③ 随机设置顾客的国籍。

```
/*首先创建存储过程 setCustomerNations(),然后执行该存储过程*/
CREATE OR REPLACE PROCEDURE Sales.setCustomerNations() AS
DECLARE
    nationCount INT;
    res RECORD;  /*定义游标结果的变量 res 为记录类型*/
    CURSOR mycursor FOR SELECT custkey FROM Sales.Customer;
BEGIN
    SELECT COUNT(*) INTO nationCount FROM Sales.Nation;/*计算 Nation 表中的记录个数*
    OPEN mycursor;
    LOOP
        FETCH mycursor INTO res;/*从游标中取出的结果保存在 res 变量中*/
        IF mycursor%NOTFOUND THEN
            EXIT;
        END IF;
        UPDATE Sales.Customer /*随机设置顾客的国籍*/
        SET nationkey=(SELECT nationkey
                       FROM Sales.Nation LIMIT 1
                       OFFSET MOD(CAST(RANDOM()*1000 AS INTEGER),nationCount)
        WHERE custkey=res.custkey;
    END LOOP;
    CLOSE mycursor;
END;
CALL sales.setCustomerNations();/*调用存储过程设置顾客的国籍*/
```

④ 随机设置供应商的国籍。

```
/*首先创建存储过程 setSupplierNations(),然后执行该存储过程*/
CREATE OR REPLACE PROCEDURE sales.setSupplierNations() AS
```

```
DECLARE
nationCount INT;
res RECORD;                    /*定义游标结果为记录类型变量*/
CURSOR mycursor FOR SELECT suppkey FROM Sales.Supplier;
BEGIN
    SELECT COUNT(*) INTO nationCount FROM Sales.Nation;/*计算 Nation 表中的记录个数*/
    OPEN mycursor;
    LOOP
        FETCH mycursor INTO res;/*从游标中取出的结果保存在 res 记录类型的变量中*/
        IF mycursor%NOTFOUND THEN
            EXIT;
        END IF;
        UPDATE Sales.Supplier                    /*随机设置供应商的国籍*/
        SET nationkey=(SELECT nationkey
            FROM Sales.Nation
            LIMIT 1
            OFFSET MOD(CAST(RANDOM()*1000 AS INTEGER),nationCount))
        WHERE suppkey=res.suppkey;
    END LOOP;
    CLOSE mycursor;
END;
CALL sales.setSupplierNations();      /*调用存储过程设置供应商的国籍*/
```

⑤ 随机生成 PartSupp 数据。

```
/*随机生成零件供应联系表 PartSupp 记录的存储过程:参数 p_partsuppCount 为需要产生的记录数*/
CREATE OR REPLACE PROCEDURE sales.insert_PartSupp(p_partsuppCount INT) AS
DECLARE
    supplierCount INT;
    partCount INT;
    NewPartKey INT;
    NewSuppKey INT;
    tmp INT;
BEGIN
    SELECT COUNT(*) INTO partCount FROM Sales.Part;/*计算 Part 表的记录数*/
    SELECT COUNT(*) INTO supplierCount FROM Sales.Supplier;/*计算 Supplier 表的记录数*/
    /*循环生成 PartSupp 记录:首先随机生成 partkey 和 suppkey,若 PartSupp 中不存在相应的记
    录,则增加一条新的记录*/
    FOR i IN 1..p_partsuppCount LOOP
```

```
            NewPartKey := (SELECT partkey                    /* 随机获得一个合法的零件编号 */
                    FROM Sales.Part
                    LIMIT 1
                    OFFSET
                    MOD(CAST(RANDOM() * 100 * partCount AS INTEGER),partCount));
            NewSuppKey := (SELECT suppkey                    /* 随机获得一个合法的供应商编号 */
                    FROM Sales.Supplier
                    LIMIT 1
                    OFFSET
                    MOD(CAST(RANDOM() * 100 * supplierCount AS INTEGER),supplierCount));
            SELECT partkey INTO tmp                          /* 查询将生成的零件供应记录是否已经存在 */
            FROM Sales.PartSupp
            WHERE partkey=NewPartKey AND suppkey=NewSuppKey;
              IF SQL%NOTFOUND THEN
                INSERT INTO Sales.PartSupp                   /* 插入新的零件供应记录 */
                VALUES(NewPartKey,NewSuppKey,CAST(RANDOM() * 1000 AS INTEGER),CAST(RAN-
                DOM() * 10000 AS REAL));
              END IF;
            /* 循环 200 次,插入 200 条记录,就提交一次事务,以节省内存,提高执行效率 */
            IF (MOD(p_partsuppCount,200)= 0 ) THEN
                  COMMIT;
            END IF;
        END LOOP;
    END;
    CALL Sales.insert_PartSupp(30000);                       /* 执行存储过程 */
```

⑥ 随机生成 oders 数据。

```
    /* 随机生成订单表 Orders 记录的存储过程:参数 p_orderCount 为需要产生的记录数 */
    CREATE OR REPLACE PROCEDURE sales.insert_Orders(p_orderCount INT) AS
    DECLARE
        customerCount INT;
        existingMaxOrderKey INT;
        NewOrderKey INT;
        NewCustKey INT;
        tmp INT;
        i INT;
    BEGIN
        SELECT COUNT( * ) INTO customerCount FROM Sales.Customer;/* 计算 Supplier 表的记录数 */
```

```
    SELECT MAX(orderkey) INTO existingMaxOrderKey FROM Sales.Orders;
                                                    /* 计算 Orders 表已有的最大 orderkey */
    IF existingMaxOrderKey IS NULL THEN
        existingMaxOrderKey = 0;
    END IF;
    /* 循环产生订单记录:首先找到已有订单中的最大订单号,在此基础上生成新的订单记录号 */
    FOR i IN 1..p_orderCount LOOP
        NewOrderKey := existingMaxOrderKey+i;
        NewCustKey = (SELECT custkey      /* 随机获得一个客户编号 */
                      FROM Sales.Customer
                      LIMIT 1
                      OFFSET
                      MOD(CAST(RANDOM() * 100 * customerCount AS INTEGER),
                          customerCount));
        INSERT INTO Sales.Orders(orderkey,custkey,orderdate)   /* 插入一条订单记录 */
            VALUES(NewOrderKey,NewCustKey,
            DATEADD('day',MOD( CAST( RANDOM() * 1000 AS INTEGER),365),
            CURRENT_DATE));
    /* 循环 200 次,插入 200 条记录,就提交一次事务,以节省内存提高执行效率 */
    IF (MOD(p_orderCount,200)= 0 ) THEN
        COMMIT;
    END IF;
    END LOOP;
END;
CALL sales.insert_Orders(30000);          /* 执行存储过程 */
```

⑦ 随机生成 Lineitem 数据。

```
/* 随机生成订单明细表 Lineitem 记录的存储过程:参数 p_orderCount 为需要产生订单明细的订单数 */
CREATE OR REPLACE PROCEDURE sales.insert_Lineitem(p_orderCount INT) AS
    DECLARE
        NewOrderKey INT;
        NewPartKey INT;
        NewSuppKey INT;
        L_orderloop INT;
        L_linenumber   INT;
        L_existingMaxLinenumber INT;
        partsuppCount INT;
        lineitemCount   INT;
```

```
    CURSOR mycursor FOR SELECT orderkey FROM sales.Orders;
BEGIN
    SELECT COUNT( * ) INTO partsuppCount FROM Sales.PartSupp;/ * 计算 PartSupp 的记录数 * /
    OPEN mycursor;
    IF mycursor%ISOPEN THEN
      FOR L_orderloop IN 1..p_orderCount LOOP
       / * 获取 Orders 订单表的记录,如果没有记录,或者是已经产生足够的订单明细了,
       就退出循环 * /
       FETCH mycursor INTO NewOrderKey;
       EXIT WHEN (mycursor%NOTFOUND);
       / * 计算 Lineitem 表中当前订单的最大的 linenumber,在此基础上产生新的 linenum-
       ber 号 * /
       SELECT MAX(linenumber) INTO L_existingMaxLinenumber FROM Sales.Lineitem
       WHERE orderkey=NewOrderKey;
       IF L_existingMaxLinenumber IS NULL THEN
         L_existingMaxLinenumber=0;
       END IF;
       / * 随机设置每个订单产生的订单明细记录条数:最多产生 3 条明细记录 * /
       lineitemCount:=MOD(CAST(RANDOM() * 1000 AS INTEGER),3);
       FOR L_linenumber IN 1..lineitemCount LOOP/ * 循环产生当前订单的订单明细记录 * /
       / * 随机生成订单明细记录中所购买的对应零件供应联系表中的记录号 * /
            SELECT partkey,suppkey INTO NewPartKey,NewSuppKey
            FROM sales.partsupp LIMIT 1 OFFSET MOD(CAST(RANDOM() * 1000000
                 AS INTEGER),partsuppCount );
            INSERT INTO sales.Lineitem (orderkey,partkey,suppkey,linenumber,quantity,
                                        discount,extendedprice)      / * 插入订单明细
                                        记录 * /
            VALUES(NewOrderKey,NewPartKey,NewSuppKey,L_linenumber+L_existing-
                   MaxLinenumber,CAST(RANDOM() * 100 AS INTEGER),RAN-
                   DOM(),0);
     END LOOP;
     / * 循环 200 次,插入 200 条记录,就提交一次事务,以节省内存提高执行效率 * /
     IF (MOD(p_orderCount,200)=0 ) THEN
        COMMIT;
     END IF;
    END LOOP;
  CLOSE mycursor;
```

```
        END IF;
END;
CALL sales.insert_Lineitem(20000);          /*执行存储过程 */
UPDATE Sales.Lineitem                       /*设置订单明细记录中的销售价格 extendedprice */
SET extendedprice=quantity * Part.retailprice
    FROM Sales.Part WHERE Sales.Part.partkey=Sales.Lineitem.partkey;
```