

数据库实验报告

李甘 2023202296

实验一 基于文件系统的商城库存管理系统

1. 实现方法

系统以CSV文件作为持久化介质，商品与进销记录分别存储于products.csv与records.csv；内存中加载为结构体列表，按需检索、排序与统计，商品删除采用逻辑删除。所有增删改查通过顺序读写与追加实现，并对时间与数量等输入做基本校验。

商品数据结构

字段	含义	示例
id	商品编号（唯一）	P001
name	商品名称	Apple
category	商品类别	Fruit
stock	当前库存	120
deleted	逻辑删除标记	false

进销记录数据结构

字段	含义	示例
productId	商品编号	P001
productName	商品名称（冗余便于历史可读）	Apple
type	记录类型（进货/销售）	Purchase
operatorName	操作人	Alice
timestamp	时间（YYYY-MM-DD HH:MM:SS）	2025-10-10 09:30:00
quantity	数量（正整数）	50

2. 运行截图

```
==== 商城库存管理 ====
```

- 1) 添加商品
- 2) 按类别浏览 (库存排序)
- 3) 进货
- 4) 销售
- 5) 删除商品
- 6) 查询进销记录
- 7) 销量汇总
- 8) 列出所有商品
- 0) 退出

选择： 1

商品编号： 0

商品名称： A

商品类别： C1

初始库存(整数) : 100

添加成功

```
==== 商城库存管理 ====
```

- 1) 添加商品
- 2) 按类别浏览 (库存排序)
- 3) 进货
- 4) 销售
- 5) 删除商品
- 6) 查询进销记录
- 7) 销量汇总
- 8) 列出所有商品
- 0) 退出

选择： 2

类别： C1

按库存排序(asc/desc): asc

ID	名称	类别	库存
0	A	C1	100

```
==== 商城库存管理 ====
```

- 1) 添加商品
- 2) 按类别浏览 (库存排序)
- 3) 进货
- 4) 销售
- 5) 删除商品
- 6) 查询进销记录
- 7) 销量汇总
- 8) 列出所有商品
- 0) 退出

选择： 3

商品编号： 0

数量： 100

操作人： P1

时间(YYYY-MM-DD HH:MM:SS): 2026-01-01 00:00:00

进货成功

```
==== 商城库存管理 ====
1) 添加商品
2) 按类别浏览 (库存排序)
3) 进货
4) 销售
5) 删除商品
6) 查询进销记录
7) 销量汇总
8) 列出所有商品
0) 退出
选择: 4
商品编号: 0
数量: 1
操作人: P2
时间(YYYY-MM-DD HH:MM:SS): 2026-01-02 00:00:00
销售成功
```

```
==== 商城库存管理 ====
1) 添加商品
2) 按类别浏览 (库存排序)
3) 进货
4) 销售
5) 删除商品
6) 查询进销记录
7) 销量汇总
8) 列出所有商品
0) 退出
选择: 8
ID      名称    类别    库存
0       A        C1      199
```

```
==== 商城库存管理 ====
1) 添加商品
2) 按类别浏览 (库存排序)
3) 进货
4) 销售
5) 删除商品
6) 查询进销记录
7) 销量汇总
8) 列出所有商品
0) 退出
选择: 6
按商品编号过滤(可留空):
按操作人过滤(可留空):
开始时间(可留空):
结束时间(可留空):
商品    类型    操作人    时间    数量
A      进货     P1      2026-01-01 00:00:00      100
A      销售     P2      2026-01-02 00:00:00      1
```

```
==== 商城库存管理 ====
1) 添加商品
2) 按类别浏览 (库存排序)
3) 进货
4) 销售
5) 删除商品
6) 查询进销记录
7) 销量汇总
8) 列出所有商品
0) 退出
选择: 7
按类别汇总(可留空):
开始时间(可留空):
结束时间(可留空):
总销量: 1
```

```
==== 商城库存管理 ====
1) 添加商品
2) 按类别浏览 (库存排序)
3) 进货
4) 销售
5) 删除商品
6) 查询进销记录
7) 销量汇总
8) 列出所有商品
0) 退出
选择: 5
商品编号: 0
已删除(逻辑)
```

```
==== 商城库存管理 ====
1) 添加商品
2) 按类别浏览 (库存排序)
3) 进货
4) 销售
5) 删除商品
6) 查询进销记录
7) 销量汇总
8) 列出所有商品
0) 退出
选择: 8
ID      名称      类别      库存
```

实验二 数据库系统的使用

我在我的安装有 Fedora Linux 的笔记本电脑上用 PostgreSQL 完成了下面的实验。

1. 安装并启动数据库

```
sudo dnf install postgresql-server postgresql-contrib
sudo /usr/bin/postgresql-setup --initdb
sudo systemctl status postgresql
```

```
sudo systemctl start postgresql
sudo -u postgres psql
```

```
nictheboy@laptop ~ [1]> sudo dnf install postgresql-server postgresql-contrib
Updating and loading repositories:
  Visual Studio Code
  RPM Fusion for Fedora 43 - Nonfree - Updates
  RPM Fusion for Fedora 43 - Free - Updates
  Fedora 43 - x86_64 - Updates
  Docker CE Stable - x86_64
Repositories loaded.
Package "postgresql-server-18.1-1.fc43.x86_64" is already installed.
Package "postgresql-contrib-18.1-1.fc43.x86_64" is already installed.

Nothing to do.
nictheboy@laptop ~> sudo /usr/bin/postgresql-setup --initdb
 * Initializing database in '/var/lib/pgsql/data'
 * Initialized, logs are in '/var/lib/pgsql/initdb_postgresql.log'
nictheboy@laptop ~> sudo systemctl status postgresql
● postgresql.service - PostgreSQL database server
  Loaded: loaded (/usr/lib/systemd/system/postgresql.service; disabled; preset: disabled)
  Drop-In: /usr/lib/systemd/system/service.d
            └─10-timeout-abort.conf
    Active: inactive (dead)
nictheboy@laptop ~ [3]> sudo systemctl start postgresql
nictheboy@laptop ~> sudo -u postgres psql
psql (18.1)
Type "help" for help.

postgres=#
```

2. 创建表结构

```
CREATE TABLE nation ( N_NATIONKEY INTEGER NOT NULL,
N_NAME CHAR(25) NOT NULL,
N_REGIONKEY INTEGER NOT NULL,
N_COMMENT VARCHAR(152));

CREATE TABLE region ( R_REGIONKEY INTEGER NOT NULL,
R_NAME CHAR(25) NOT NULL,
R_COMMENT VARCHAR(152));

CREATE TABLE part ( P_PARTKEY      INTEGER NOT NULL,
P_NAME          VARCHAR(55) NOT NULL,
P_MFGR          CHAR(25) NOT NULL,
P_BRAND         CHAR(10) NOT NULL,
P_TYPE          VARCHAR(25) NOT NULL,
P_SIZE          INTEGER NOT NULL,
P_CONTAINER     CHAR(10) NOT NULL,
P_RETAILPRICE  DECIMAL(15,2) NOT NULL,
P_COMMENT       VARCHAR(23) NOT NULL );

CREATE TABLE supplier ( S_SUPPKEY      INTEGER NOT NULL,
S_NAME          CHAR(25) NOT NULL,
S_ADDRESS       VARCHAR(40) NOT NULL,
S_NATIONKEY     INTEGER NOT NULL,
S_PHONE         CHAR(15) NOT NULL,
S_ACCTBAL      DECIMAL(15,2) NOT NULL,
S_COMMENT       VARCHAR(101) NOT NULL);

CREATE TABLE partsupp ( PS_PARTKEY      INTEGER NOT NULL,
PS_SUPPKEY      INTEGER NOT NULL,
PS_AVAILQTY     INTEGER NOT NULL,
PS_SUPPLYCOST   DECIMAL(15,2) NOT NULL,
PS_COMMENT      VARCHAR(199) NOT NULL );
```

```
CREATE TABLE customer ( C_CUSTKEY      INTEGER NOT NULL,
C_NAME          VARCHAR(25) NOT NULL,
C_ADDRESS       VARCHAR(40) NOT NULL,
C_NATIONKEY     INTEGER NOT NULL,
C_PHONE         CHAR(15) NOT NULL,
C_ACCTBAL      DECIMAL(15,2) NOT NULL,
C_MKTSEGMENT   CHAR(10) NOT NULL,
C_COMMENT       VARCHAR(117) NOT NULL);

CREATE TABLE orders  ( O_ORDERKEY      INTEGER NOT NULL,
O_CUSTKEY        INTEGER NOT NULL,
O_ORDERSTATUS    CHAR(1) NOT NULL,
O_TOTALPRICE    DECIMAL(15,2) NOT NULL,
O_ORDERDATE      DATE NOT NULL,
O_ORDERPRIORITY  CHAR(15) NOT NULL,
O_CLERK          CHAR(15) NOT NULL,
O_SHIPPRIORITY   INTEGER NOT NULL,
O_COMMENT        VARCHAR(79) NOT NULL);

CREATE TABLE lineitem ( L_ORDERKEY      INTEGER NOT NULL,
L_PARTKEY        INTEGER NOT NULL,
L_SUPPKEY        INTEGER NOT NULL,
L_LINENUMBER    INTEGER NOT NULL,
L_QUANTITY       DECIMAL(15,2) NOT NULL,
L_EXTENDEDPRICE DECIMAL(15,2) NOT NULL,
L_DISCOUNT      DECIMAL(15,2) NOT NULL,
L_TAX            DECIMAL(15,2) NOT NULL,
L_RETURNFLAG    CHAR(1) NOT NULL,
L_LINESTATUS    CHAR(1) NOT NULL,
L_SHIPDATE       DATE NOT NULL,
L_COMMITDATE    DATE NOT NULL,
L_RECEIPTDATE   DATE NOT NULL,
L_SHIPINSTRUCT  CHAR(25) NOT NULL,
L_SHIPMODE       CHAR(10) NOT NULL,
L_COMMENT        VARCHAR(44) NOT NULL);
```

```
O_ORDERSTATUS    CHAR(1) NOT NULL,
O_TOTALPRICE     DECIMAL(15,2) NOT NULL,
O_ORDERDATE      DATE NOT NULL,
O_ORDERPRIORITY   CHAR(15) NOT NULL,
O_CLERK          CHAR(15) NOT NULL,
O_SHIPPRIORITY    INTEGER NOT NULL,
O_COMMENT         VARCHAR(79) NOT NULL);

CREATE TABLE lineitem ( L_ORDERKEY      INTEGER NOT NULL,
L_PARTKEY        INTEGER NOT NULL,
L_SUPPKEY        INTEGER NOT NULL,
L_LINENUMBER     INTEGER NOT NULL,
L_QUANTITY        DECIMAL(15,2) NOT NULL,
L_EXTENDEDPRICE   DECIMAL(15,2) NOT NULL,
L_DISCOUNT        DECIMAL(15,2) NOT NULL,
L_TAX             DECIMAL(15,2) NOT NULL,
L_RETURNFLAG      CHAR(1) NOT NULL,
L_LINESSTATUS     CHAR(1) NOT NULL,
L_SHIPDATE        DATE NOT NULL,
L_COMMITDATE      DATE NOT NULL,
L_RECEIPTDATE     DATE NOT NULL,
L_SHIPINSTRUCT    CHAR(25) NOT NULL,
L_SHIPMODE        CHAR(10) NOT NULL,
L_COMMENT         VARCHAR(44) NOT NULL);

CREATE TABLE
postgres=# \d
      List of relations
 Schema |   Name   | Type  | Owner
-----+-----+-----+-----
 public | customer | table | postgres
 public | lineitem | table | postgres
 public | nation   | table | postgres
 public | orders   | table | postgres
 public | part     | table | postgres
 public | partsupp | table | postgres
 public | region   | table | postgres
 public | supplier | table | postgres
(8 rows)
```

3. 添加主键和外键

-- 1. REGION 表

```
ALTER TABLE REGION ADD PRIMARY KEY (R_REGIONKEY);
```

-- 2. NATION 表

```
ALTER TABLE NATION ADD PRIMARY KEY (N_NATIONKEY);
ALTER TABLE NATION ADD FOREIGN KEY (N_REGIONKEY) REFERENCES
REGION(R_REGIONKEY);
```

-- 3. PART 表

```
ALTER TABLE PART ADD PRIMARY KEY (P_PARTKEY);
```

-- 4. SUPPLIER 表

```
ALTER TABLE SUPPLIER ADD PRIMARY KEY (S_SUPPKEY);
ALTER TABLE SUPPLIER ADD FOREIGN KEY (S_NATIONKEY) REFERENCES
NATION(N_NATIONKEY);
```

-- 5. PARTSUPP 表

```
ALTER TABLE PARTSUPP ADD PRIMARY KEY (PS_PARTKEY, PS_SUPPKEY);
ALTER TABLE PARTSUPP ADD FOREIGN KEY (PS_PARTKEY) REFERENCES
PART(P_PARTKEY);
ALTER TABLE PARTSUPP ADD FOREIGN KEY (PS_SUPPKEY) REFERENCES
SUPPLIER(S_SUPPKEY);
```

-- 6. CUSTOMER 表

```
ALTER TABLE CUSTOMER ADD PRIMARY KEY (C_CUSTKEY);
ALTER TABLE CUSTOMER ADD FOREIGN KEY (C_NATIONKEY) REFERENCES
NATION(N_NATIONKEY);
```

-- 7. ORDERS 表

```
ALTER TABLE ORDERS ADD PRIMARY KEY (O_ORDERKEY);
ALTER TABLE ORDERS ADD FOREIGN KEY (O_CUSTKEY) REFERENCES
CUSTOMER(C_CUSTKEY);
```

-- 8. LINEITEM 表

```
ALTER TABLE LINEITEM ADD PRIMARY KEY (L_ORDERKEY, L_LINENUMBER);
ALTER TABLE LINEITEM ADD FOREIGN KEY (L_ORDERKEY) REFERENCES
ORDERS(O_ORDERKEY);
ALTER TABLE LINEITEM ADD FOREIGN KEY (L_PARTKEY, L_SUPPKEY) REFERENCES
PARTSUPP(PS_PARTKEY, PS_SUPPKEY);
```

4. 导入数据

先把数据文件末尾的 | 去掉：

```
sed -i 's/|$//' *.tbl
```

然后导入数据：

```
COPY region FROM '/tmp/data/region.tbl' WITH DELIMITER '|' NULL '';
COPY nation FROM '/tmp/data/nation.tbl' WITH DELIMITER '|' NULL '';
COPY part FROM '/tmp/data/part.tbl' WITH DELIMITER '|' NULL '';
COPY supplier FROM '/tmp/data/supplier.tbl' WITH DELIMITER '|' NULL '';
COPY customer FROM '/tmp/data/customer.tbl' WITH DELIMITER '|' NULL '';
COPY partsupp FROM '/tmp/data/partsupp.tbl' WITH DELIMITER '|' NULL '';
COPY orders FROM '/tmp/data/orders.tbl' WITH DELIMITER '|' NULL '';
COPY lineitem FROM '/tmp/data/lineitem.tbl' WITH DELIMITER '|' NULL ';
```

```
postgres=# COPY region FROM '/tmp/data/region.tbl' WITH DELIMITER '|';
COPY region FROM '/tmp/data/region.tbl' WITH DELIMITER '|';
COPY nation FROM '/tmp/data/nation.tbl' WITH DELIMITER '|';
COPY part FROM '/tmp/data/part.tbl' WITH DELIMITER '|';
COPY supplier FROM '/tmp/data/supplier.tbl' WITH DELIMITER '|';
COPY customer FROM '/tmp/data/customer.tbl' WITH DELIMITER '|';
COPY partsupp FROM '/tmp/data/partsupp.tbl' WITH DELIMITER '|';
COPY orders FROM '/tmp/data/orders.tbl' WITH DELIMITER '|';
COPY lineitem FROM '/tmp/data/lineitem.tbl' WITH DELIMITER '|';
COPY 5
COPY 25
COPY 2000
COPY 100
COPY 1500
COPY 8000
COPY 15000
COPY 60175
postres=#

```

5. 验证和查询

```
SELECT count(*) FROM customer;
SELECT count(*) FROM orders;
SELECT count(*) FROM lineitem;

-- 查询余额最高的前5个消费者
SELECT c_name, c_acctbal FROM customer ORDER BY c_acctbal DESC LIMIT 5;

-- 带 Join 的查询：查询每个国家的供应商数量
SELECT n.n_name, count(*)
FROM supplier s
JOIN nation n ON s.s_nationkey = n.n_nationkey
GROUP BY n.n_name;
```

```
postgres=# SELECT count(*) FROM customer;
count
-----
 1500
(1 row)

postgres=# SELECT count(*) FROM orders;
count
-----
 15000
(1 row)

postgres=# SELECT count(*) FROM lineitem;
count
-----
 60175
(1 row)

postgres=# SELECT c_name, c_acctbal FROM customer ORDER BY c_acctbal DESC LIMIT 5;
c_name      | c_acctbal
-----+-----
Customer#000000213 | 9987.71
Customer#000000045 | 9983.38
Customer#000001106 | 9977.62
Customer#000000200 | 9967.60
Customer#000000140 | 9963.15
(5 rows)
```

```
postgres=# SELECT n.n_name, count(*)  
FROM supplier s  
JOIN nation n ON s.s_nationkey = n.n_nationkey  
GROUP BY n.n_name;  
      n_name       | count  
-----+-----  
ETHIOPIA        |    3  
EGYPT           |    6  
IRAN            |    2  
SAUDI ARABIA   |    1  
RUSSIA          |    5  
VIETNAM         |    6  
PERU             |    4  
ALGERIA         |    3  
ARGENTINA       |    3  
JAPAN            |    4  
IRAQ             |    2  
CANADA           |    3  
BRAZIL            |    2  
INDONESIA        |    5  
FRANCE           |    2  
JORDAN           |    1  
MOROCCO          |    2  
UNITED STATES   |    8  
CHINA            |    7  
GERMANY          |    5  
ROMANIA          |    5  
INDIA             |    5  
UNITED KINGDOM  |    3  
MOZAMBIQUE       |    7  
KENYA             |    6  
(25 rows)
```

```
postgres=#
```