



Oracle

主动数据库机制的应用

陈芬 (深圳市福田区税务局 518038)

摘要: 通过 Oracle 提供的约束定义和触发器定义, 把应用数据库设计成主动数据库系统, 可以在数据库端由服务器监控数据库的状态和操作, 可以自动控制数据的完整性和安全性, 还可以自动生成跟踪、日志、审计等用户感兴趣的信息。

关键词: 主动数据库 完整性约束 触发器

1 主动数据库系统(active database system)

主动数据库系统是指由数据库服务器监控数据库状态和操作, 在状态改变时, 或在对数据库操作时, 根据不同的条件实时地触发响应, 这些响应可以自动控制数据库的状态, 可以根据权限和业务规则允许或禁止操作的进行, 可以根据状态的变化产生用户维护应用系统需要的信息等。这一系列过程遵循“事件-条件-动作”规则(event-condition-action rules)。这种规则在数据库中定义, 存储在数据库中, 与用户和应用无关, 可以被应用程序共享, 由服务器进行优化。

“事件-条件-动作”规则一般描述为:

```
on event
if condition
then action
```

这些规则也可以在客户端应用程序中定义, 事实上最初始的数据库系统也是这样做的。但应用程序开发时不能从整体上考虑到数据的一致性和完整性, 可能会造成疏忽; 不同开发人员考虑的角度不同, 反而会人为造成数据的不一致, 这也正是提出主动数据库系统的原因。因此, 应用数据库系统的一致性和完整性应由数据库设计人员在数据库设计时就充分考虑, 通过设计主动数据库系统, 避免上述问题的产生。

Oracle 数据库提供了引用完整性约束(integrity constraint)和触发器(trigger)两种实现主动数据库的机制。数据库设计人员通过定义表上的引用完整性和数据库级或表级的触发器, 可以实现主动的应用数据库系统。完整性约束实际上是由系统隐含定义了事件-条件-动作

规则, 而触发器必须由系统开发人员明确地定义其事件-条件-动作规则。

2 Oracle 引用完整性约束的应用

Oracle 在表一级定义了五种引用完整性约束, 分别为非空(Not Null)、唯一(Unique Key)、主键(Primary Key)、检查(Check)、外键(Foreign Key)。Oracle 隐含定义的事件-条件-动作分别为:

(1) 非空: 定义表中的某个列不能有空值

事件 -- 插入和修改;

条件 - 定义为非空的列上有空值;

动作 -- 不允许插入或修改, 并给出出错信息。

(2) 唯一: 定义表中的某个列或某些列的组合没有重复值

事件 -- 插入和修改;

条件 - 定义为唯一的列上有重复值;

动作 -- 不允许插入或修改, 并给出出错信息;

(3) 主键: 定义表中的某个列或某些列为表的主关键字, 不能为空且不允许有重复值

事件 -- 插入和修改;

条件 - 定义为主键的列上有空值或重复值;

动作 -- 不允许插入或修改, 并给出出错信息;

(4) 检查: 由用户定义某个列或某些列的取值范围

事件 -- 插入和修改;

条件 - 列上的取值超出定义的范围;

动作 -- 不允许插入或修改, 并给出出错信息;

(5) 外键: 定义某个列或某些列必须是其他表的主键

或唯一键；在父表操作和子表操作时有不同的响应规则

事件 - 父表删除；

条件 - 有子记录存在；

动作 - 如果定义了级连删除 (delete cascade)，则同时删除子记录，否则不允许删除，并给出出错信息；

事件 -- 子表插入或修改；

条件 - 无父记录存在；

动作 -- 不允许插入或修改，并给出出错信息。

Oracle 的约束可以在生成表 (create table) 时定义，也可以在表生成后，用 alter table add/drop constraint 来增加或删除约束。

约束的定义：

```
constraint constraint_name
{not null (column [,column...]) |
unique (column [,column...]) |
primary key (column [,column...]) |
foreign key (column [,column...])
reference [user.] table [(column [,column...]) |
check (check condition)};
```

这是本单位企业档案管理数据库设计中部分表定义的引用完整性约束，以及通过这些约束实现的对数据库完整性的主动控制。Company 为企业信息表，invester 为投资者信息表，invest 为投资比例表。Invest 中的 c_id、i_id 分别指向 company 中的 id 和 investor 中的 id (因篇幅原因，这里省去了其他列的定义)。

```
create table company (
    id number,
    cname varchar2(100), ...,
    constraint company_id_pk primary key (id), -- 主键
    constraint company_cname_nn not null (cname), -- 非空
    constraint company_cname_uk unique(cname)); -- 唯一
create table investor (
    id number,
    iname varchar2(100), ...,
    constraint investor_id_pk primary key (id), -- 主键
    constraint investor_iname_nn not null (iname), -- 非空
    constraint investor_iname_uk unique(iname)); -- 唯一
create table invest(
    c_id number,
    i_id number,
    ratio number,
```

```
constraint invest_pk primary key (c_id, I_id), -- 主键
constraint invest_c_id_fk foreign key (c_id) -- 外键
references company(id),
constraint invest_i_id_fk foreign key (i_id) -- 外键
references investor(id),
constraint invest_ratio_ck check (ratio >0 and ratio
<=1)); -- 检查
```

3 Oracle 中触发器的应用

触发器定义了当一些数据库相关事件发生时数据库应采取的动作。当修改表、创建对象、用户操作或数据库系统操作发生时，Oracle 自动地、隐含地引发 (fired) 相应的触发器，与用户和应用程序无关。触发器可用于增加描述性的引用完整性，加强复杂业务的规则；或者监控数据的变动，生成日志或跟踪文件，生成审计信息；也可以根据不同的条件 (如时间、用户) 限制对数据库的操作，提供另一种安全控制的手段。具体应用有：

- 自动生成列的值；
- 防止非法事务；
- 加强复杂的安全授权检查；
- 在分布式数据库中加强各结点间的引用完整性；
- 加强用完整性约束不能定义的复杂业务规则；
- 提供传输事务的跟踪日志；
- 提供复杂有效的审计信息；
- 保证表之间的同步复制；
- 生成表操作的统计信息；
- 当有增、删、改视图上的数据时，把修改写到视图相应的基表上；

· 可把有关数据库事件、用户事件、SQL 语句的信息写入应用程序中。

引发触发器的事件包括：数据库实例的启动和关闭；一个特定的错误信息或任何错误信息；用户的登录和退出；表或视图上的 DML (Database Manipulation Language 数据库操作语言) 语句；DDL (Database Difinition Language 数据库定义语言) 语句。根据触发器触发事件的级别可分为数据库级触发器和表级触发器，表级触发器又可分为行级触发器和语句级触发器，行级触发器对处理中的每一行都执行一次，语句级触发器对每个处理执行一次，可同时处理多行数据。

在 Oracle 中定义数据库级触发器的语法为：

```
create [or replace] trigger trigger
```

```
event
[when (condition)]
```

```
action;
```

例如, 在 SYS 用户或由操作系统认证的用户(用户名为 OPS 开头的用户) 登录时生成日志信息:

```
create or replace trigger log_logon
after logon database
when (user='SYS' or user like 'OPS%')
begin
insert into sys.event_log
values ('Logon' || USER || ' at '
|| to_char(sysdate, 'YYYY-MM-DD HH24 :
MI :SS'));
commit;
end;
```

定义表级触发器的语法为:

```
create [or replace] trigger [user.] trigger
{before | after | instead of}
{delete | insert | update [ of column [,column]...]}
[or { delete | insert | update [ of column [,
column]...]}]...
on [user.] {table | view}
[[referencing { old [as] old
| new [as] new}]]...
for each { row | statement }
[when (condition)]
pl/sql_block
```

例如, 为一些非常重要的表格(假设为 important)数

据模拟一个回收站:

```
create or replace trigger recycle
before delete on (important)
for each row -- 这是一个行级触发器
begin -- 事件无条件触发响应
insert into recycled values (... , user, sysdate);
end;
```

4 结束语

Oracle的自动数据库机制提供了完整性、安全性、统计审计等功能。系统开发人员在数据库设计阶段充分考虑整个数据库的完整性, 把完整性的控制放在数据库端, 并由系统通过定义的完整性约束和触发器来自动维护, 而与今后的用户、开发人员、应用程序无关, 不但可以减少编程的代码, 还可以避免不同开发人员造成的完整性控制的不一致性。用 Oracle 自动数据库同样可在用户或时间上对数据库的安全进行监控, 但必须说明由trigger提供的安全性只是一种辅助考虑, 应以 Oracle 的权限控制及其他安全措施为主要手段。Oracle主动数据库同样可以为用户和应用程序生成日志、审计的应用信息, 使用户的应用程序功能更齐全, 也更方便维护人员对数据库的监控和维护。■

参考文献

- 1 Active Database Systems Umeshar Dayal, Eric N.Hanson, Jennifer Widom.
- 2 Oracle Architecture and Adimnistration Oracle University Press.
- 3 Oracle 完全参考手册 机械工业出版社.