事务处理和会话范围，对待临时表上索引的方式与对待临时表的方式相同。因为没有两个会话或事务处理能够操作相同的行，所以在临时表上不要求dml锁定。

表类型

- 临时表的创建

```
CREATE global temporary TABLE temp(
first varchar2(15),
second varchar2(15),
last varchar2(2000)
)
on commit preserve rows;
```

on commit preserve rows:为特定的会话数据创建全局临时表

on commit delete rows:为特定的事务数据创建全局临时表

表类型

- **分区表**
  可以将非常大的表分割成比较小的分区，以便创建分区表。对于应用，分区表实际上就像一个表，但是由于它们工作在独立的分区上，而不是整个表，所以需要一些辅助的管理。
- **簇表**
  也称为簇，是物理上存储在一起的两个或多个表。因为这些表被认为总是会一起受到查询(使用SQL语句中的连接形式)，所以会将表存储在相同的数据块中，而不是他们各自的数据块中。由于所有需要的行都存储在公共的数据块上，所以这可以帮助减少查询中所需的磁盘读取量。
- **散列簇表**
  散列簇表与在磁盘上将两个或多个表实际存储在一起的簇表相似。两者之间的区别是Oracle用来存储和获取行的方法。簇表会使用分离索引中存储的键值获取行，而在散列簇中，Oracle会使用散列函数来判断存储所要获取的行的数据块的位置。

表的特性

- 表的创建参数

```
CREATE TABLE dept(
deptno number(2) not null,
dname varchar2(14 byte),
loc varchar2(13 byte),
constraint pk_dept_1 primary key (deptno)
)
tablespace users
storage(initial 64k next 0k minextents 1
maxextents 2147483645 pctincrease 0)
logging;
```

表的特性

- tablespace子句
  在创建表时，必须将表放置在某个表空间中。该表空间可以使用tablespace子句指定。tablespace子句是可选的，如果没有明确规定tablespace子句，那么所建立的表就会存放在建立表的用户帐户默认的表空间中。

表的特性

- storage子句
  在数据库中，有效管理空间消耗将会直接影响数据库增长和存储数据的能力。当用户在Oracle中创建表、索引等对象时，用户可以规定这些对象怎样消耗磁盘上的空间。这需要使用对象的storage子句来完成。如果没有使用storage子句创建表空间，那么存储属性就会获取默认值。当创建其他对象时，它们会继承表空间的存储属性。例如，如果表空间具有minextents 5属性，那么所创建的表也会获取minextents 5属性。

  当规定create tables命令中的storage子句时，用户可以使用这些参数：initial、next、pctincrease、minextents、maxextents等。

表的特性

- logging子句
  为了提高数据库的可靠性，记录数据库中所有数据的改变，可以在创建表时使用logging子句。这时表示对数据库的操作会产生重做日志。如果所发生的故障使数据不能从内存传递到数据库的数据文件中，那么就可以从重做日志中获取这些改变。这样，可以防止数据丢失，提高系统的可用性。

表的特性

- Logging子句
  当在create table语句中规定了nologging子句时，就认为这个表是非日志记录表。在这个表上进行的操作可能会导致数据库中很少的日志记录。

  但这并不意味着当使用nologging子句创建表时，在表上执行的操作就不会产生重做日志。除了以下所列出的活动之外，表中所发生的改变的重做日志活动正常：
  - Create table as select
  - SQL*Loader直接路径加载
  - 直接路径插入

唯一避免重做日志的方式就是使用全局临时表

表的特性

- Cache和nocache
  当在Oracle中执行全表搜索时，读入缓存区的数据块将会存储在最近最少使用列表的最近最少使用的一端。这就意味着，只要执行常规查询，就必须当从缓存区中读取数据时，就会将这些数据块换出缓存区。

  当创建表时，cache子句可以忽视这种行为。当在使用cache子句建立的表上执行全表搜索时，就会将数据块读入缓存区，并且放置到最近最常使用的一段。

  nocache是建立表时的默认值，如果明确指定cache子句时所建立的表就是nocache表，那么从nocache表中读取的数据块通常会被交换出SGA。

修改表

- 在Oracle数据库中创建表并不困难，但是通过一次尝试就创建一个完美的表几乎是不可能的。当表创建之后，在使用过程中，经常需要进行修改。
- 修改表可以使用alter table命令，常见的修改表的操作如下：
  - 在表中增加、修改或删除列
  - 重命名表
  - 将表移动到新的表空间
  - 改变表的存储属性
  - 改变表的某些特征

修改表

- 改变表结构-添加列

```
ALTER TABLE table_name
ADD new_column_name datatype
[NOT NULL];
```

修改表

- 改变表结构-改变列名

```
ALTER TABLE table_name
RENAME COLUMN old_column_name
              TO new_column_name;
```

修改表

- 改变表结构-改变列的数据类型

```
ALTER TABLE table_name
MODIFY column_name new_datatype;
```

修改表

- 改变表结构-删除列

```
ALTER TABLE table_name
DROP COLUMN column_name;
```

- 对于数据量比较大的表，用户还可以采用另外一种操作方式来避免对表进行整体重写，语法格式如下：

```
ALTER TABLE table_name
SET UNUSED COLUMN column_name;
```

修改表

- 更改表名

```
RENAME old_table_name TO new_table_name;
```

修改表

- 添加not null约束

```
ALTER TABLE Department
MODIFY (DAddr NOT NULL);
```

修改表

- 改变表的特性
  在创建表时，用户一般不知道在它们所支持的应用的生命期内施加给这些表的所有要求，尽可能地具有前瞻性地构建表，但是用户会不时地意识到通过改变表的属性可以获得更好的行，或消耗更少的资源。

```
ALTER TABLE table_name [CACHE|NOCACHE];
```

```
ALTER TABLE table_name [LOGGING|NOLOGGING];
```

```
ALTER TABLE table_name MOVE TABLESPACE tablespace_name
```

删除表

```
DROP TABLE table_name [cascade constraints];
```

- 通过指定cascade constraint，将会在删除表的同时，删除所有子表的外键约束

truncate table

- truncate table是用来删除表中所有数据、但是不删除表本身的DDL语句。同时，该表的索引也被删除。
  truncate table命令能够用于堆组织表、索引组织表、临时表等。
  Truncate table是DDL语句，不生成回滚数据，所以比使用delete命令更为有效。
- truncate table命令的语法格式如下：

```
TRUNCATE TABLE [schema.]table_name
[DROP STORAGE | REUSE STORAGE]
```

DROP/REUSE STORAGE用于控制是否重新利用已经为表分配的全部的空间

数据字典

- 每一个数据库都有一个数据字典。数据字典是用户的整个Oracle数据库的编目。当建立用户、表、约束或其他数据库对象时，Oracle都会自动维护一个在数据库中存储的项目编目。用户可以使用这个编目回答下面一些问题：
  - 我所拥有的所有表的名称和特性是什么？
  - 我有权限去查看的所有表的名称和属性是什么？
  - 如何检索我所创建的一些存储过程代码的源代码？
  - 向我展示为特定表空间存储数据的所有数据文件？
  - 罗列出那些依赖于其他对象的对象。
  - 罗列出特定表的所有列和列的属性。

小结

- 概述
- SQL数据类型
- 创建表和约束
- 表类型
- 表的特性
- 修改表
- 删除表
- truncate table
- 数据字典

The End