第9章 触发器

9.1 触发器的种类和触发事件

触发器必须由事件才能触发。触发器的触发事件分可为3类，分别是DML事件、DDL事件和数据库事件。

每类事件包含若干个事件，如表9-1所示。数据库的事件是具体的，在创建触发器时要指明触发的事件。

表9-1 触发器事件

种 类	关 键 字	含 义
DML 事件(3 种)	INSERT	在表或视图中插入数据时触发
	UPDATE	修改表或视图中的数据时触发
	DELETE	在删除表或视图中的数据时触发
DDL 事件(3 种)	CREATE	在创建新对象时触发
	ALTER	修改数据库或数据库对象时触发
	DROP	删除对象时触发
数据库事件(5 种)	STARTUP	数据打开时触发
	SHUTDOWN	在使用 NORMAL 或 IMMEDIATE 选项关闭数据库时触
	LOGON	当用户连接到数据库并建立会话时触发
	LOGOFF	当一个会话从数据库中断开时触发
	SERVERERROR	发生服务器错误时触发

触发器的类型可划分为4种：数据操纵语言(DML)触发器、替代(INSTEAD OF)触发器、数据定义语言(DDL)触发器和数据库事件触发器。

各类触发器的作用如表9-2所示。

表9-2 触发器

种　类	简　称	作　　用
数据操纵语言触发器	DML 触发器	创建在表上，由 DML 事件引发的触发器
替代触发器	INSTEAD OF 触发器	创建在视图上，用来替换对视图进行的插入、删除和修改操作
数据定义语言触发器	DDL 触发器	定义在模式上，触发事件是数据库对象的创建和修改
数据库事件触发器	—	定义在整个数据库或模式上，触发事件是数据库事件

9.2 DML触发器

9.2.1 DML触发器的要点

DML触发器是定义在表上的触发器，由DML事件引发。编写DML触发器的要素是：

* 确定触发的表，即在其上定义触发器的表。

* 确定触发的事件，DML触发器的触发事件有INSERT、UPDATE和DELETE三种，说明见表9-1。

* 确定触发时间。触发的时间有BEFORE和AFTER两种，分别表示触发动作发生在DML语句执行之前和语句执行之后。

* 确定触发级别，有语句级触发器和行级触发器两种。语句级触发器表示SQL语句只触发一次触发器，行级触发器表示SQL语句影响的每一行都要触发一次。

由于在同一个表上可以定义多个DML触发器，因此触发器本身和引发触发器的SQL语句在执行的顺序上有先后的关系。它们的顺序是：

* 如果存在语句级BEFORE触发器，则先执行一次语句级BEFORE触发器。

* 在SQL语句的执行过程中，如果存在行级BEFORE触发器，则SQL语句在对每一行操作之前，都要先执行一次行级BEFORE触发器，然后才对行进行操作。如果存在行级AFTER触发器，则SQL语句在对每一行操作之后，都要再执行一次行级AFTER触发器。

* 如果存在语句级AFTER触发器，则在SQL语句执行完毕后，要最后执行一次语句级AFTER触发器。

DML触发器还有一些具体的问题，说明如下：

* 如果有多个触发器被定义成为相同时间、相同事件触发，且最后定义的触发器是有效的，则最后定义的触发器被触发，其他触发器不执行。

* 一个触发器可由多个不同的DML操作触发。在触发器中，可用INSERTING、DELETING、UPDATING谓词来区别不同的DML操作。这些谓词可以在IF分支条件语句中作为判断条件来使用。

* 在行级触发器中，用:new 和:old(称为伪记录)来访问数据变更前后的值。但要注意，INSERT语句插入一条新记录，所以没有:old记录，而DELETE语句删除掉一条已经存在的记录，所以没有:new记录。UPDATE语句既有:old记录，也有:new记录，分别代表修改前后的记录。引用具体的某一列的值的方法是：

:old.字段名或:new.字段名

* 触发器体内禁止使用COMMIT、ROLLBACK、SAVEPOINT语句，也禁止直接或间接地调用含有上述语句的存储过程。

定义一个触发器时要考虑上述多种情况，并根据具体的需要来决定触发器的种类。

9.2.2 DML触发器的创建

创建DML触发器需要CREATE TRIGGER系统权限。创建DML触发器的语法如下：

CREATE [OR REPLACE] TRIGGER 触发器名

{BEFORE|AFTER|INSTEAD OF} 触发事件1 [OR 触发事件2...]

ON 表名

WHEN 触发条件

[FOR EACH ROW]

DECLARE

声明部分

BEGIN

主体部分

END;

其中：

OR REPLACE：表示如果存在同名触发器，则覆盖原有同名触发器。

BEFORE、AFTER和INSTEAD OF：说明触发器的类型。

WHEN 触发条件：表示当该条件满足时，触发器才能执行。

触发事件：指INSERT、DELETE或UPDATE事件，事件可以并行出现，中间用OR连接。

对于UPDATE事件，还可以用以下形式表示对某些列的修改会引起触发器的动作：

UPDATE OF 列名1，列名2...

ON 表名：表示为哪一个表创建触发器。

FOR EACH ROW：表示触发器为行级触发器，省略则为语句级触发器。

触发器的创建者或具有DROP ANY TIRGGER系统权限的人才能删除触发器。删除触发器的语法如下：

DROP TIRGGER 触发器名

可以通过命令设置触发器的可用状态，使其暂时关闭或重新打开，即当触发器暂时不用时，可以将其置成无效状态，在使用时重新打开。该命令语法如下：

ALTER TRIGGER 触发器名 {DISABLE|ENABLE}

其中，DISABLE表示使触发器失效，ENABLE表示使触发器生效。

同存储过程类似，触发器可以用SHOW ERRORS检查编译错误。

9.2.3 行级触发器的应用

在行级触发器中，SQL语句影响的每一行都会触发一次触发器，所以行级触发器往往用在对表的每一行的操作进行控制的场合。若在触发器定义中出现FOR EACH ROW子句，则为行级触发器。

【训练1】 创建包含插入、删除、修改多种触发事件的触发器DML_LOG，对EMP表的操作进行记录。用INSERTING、DELETING、UPDATING谓词来区别不同的DML操作。

在创建触发器之前，需要先创建事件记录表LOGS，该表用来对操作进行记录。该表的字段含义解释如下：

LOG_ID：操作记录的编号，数值型，它是该表的主键，由序列自动生成。

LOG_TABLE：进行操作的表名，字符型，非空，该表设计成可以由多个触发器共享使用。比如我们可以为dept表创建类似的触发器，同样将操作记录到该表。

LOG_DML：操作的动作，即INSERT、DELETE或UPDATE三种之一。

LOG_KEY_ID：操作时表的主键值，数值型。之所以记录表的主键，是因为主键是表的记录的惟一标识，可以识别是对哪一条记录进行了操作。对于emp表，主键是empno。

LOG_DATE：操作的日期，日期型，取当前的系统时间。

LOG_USER：操作者，字符型，取当时的操作者账户名。比如登录SCOTT账户进行操作，在该字段中，记录账户名为SCOTT。

步骤1：在SQL*Plus中登录STUDENT账户，创建如下的记录表LOGS：

```
CREATE TABLE logs(
LOG_ID NUMBER(10) PRIMARY KEY,
LOG_TABLE VARCHAR2(10) NOT NULL,
LOG_DML VARCHAR2(10),
LOG_KEY_ID NUMBER(10),
LOG_DATE DATE,
LOG_USER VARCHAR2(15)
);
```

执行结果：

表已创建。

步骤2：创建一个LOGS表的主键序列LOGS_ID_SEQ：

```
CREATE SEQUENCE logs_id_squ INCREMENT BY 1
START WITH 1 MAXVALUE 9999999 NOCYCLE
NOCACHE;
```

执行结果：

序列已创建。

步骤3：创建和编译以下触发器：

```
CREATE OR REPLACE TRIGGER DML_LOG
BEFORE --触发时间为操作前
DELETE OR INSERT OR UPDATE -- 由三种事件触发
ON emp
FOR EACH ROW -- 行级触发器
BEGIN
```

```
    IF INSERTING THEN
    INSERT INTO logs
    VALUES(logs_id_squ.NEXTVAL,'EMP','INSERT',:
new.empno,SYSDATE,USER);
     ELSIF DELETING THEN
     INSERT INTO logs
    VALUES(logs_id_squ.NEXTVAL,'EMP','DELETE',:
old.empno,SYSDATE,USER);
     ELSE
            INSERT INTO logs
    VALUES(logs_id_squ.NEXTVAL,'EMP','UPDATE',:
new.empno,SYSDATE,USER);
    END IF;
    END;
```

执行结果：

触发器已创建

步骤4：在EMP表中插入记录：

```
INSERT INTO emp(empno,ename,job,sal)
VALUES(8001,'MARY','CLERK',1000);
COMMIT;
```

执行结果：

已创建1行。

提交完成。

步骤5：检查LOGS表中记录的信息：

```
SELECT * FROM LOGS;
```

执行结果为：

```
LOG_ID       LOG_TABLE                LOG_DML
LOG_KEY_ID LOG_DATE   LOG_USER
---------- ---------- ---------- ----------
---------- ---------- ----------
         1 EMP        INSERT           8001      29-3月 -04
STUDENT
```

已选择 1 行。

说明：本例中在emp表上创建了一个由INSERT或DELETE或UPDATE事件触发的行级触发器，触发器的名称是LOG_EMP。对于不同的操作，记录的内容不同。本例中只插入了一条记录，如果用一条不带WHERE条件的UPDATE语句来修改所有雇员的工资，则将逐行触发触发器。

INSERT、DELETE和UPDATE都能引发触发器动作，在分支语句中使用INSERTING、DELETING和UPDATING来区别是由哪种操作引发的触发器动作。

在本例的插入动作中，LOG_ID字段由序列LOG_ID_SQU自动填充为1；LOGS表LOG_KEY_ID字段记录的是新插入记录的主键8001；LOD_DML字段记录的是插入动作INSERT；LOG_TABLE字段记录当前表名EMP；LOG_DATE字段记录插入的时间04年3月1日；LOG_USER字段记录插入者STUDENT。

【练习1】修改、删除刚刚插入的雇员记录，提交后检查LOGS表的结果。

【练习2】为DEPT表创建同样的触发器，使用LOGS表进行记录，并检验结果。

【训练2】 创建一个行级触发器LOG_SAL，记录对职务为CLERK的雇员工资的修改，且当修改幅度超过200时才进行记录。用WHEN条件限定触发器。

在创建触发器之前，需要先创建事件记录表LOGERR，该表用来对操作进行记录。该表的字段含义解释如下：

NUM：数值型，用于记录序号。

MESSAGE：字符型，用于记录错误信息。

步骤1：在SQL*Plus中登录STUDENT账户，创建如下的记录表LOGERR：

```
CREATE TABLE logerr(
NUM NUMBER(10) NOT NULL,
MESSAGE VARCHAR2(50) NOT NULL
);
```

执行结果：

表已创建。

步骤2：创建和编译以下触发器：

```
CREATE OR REPLACE TRIGGER log_sal
BEFORE
UPDATE OF sal
ON emp
FOR EACH ROW
WHEN (new.job='CLERK' AND (ABS(new.sal-old.sal)>200))
```

```
DECLARE
v_no NUMBER;
BEGIN
SELECT COUNT(*) INTO v_no FROM logerr;
INSERT INTO logerr VALUES(v_no+1,'雇员'||:new.ename||'的原工资：'||:old.sal||'新工资：'||:new.sal);
END;
```

执行结果：

触发器已创建。

步骤3：在EMP表中更新记录：

```
UPDATE emp SET sal=sal+550 WHERE empno=7788;
UPDATE emp SET sal=sal+500 WHERE empno=7369;
UPDATE emp SET sal=sal+50 WHERE empno=7876;
COMMIT;
```

执行结果：

```
已更新 1 行。
已更新 1 行。
已更新 1 行。
提交完成。
```

步骤4：检查LOGSAL表中记录的信息：

SELECT * FROM logerr;

执行结果为：

NUM MESSAGE

---------- ----------

1 雇员SMITH的原工资：800新工资：1300

已选择 1 行。

说明：本例中，在emp表的sal列上创建了一个由UPDATE事件触发的行级触发器，触发器的名称是LOG_SAL。该触发器由WHEN语句限定，只有当被修改工资的雇员职务为CLERK，且修改的工资超过200时才进行触发，否则不进行触发。

所以在验证过程中，虽然修改了3条记录，但通过查询语句发现：第一条修改语句修改编号为7788的SCOTT记录，因为SCOTT的职务是ANALYST，不符合WHEN条件，没有引起触发器动作；第二条修改语句修改编号为7369的SMITH的记录，职务为CLERK，因为增加的工资(500)超过了200，所以引起触发器动作，并在LOGERR表中进行了记录；第三条修改语句修改编号为7876的雇员ADAMS的记录，虽然ADAMS的职务为CLERK，但修改的工资(50)没有超过200，所以没有引起触发器动作。

注意：在WHEN条件中引用new和old不需要在前面加“:”。

在以上实例中，记录了对工资的修改超出范围的信息，但没有限制对工资的修改。那么当对雇员工资的修改幅度不满足条件时，能否直接限制对工资的修改呢？答案是肯定的。

【训练3】 创建触发器CHECK_SAL，当对职务为CLERK的雇员的工资修改超出500至2000的范围时，进行限制。

步骤1：创建和编译以下触发器：

```
CREATE OR REPLACE TRIGGER CHECK_SAL
BEFORE
UPDATE
ON emp
FOR EACH ROW
BEGIN
```

```
    IF :new.job='CLERK' AND (:new.sal<500
OR :new.sal>2000) THEN
    RAISE_APPLICATION_ERROR(-20001, '工资修改超出范围,操作取消！');
    END IF;
    END;
```

执行结果：

触发器已创建。

步骤2：在EMP表中插入记录：

```
UPDATE emp SET sal=800 WHERE empno=7876;
UPDATE emp SET sal=450 WHERE empno=7876;
COMMIT;
```

执行结果：

```
UPDATE emp SET sal=450 WHERE empno=7876
                    *
ERROR 位于第 1 行:
ORA-20001: 工资修改超出范围,操作取消!
ORA-06512: 在"STUDENT.CHECK_SAL", line 3
ORA-04088: 触发器 'STUDENT.CHECK_SAL' 执行过程中出错提交完成。
```

步骤3：检查工资的修改结果：

SELECT empno,ename,job,sal FROM emp WHERE empno=7876;

执行结果为：

EMPNO	ENAME	JOB	SAL
7876	ADAMS	CLERK	800

说明：在触发器中，当IF语句的条件满足时，即对职务为CLERK的雇员工资的修改超出指定范围时，用RAISE_APPLICATION_ERROR语句来定义一个临时定义的异常，并立即引发异常。由于触发器是BEFORE类型，因此触发器先执行，触发器因异常而终止，SQL语句的执行就会取消。

通过步骤2的执行信息可以看到，第一条语句修改编号为7876的雇员ADAMS的工资为800，成功执行。第二条语句修改雇员ADAMS的工资为450，发生异常，执行失败。这样就阻止了不符合条件的工资的修改。通过步骤3的查询可以看到，雇员ADAMS最后的工资是800，即发生异常之前的修改结果。

【练习3】限定对emp表的修改，只能修改部门10的雇员工资。

【训练4】 创建一个行级触发器CASCADE_UPDATE，当修改部门编号时，EMP表的相关行的部门编号也自动修改。该触发器称为级联修改触发器。

步骤1：创建和编译以下触发器：

```
CREATE TRIGGER CASCADE_UPDATE
AFTER
UPDATE OF deptno
ON DEPT
FOR EACH ROW
BEGIN
```

```
UPDATE EMP SET EMP.DEPTNO=:NEW.DEPTNO
 WHERE EMP.DEPTNO=:OLD.DEPTNO;
END;
```

执行结果：

触发器已创建

步骤2：验证触发器：

```
UPDATE dept SET deptno=11 WHERE deptno=10;
COMMIT;
```

执行结果：

已更新 1 行。

执行查询：

```
SELECT empno,ename,deptno FROM emp;
```

执行结果：

```
 EMPNO ENAME      DEPTNO
---------- ---------- ----------
 7369 SMITH       20
 7499 ALLEN       30
7521 WARD         30
7566 JONES        20
```

7654 MARTIN		30
7698 BLAKE		30
7782 CLARK		11
7839 KING	11	
7844 TURNER		30
7876 ADAMS		20
7900 JAMES	30	
7902 FORD	20	
7934 MILLER		11
7788 SCOTT	20	

说明：通过检查雇员的部门编号，发现原来编号为10的部门编号被修改为11。

本例中的UPDATE OF deptno表示只有在修改表的DEPTNO列时才引发触发器，对其他列的修改不会引起触发器的动作。在触发器中，对雇员表的部门编号与修改之前的部门编号一样的雇员，修改其部门编号为新的部门编号。注意，在语句中同时用到了:new和:old来引用修改部门编号前后的部门编号。

【练习4】建立级联删除触发器CASCADE_DELETE，当删除部门时，级联删除EMP表的雇员记录。

利用触发器还可以修改数据。

【训练5】 将插入的雇员的名字变成以大写字母开头。

步骤1：创建和编译以下触发器：

```
CREATE OR REPLACE TRIGGER INITCAP
BEFORE INSERT
ON EMP
FOR EACH ROW
BEGIN
 :new.ename:=INITCAP(:new.ename);
END;
```

执行结果：

触发器已创建。

步骤2：验证运行结果：

```
INSERT INTO emp(empno,ename,job,sal) VALUES(1000,'BILL','CLERK',1500);
```

执行结果：

已创建 1 行。

执行查询：

```
SELECT ename,job,sal FROM emp WHERE empno=1000;
```

执行结果：

```
ENAME      JOB            SAL
---------- ------------ ----------------------
Bill       CLERK          1500
```

说明：在本例中，通过直接为:new.ename进行赋值，修改了插入的值，但是这种用法只能在BEFORE型触发器中使用。验证结果为，在插入语句中雇员名称为大写的BILL，查询结果中雇员名称已经转换成以大写开头的Bill。

【练习5】限定一次对雇员的工资修改不超过原工资的10%。

9.2.4 语句级触发器的应用

同行级触发器不同，语句级触发器的每个操作语句不管操作的行数是多少，只触发一次触发器，所以语句级触发器适合于对整个表的操作权限等进行控制。在触发器定义中若省略FOR EACH ROW子句，则为语句级触发器。

【训练1】 创建一个语句级触发器CHECK_TIME，限定对表EMP的修改时间为周一至周五的早8点至晚5点。

步骤1：创建和编译以下触发器：

```
CREATE OR REPLACE TRIGGER CHECK_TIME
BEFORE
UPDATE OR INSERT OR DELETE
ON EMP
BEGIN
 IF (TO_CHAR(SYSDATE,'DY') IN ('SAT','SUN'))
 OR TO_CHAR(SYSDATE,'HH24')< '08'
 OR TO_CHAR(SYSDATE,'HH24')>='17' THEN
```

RAISE_APPLICATION_ERROR(-20500,'非法时间修改表错误！');

END IF;

END;

执行结果：

触发器已创建。

步骤2：当前时间为18点50分，在EMP表中插入记录：

UPDATE EMP SET SAL=3000 WHERE EMPNO=7369;

显示结果为：

UPDATE EMP SET SAL=3000 WHERE EMPNO=7369

*

ERROR 位于第 1 行:

ORA-20500: 非法时间修改表错误!

ORA-06512: 在"STUDENT.CHECK_TIME", line 5

ORA-04088: 触发器 'STUDENT.CHECK_TIME' 执行过程中出错

说明：通过引发异常限制对数据库进行的插入、删除和修改操作的时间。SYSDATE用来获取系统当前时间，并按不同的格式字符串进行转换。“DY”表示获取英文表示的星期简写，“HH24”表示获取24小时制时间的小时。

当在18点50分修改表中的数据时，由于时间在8点至17点(晚5点)之外，所以产生“非法时间修改表错误”的用户自定义错误，修改操作终止。

【练习1】设计一个语句级触发器，限定只能对EMP进行修改操作，不能对EMP进行插入和删除操作。在需要进行插入和删除时，将触发器设置为无效状态，完成后重新设置为生效状态。

9.3 数据库事件触发器

数据库事件触发器有数据库级和模式级两种。前者定义在整个数据库上，触发事件是数据库事件，如数据库的启动、关闭，对数据库的登录或退出。后者定义在模式上，触发事件包括模式用户的登录或退出，或对数据库对象的创建和修改(DDL事件)。

数据库事件触发器的触发事件的种类和级别如表9-3所示。

表9-3 数据库事件触发器的触发事件

种 类	关 键 字	说 明
模式级	CREATE	在创建新对象时触发
	ALTER	修改数据库或数据库对象时触发
	DROP	删除对象时触发
数据库级	STARTUP	数据库打开时触发
	SHUTDOWN	在使用 NORMAL 或 IMMEDIATE 选项关闭数据库时触发
	SERVERERROR	发生服务器错误时触发
数据库级与模式级	LOGON	当用户连接到数据库，建立会话时触发
	LOGOFF	当会话从数据库中断开时触发

9.3.1 定义数据库事件和模式事件触发器

创建数据库级触发器需要ADMINISTER DATABASE TRIGGER系统权限，一般只有系统管理员拥有该权限。

对于模式级触发器，为自己的模式创建触发器需要CREATE TRIGGER权限，如果是为其他模式创建触发器，需要CREATE ANY TRIGGER权限。

数据库事件和模式事件触发器的创建语法与DML触发器的创建语法类似。数据库事件或模式事件触发器的创建语法如下：

CREATE [OR REPLACE] TRIGGER 触发器名

{BEFORE|AFTER }

{DDL事件1 [DDL事件2...]| 数据库事件1 [数据库事件2...]}

ON {DATABASE| [模式名.]SCHEMA }

[WHEN (条件)]

DECLARE

声明部分

BEGIN

主体部分

END;

其中：DATABASE表示创建数据库级触发器，数据库级要给出数据库事件；SCHEMA表示创建模式级触发器，模式级要给出模式事件或DDL事件。

在数据库事件触发器中，可以使用如表9-4所示的一些事件属性。不同类型的触发器可以使用的事件属性有所不同。

表9-4 数据库事件属性

属　性	适用触发器类型	说　　明
Sys.sysevent	所有类型	返回触发器触发事件字符串
Sys.instance_num	所有类型	返回 Oracle 实例号
Sys.database_name	所有类型	返回数据库名字
Sys.server_error(stack_position)	SERVERERROR	从错误堆栈指定位置返回错误号，参数为 1 表示最近的错误
Is_servererror(error_number)	SERVERERROR	判断堆栈中是否有参数指定的错误号
Sys.login_user	所有类型	返回导致触发器触发的用户名
Sys.dictionary_obj_type	CREATE、ALTER、DROP	返回 DDL 触发器触发时涉及的对象类型
Sys. dictionary_obj_name	CREATE、ALTER、DROP	返回 DDL 触发器触发时涉及的对象名称
Sys.des_encrypted_password	CREATE、ALTER、DROP	创建或修改用户时，返回加密后的用户密码

9.3.2 数据库事件触发器

下面是一个综合的数据库事件触发器练习。先为STUDENT账户授予创建数据库事件触发器的权限，ADMINISTER DATABASE TRIGGER，然后创建有关的表和触发器，最后予以验证。

【训练1】 创建触发器，对本次数据库启动以来的用户登录时间进行记录，每次数据库启动后，先清空该表。

步骤1：创建登录事件记录表：

```
CREATE TABLE userlog (
USERNAME VARCHAR2(20),
LOGON_TIME DATE);
```

执行结果：

表已创建。

步骤2：创建数据库STARTUP事件触发器：

```
CREATE OR REPLACE TRIGGER INIT_LOGON
AFTER
STARTUP
ON DATABASE
BEGIN
 DELETE FROM userlog;
END;
```

执行结果：

触发器已创建。

步骤3：创建数据库LOGON事件触发器：

```
CREATE OR REPLACE TRIGGER
  DATABASE_LOGON
AFTER
LOGON
ON DATABASE
BEGIN
 INSERT INTO userlog
 VALUES(sys.login_user,sysdate);
END;
```

执行结果：

触发器已创建。

步骤4：验证DATABASE_LOGON触发器：

CONNECT SCOTT/TIGER@MYDB;

CONNECT STUDENT/STUDENT@MYDB;

执行结果：

已连接。

已连接。

执行查询：

SELECT username,TO_CHAR(logon_time,'YYYY/MM/DD HH24:MI:SS') FROM userlog;

执行结果：

USERNAME TO_CHAR(LOGON_TIME,

--- ---

SCOTT 2004/03/29 22:42:20

STUDENT 2004/03/29 22:42:20

步骤5：验证INIT_LOGON触发器。

重新启动数据库，登录STUDENT账户：

SELECT username,TO_CHAR(logon_time,'YYYY/MM/DD HH24:MI:SS') FROM userlog;

执行结果：

USERNAME TO_CHAR(LOGON_TIME,

STUDENT 2004/03/29 22:43:59

已选择 1 行

说明：本例中共创建了两个数据库级事件触发器。DATABASE_LOGON在用户登录时触发，向表userlog中增加一条记录，记录登录用户名和登录时间。INIT_LOGON在数据库启动时触发，清除userlog表中记录的数据。所以当数据库重新启动后，重新登录STUDENT账户，此时userlog表中只有一条记录。

【训练2】 创建STUDENT_LOGON模式级触发器，专门记录STUDENT账户的登录时间：

```
CREATE OR REPLACE TRIGGER
  STUDENT_LOGON
AFTER
LOGON ON STUDENT.SCHEMA
BEGIN
 INSERT INTO userlog
 VALUES(sys.login_user,sysdate);
END;
```

执行结果：

触发器已创建。

说明：为当前模式创建触发器，可以省略SCHEMA前面的模式名。

【练习1】修改DATABASE_LOGON触发器和userlog表，增加对退出时间的记录。

9.4 DDL事件触发器

【训练1】 通过触发器阻止对emp表的删除。

步骤1：创建DDL触发器：

```
CREATE OR REPLACE TRIGGER NODROP_EMP
 BEFORE
DROP ON SCHEMA
BEGIN
IF Sys.Dictionary_obj_name='EMP' THEN
```

RAISE_APPLICATION_ERROR(-20005,'错误信息：不能删除emp表！');

END IF;

END;

执行结果：

触发器已创建。

步骤2：通过删除emp表验证触发器：

DROP TABLE emp;

执行结果：

DROP TABLE emp

*

ERROR 位于第 1 行:

ORA-00604: 递归 SQL 层 1 出现错误

ORA-20005: 错误信息：不能删除emp表！

ORA-06512: 在line 3

说明：该触发器阻止在当前模式下对emp表的删除，但不阻止删除其他对象。Sys.Dictionary_obj_name属性返回要删除的对象名称。

9.5 替代触发器

【训练1】 在emp表的视图上，通过触发器修改emp表。

步骤1：创建视图emp_name：

CREATE VIEW emp_name AS SELECT ename FROM emp;

执行结果：

视图已建立。

步骤1：创建替代触发器：

CREATE OR REPLACE TRIGGER change_name

INSTEAD OF INSERT ON emp_name

DECLARE

```
V_EMPNO NUMBER(4);
BEGIN
SELECT MAX(EMPNO)+1 INTO V_EMPNO FROM EMP;
INSERT INTO emp(empno,ename)
VALUES(V_EMPNO,:new.ename);
END;
```

执行结果：

触发器已创建。

步骤2：向emp_name视图插入记录：

```
INSERT INTO emp_name VALUES('BROWN');
COMMIT;
```

执行结果：

```
已创建 1 行。
提交完成。
```

说明：向视图直接插入雇员名将会发生错误，因为emp表的雇员编号列不允许为空。通过创建替代触发器，将向视图插入雇员名称转换为向emp表插入雇员编号和雇员名称，雇员编号取当前的最大雇员编号加1。试检查emp表的雇员列表。

【训练2】 在emp表的视图emp_name上，通过触发器阻止对emp表的删除。

步骤1：阻止通过视图删除雇员，并显示用户自定义错误信息：

```
CREATE OR REPLACE TRIGGER delete_from_ename
INSTEAD OF DELETE ON emp_name
BEGIN
 RAISE_APPLICATION_ERROR(-20006,'错误信息：不能在视图中删除emp表的雇员！');
END;
```

执行结果：

触发器已创建。

步骤2：通过对视图进行删除来验证触发器：

DELETE FROM emp_name;

执行结果：

DELETE FROM emp_name

*

ERROR 位于第 1 行:

ORA-20006: 错误信息：不能在视图中删除emp表的雇员！

ORA-06512:

在"STUDENT.DELETE_FROM_ENAME", line 2

ORA-04088: 触发器 'STUDENT.DELETE_FROM_ENAME' 执行过程中出错

说明：可以通过视图emp_name对雇员进行删除，比如执行DELETE FROM emp_name语句将删除雇员表的全部雇员。但是由于在emp_name视图中只能看到一部分雇员信息，所以删除可能会产生误操作。通过定义一个替代触发器，可阻止通过emp_name视图对emp表雇员进行删除，但不阻止直接对emp表进行删除。

9.6 查看触发器

【训练1】 显示触发器CHECK_TIME的体部分：

SELECT TRIGGER_BODY FROM USER_TRIGGERS WHERE TRIGGER_NAME='CHECK_TIME';

结果为：

TRIGGER_BODY

BEGIN

IF (TO_CHAR(SYSDATE,'DY') IN ('SAT','SUN'))

OR TO_CHAR(SYSDATE,'HH24')<

TRIGGER_BODY字段为LONG类型，只显示出脚本的一部分内容。

9.7 阶段训练

【训练1】 创建触发器，进行表的同步复制。

步骤1：创建emp表的复本employee:

CREATE TABLE employee AS SELECT * FROM emp;

执行结果：

表已创建。

步骤2：创建和编译以下触发器:

```
CREATE OR REPLACE TRIGGER DUPLICATE_EMP
AFTER
UPDATE OR INSERT OR DELETE
ON EMP
FOR EACH ROW
BEGIN
 IF INSERTING THEN
 INSERT INTO employee

VALUES(:new.empno,:new.ename,:new.job,:new.mgr,
 :new.hiredate,:new.sal,:new.comm,:new.deptno);
```

```
ELSIF DELETING THEN
 DELETE FROM employee
 WHERE empno=:old.empno;
ELSE
 UPDATE employee SET
 empno=:new.empno,
 ename=:new.ename,
 job=:new.job,
```

```
mgr=:new.mgr,
  hiredate=:new.hiredate,
  sal=:new.sal,
  comm=:new.comm,
  deptno=:new.deptno
  WHERE empno=:old.empno;
 END IF;
END;
```

执行结果：

触发器已创建。

步骤3：对emp表进行插入、删除和更新：

```
DELETE FROM emp WHERE empno=7934;
INSERT INTO emp(empno,ename,job,sal)
VALUES(8888,'ROBERT','ANALYST',2900);
UPDATE emp SET sal=3900 WHERE empno=7788;
COMMIT;
```

执行结果：

```
已删除 1 行。
已创建 1 行。
已更新 1 行。
提交完成。
```

步骤4：检查emp表和employee表中被插入、删除和更新的雇员。

运行结果略，请自行验证。

说明：在触发器中判断触发事件，根据不同的事件对employee表进行不同的操作。

【练习1】创建一个emp表的触发器EMP_TOTAL，每当向雇员表插入、删除或更新雇员信息时，将新的统计信息存入统计表EMPTOTAL，使统计表总能够反映最新的统计信息。

统计表是记录各部门雇员总人数、总工资的统计表，结构如下：

部门编号 number(2)

总人数 number(5)

总工资 number(10,2)

9.8 练习

1. 下列有关触发器和存储过程的描述，正确的是：

A. 两者都可以传递参数

B. 两者都可以被其他程序调用

C. 两种模块中都可以包含数据库事务语句

D. 两者创建的系统权限不同

2. 下列事件，属于DDL事件的是：

A. INSERT　　B. LOGON

C. DROP　　D. SERVERERROR

3. 假定在一个表上同时定义了行级和语句级触发器，在一次触发当中，下列说法正确的是：

A. 语句级触发器只执行一次

B. 语句级触发器先于行级触发器执行

C. 行级触发器先于语句级触发器执行

D. 行级触发器对表的每一行都会执行一次

4. 有关行级触发器的伪记录，下列说法正确的是：

A. INSERT事件触发器中，可以使用:old伪记录。

B. DELETE事件触发器中，可以使用:new伪记录。

C. UPDATA事件触发器中，只能使用:new伪记录。

D. UPDATA事件触发器中，可以使用:old伪记录。

5. 下列有关替代触发器的描述，正确的是：

A. 替代触发器创建在表上

B. 替代触发器创建在数据库上

C. 通过替代触发器可以向基表插入数据

D. 通过替代触发器可以向视图插入数据