数据库系统课程实验报告

实验名称:	数据更新
实验日期:	2022/5/26
实验地点:	四号楼
提交日期:	2022/5/26

学号:	22920202202879
姓名:	陈奕培
专业年级:	软工 2020 级
学年学期:	2021-2022 学年第二学期

1.实验目的

- 熟练掌握单条记录和小批量数据插入的方法（INSERT）
- 熟练掌握使用子查询实现数据插入的方法(INSERT INTO…SUBQUERY)
- 熟练掌握数据修改和删除的方法（UPDATE,DELETE,TRUNCATE）

2.实验内容和步骤

（1）为地区表 regions 新增一条记录：（‘5’,’Oceania’）。

```
INSERT INTO regions VALUES ('5','Oceania');
```

```
sales=# INSERT INTO regions VALUES ('5','Oceania');
INSERT 0 1
```

（2）将 countries 表中的国家名为 Austrialia 的 region_id 改为 5。

```
UPDATE countries SET region_id='5' WHERE country_name='Australia';
```

```
sales=# UPDATE countries SET region_id='5' WHERE country_name='Australia';
UPDATE 1
```

（3）使用一条批量插入数据语句为 countries 表新增 5 条记录：('NO','Norway','1'), ('ES','Spain','1'),('SE','Sweden','1'), ('PT','Portugal','1'), ('NZ','New Zealand','5')。

```
INSERT INTO countries VALUES ('NO','Norway','1'), ('ES','Spain','1'),
('SE','Sweden','1'), ('PT','Portugal','1'), ('NZ','New Zealand','5');
```

```
UPDATE 1
sales=# INSERT INTO countries VALUES ('NO','Norway','1'), ('ES','Spain','1'),
('SE','Sweden','1'), ('PT','Portugal','1'), ('NZ','New Zealand','5');
sales-# INSERT 0 5
```

（4）创建一张名为 Asia_countries(country_id,country_name)的新表，其中字段为 countries 表 中的同名字段。

```
CREATE TABLE asia_countries (
country_id CHAR(2),
country_name VARCHAR2(40));
```

```
sales=#
sales=# CREATE TABLE asia_countries (
country_id CHAR(2),
country_name VARCHAR2(40));sales(# sales(#
CREATE TABLE
```

（5）将 countries 表中所有亚洲国家的数据插入到该表中。(要求使用插入子查询结果的方法实现)

```
INSERT INTO asia_countries
SELECT country_id,country_name
FROM countries,regions
WHERE regions.region_id=countries.region_id AND region_name='Asia';
```

```
sales=#
sales=# INSERT INTO asia_countries
SELECT country_id,country_name
FROM countries,regions
WHERE regions.region_id=countries.region_id AND region_name='Asia';
INSERT 0 5
```

（6）创建一张名为 order_total(order_id,total_price)的视图，该视图存放每个订单号及其总价，其中 total_price 为总价，其值为数量 quantity 与单价 unit_price 乘积之和，order_id，quantity 和 unit_price 为 order_items 表中的同名字段。

```
CREATE VIEW order_total(order_id,total_price) AS
SELECT order_id,SUM(quantity*unit_price) AS total_price
FROM order_items
GROUP BY order_id;
```

```
sales=#
sales=# CREATE VIEW order_total(order_id,total_price) AS
SELECT order_id,SUM(quantity*unit_price) AS total_price
FROM order_items
GROUP BY order_id;sales-# sales-# sales-#
CREATE VIEW
```

（7）查询 order_total 视图中订单号 order_id 为 97 的总价并记录该结果。

```
SELECT * FROM order_total WHERE order_id='97';
```

```
sales=# SELECT * FROM order_total WHERE order_id='97'
sales-# ;
 order_id | total_price
----------+-------------
       97 | 616763.1900
(1 row)
```

（8）将 order_items 表中 product_id 为 99 的单价 unit_price 增加 4 元。

```
UPDATE order_items SET unit_price=unit_price+4 WHERE product_id='99';
```

```
sales=# UPDATE order_items SET unit_price=unit_price+4 WHERE product_id='99';
UPDATE 2
sales=#
```

（9）查询视图 order_total 中订单号 order_id 为 97 的总价，将其与第（7）步的结果进行比较，观察其异同。

```
SELECT * FROM order_total WHERE order_id='97';
```

```
sales=# SELECT * FROM order_total WHERE order_id='97';
 order_id | total_price
----------+-------------
       97 | 616955.1900
(1 row)
```

总价提升，视图会跟着表一起修改

（10）使用 delete 命令删除 Asia_countries 表中 country_id 为 IN 的记录。

```
DELETE FROM asia_countries WHERE country_id='IN';
```

```
sales=# DELETE FROM asia_countries WHERE country_id='IN';
DELETE 1
```

（11）使用 truncate 命令清空 Asia_countries 表的所有记录。

```
truncate asia_countries;
```

```
sales=# truncate asia_countries;
TRUNCATE TABLE
sales=#
```

（12）删除 Asia_countries 表和视图 order_total。

```
DROP TABLE asia_countries;
DROP VIEW order_total;
```

```
sales=# DROP TABLE asia_countries;
DROP VIEW order_total;DROP TABLE
sales=#
DROP VIEW
```

（13）使用命令\d employees 查看 employees 表的外码约束语句，包括 on delete cascade 选项。

```
sales=# \d employees;
                Table "public.employees"
   Column    |              Type              | Modifiers
-------------+--------------------------------+-----------
 employee_id | numeric                        | not null
 first_name  | character varying(255)         |
 last_name   | character varying(255)         |
 email       | character varying(255)         |
 phone       | character varying(20)          |
 hire_date   | timestamp(0) without time zone |
 manager_id  | numeric                        |
 job_title   | character varying(255)         |
Indexes:
    "employees_pk" PRIMARY KEY, btree (employee_id) TABLESPACE pg_default
Foreign-key constraints:
    "fk_employees_manager" FOREIGN KEY (manager_id) REFERENCES employees(employee_id) ON DELETE CASCADE
Referenced by:
    TABLE "employees" CONSTRAINT "fk_employees_manager" FOREIGN KEY (manager_id) REFERENCES employees(employee_id) ON DELETE CASCADE
```

（14）查询 employees 表中 manager_id 为 1 的记录。

```
SELECT * FROM employees WHERE employee_id='1';
```

```
sales=# SELECT * FROM employees WHERE employee_id='1';
 employee_id | first_name | last_name |         email          |    phone     |      hire_date      | manager_id | job_title
-------------+------------+-----------+------------------------+--------------+---------------------+------------+-----------
           1 | Tommy      | Bailey    | tommy.bailey@example.com | 515.123.4567 | 2016-06-17 00:00:00 |            | President
(1 row)
```

（15）修改 employees 表的外码约束，去掉外码约束中的 on delete

cascade 选项，但保留原有的外 码引用，即 manager_id 引用本表上的 employee_id。（可通过先删后建实现）

```
ALTER TABLE employees DROP CONSTRAINT fk_employees_manager;
ALTER TABLE employees ADD CONSTRAINT fk_employees_manager
FOREIGN KEY(manager_id) REFERENCES employees(employee_id);
```

```
sales=# ALTER TABLE employees DROP CONSTRAINT fk_employees_manager;
ALTER TABLE
sales=# ALTER TABLE employees ADD CONSTRAINT fk_employees_manager
FOREIGN KEY(manager_id) REFERENCES employees(employee_id);sales-#
ALTER TABLE
sales=#
```

（16）删除 employees 表中 employee_id 为 1 的记录，观察操作结果。

```
DELETE FROM employees WHERE employee_id='1';
```

```
sales=# DELETE FROM employees WHERE employee_id='1';
ERROR:  update or delete on table "employees" violates foreign key constraint "fk_employees_manager" on ta
DETAIL:  Key (employee_id)=(1) is still referenced from table "employees".
```

删除失败，因为表内包含了对该条记录的外键约束

（17）修改 employees 表的外码约束，增加 on delete cascade 选项，即回到最初的外码约束状态。

```
ALTER TABLE employees DROP CONSTRAINT fk_employees_manager;
ALTER TABLE employees ADD CONSTRAINT fk_employees_manager
FOREIGN KEY(manager_id) REFERENCES employees(employee_id) on delete
cascade;
```

```
sales=# ALTER TABLE employees DROP CONSTRAINT fk_employees_manager;
ALTER TABLE employees ADD CONSTRAINT fk_employees_manager
FOREIGN KEY(manager_id) REFERENCES employees(employee_id) on delete cascade;
ALTER TABLE
sales=# sales-# ALTER TABLE
```

（18）再次执行第（16）步，观察操作结果。

```
DELETE FROM employees WHERE employee_id='1';
```

```
sales=# DELETE FROM employees WHERE employee_id='1';
DELETE 1
sales=#
```

删除成功，将对该条的引用级联删除

思考：

建立测试表（无 ON UPDATE CASCADE）

```
CREATE TABLE test(
    id CHAR(4) PRIMARY KEY,
    father_id CHAR(4),
    CONSTRAINT test_fk FOREIGN KEY(father_id)
    REFERENCES test(id) ON UPDATE CASCADE
);
```

插入数据

```
INSERT INTO test VALUES('1',NULL),('2','1'),('3','1');
```

对 id=1 进行修改

```
sales=# UPDATE test SET id='4' WHERE id='1';
ERROR:  update or delete on table "test" violates foreign key constraint "test_fk" on table "test"
DETAIL:  Key (id)=(1   ) is still referenced from table "test".
sales=#
```

无法修改

加上 ON UPDATE CASCADE 后，再对 id=1 修改

```
ALTER TABLE test DROP CONSTRAINT test_fk;
ALTER TABLE test ADD CONSTRAINT test_fk FOREIGN KEY(father_id)
REFERENCES test(id) ON UPDATE CASCADE;
```

```
sales=# UPDATE test SET id='4' WHERE id='1';
UPDATE 1
sales=# select * from test;
  id  | father_id
------+-----------
 4    |
 2    | 4
 3    | 4
(3 rows)
```

此时其他记录级联修改，加快效率

3.实验总结

3.1 完成的工作

建表;

建视图;

插入数据;

修改外码;

级联删除;

3.2 对实验的认识

通过学习，掌握了:

单条记录的插入、

批量记录的插入、

通过 select 语句插入、

删除命令 truncate 和 delete、

级联删除和修改;

3.3 遇到的困难及解决方法

无