《数据库系统原理》实验报告（1）					
题目：DDL 语言实验					
学号	2352018	姓名	刘彦	日期	2025.04.09

实验环境：

Docker-desktop 4.40.0

oceanbase-ce 4.3.5.1

实验步骤及结果截图：

⑴创建实验所需的数据库并使用

```
create database exp1;
use exp1;
```

```
sh-4.4# obclient -uroot@sys -h127.1 -P2881;
Welcome to the OceanBase.  Commands end with ; or \g.
Your OceanBase connection id is 3221498747
Server version: OceanBase_CE 4.3.5.1 (r101000042025031818-b6d5706eb3d2c5f501c7fa646ddbf32f3dc87069) (Built Mar 18 2025 18:13:36)

Copyright (c) 2000, 2018, OceanBase and/or its affiliates. All rights reserved.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

obclient(root@sys)[(none)]> create database exp1;
Query OK, 1 row affected (0.033 sec)

obclient(root@sys)[(none)]> use exp1
Database changed
obclient(root@sys)[exp1]>
```

⑵表结构设计，根据题目要求创建下面 5 个表（电商系统相关数据表）：

①表 Customers

```
CREATE TABLE Customers (
    CustomerID INT AUTO_INCREMENT NOT NULL,
    Name VARCHAR(50) NOT NULL,
    Email VARCHAR(100) NOT NULL UNIQUE,
    Phone VARCHAR(20) NOT NULL,
    RegistrationDate DATE NOT NULL DEFAULT (date(current_timestamp)),
    PRIMARY KEY (CustomerID),
    -- 检查邮箱是否包含 @ 和 .
    CHECK (Email LIKE '%@%.%'),
    -- 检查电话（如果非空，长度为 11 且全为数字）
    CHECK (LENGTH(Phone) = 11 AND Phone NOT LIKE '%[^0-9]%')
);
```

```
obclient(root@sys)[exp1]> CREATE TABLE Customers (
    ->     CustomerID INT AUTO_INCREMENT NOT NULL,
    ->     Name VARCHAR(50) NOT NULL,
    ->     Email VARCHAR(100) NOT NULL UNIQUE,
    ->     Phone VARCHAR(20) NOT NULL,
    ->     RegistrationDate DATE NOT NULL DEFAULT (date(current_timestamp)),
    ->     PRIMARY KEY (CustomerID),
    ->     -- 检查邮箱是否包含 @ 和 .
    ->     CHECK (Email LIKE '%@%.%'),
    ->     -- 检查电话（如果非空，长度为 11 且全为数字）
    ->     CHECK (LENGTH(Phone) = 11 AND Phone NOT LIKE '%[^0-9]%')
    -> );
Query OK, 0 rows affected (0.069 sec)
```

②表 Categories

```
CREATE TABLE Categories (
```

```
    CategoryID INT AUTO_INCREMENT NOT NULL,
    CategoryName VARCHAR(50) NOT NULL UNIQUE,
    PRIMARY KEY (CategoryID)
);
```

```
obclient(root@sys)[exp1]> CREATE TABLE Categories (
    ->     CategoryID INT AUTO_INCREMENT NOT NULL,
    ->     CategoryName VARCHAR(50) NOT NULL UNIQUE,
    ->     PRIMARY KEY (CategoryID)
    -> );
Query OK, 0 rows affected (0.061 sec)
```

③表 Products

```
CREATE TABLE Products (
    ProductID INT AUTO_INCREMENT NOT NULL,
    ProductName VARCHAR(100) NOT NULL,
    Price DECIMAL(10,2) NOT NULL CHECK (Price > 0),
    StockQuantity INT NOT NULL CHECK (StockQuantity >= 0),
    CategoryID INT NOT NULL,
    PRIMARY KEY (ProductID),
    FOREIGN KEY (CategoryID) REFERENCES Categories(CategoryID)
);
```

```
obclient(root@sys)[exp1]> CREATE TABLE Products (
    ->     ProductID INT AUTO_INCREMENT NOT NULL,
    ->     ProductName VARCHAR(100) NOT NULL,
    ->     Price DECIMAL(10,2) NOT NULL CHECK (Price > 0),
    ->     StockQuantity INT NOT NULL CHECK (StockQuantity >= 0),
    ->     CategoryID INT NOT NULL,
    ->     PRIMARY KEY (ProductID),
    ->     FOREIGN KEY (CategoryID) REFERENCES Categories(CategoryID)
    -> );
Query OK, 0 rows affected (0.118 sec)
```

④表 Orders

```
CREATE TABLE Orders (
    OrderID INT AUTO_INCREMENT NOT NULL,
    CustomerID INT NOT NULL,
    OrderDate DATETIME NOT NULL DEFAULT current_timestamp,
    TotalAmount DECIMAL(10,2) NOT NULL CHECK (TotalAmount >= 0),
    PRIMARY KEY (OrderID),
    FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)
);
```

```
obclient(root@sys)[exp1]> CREATE TABLE Orders (
    ->     OrderID INT AUTO_INCREMENT NOT NULL,
    ->     CustomerID INT NOT NULL,
    ->     OrderDate DATETIME NOT NULL DEFAULT current_timestamp,
    ->     TotalAmount DECIMAL(10,2) NOT NULL CHECK (TotalAmount >= 0),
    ->     PRIMARY KEY (OrderID),
    ->     FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)
    -> );
Query OK, 0 rows affected (0.159 sec)
```

⑤表 OrdersItems

```
CREATE TABLE OrderItems (
    OrderItemID INT AUTO_INCREMENT NOT NULL,
```

```
    OrderID INT NOT NULL,
    ProductID INT NOT NULL,
    Quantity INT NOT NULL CHECK (Quantity > 0),
    Subtotal DECIMAL(10,2) NOT NULL CHECK (Subtotal > 0),
    PRIMARY KEY (OrderItemID),
    FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),
    FOREIGN KEY (ProductID) REFERENCES Products(ProductID)
);
```

```
obclient(root@sys)[exp1]> CREATE TABLE OrderItems (
    ->     OrderItemID INT AUTO_INCREMENT NOT NULL,
    ->     OrderID INT NOT NULL,
    ->     ProductID INT NOT NULL,
    ->     Quantity INT NOT NULL CHECK (Quantity > 0),
    ->     Subtotal DECIMAL(10,2) NOT NULL CHECK (Subtotal > 0),
    ->     PRIMARY KEY (OrderItemID),
    ->     FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),
    ->     FOREIGN KEY (ProductID) REFERENCES Products(ProductID)
    -> );
Query OK, 0 rows affected (0.133 sec)
```

触发器：设置 OrderItems.Subtotal = Products.Price * OrderItems.Quantity，满足字段 Subtotal 的设计要求。

```
DELIMITER //
CREATE TRIGGER before_orderitems_insert
BEFORE INSERT ON OrderItems
FOR EACH ROW
BEGIN
    DECLARE product_price DECIMAL(10,2);
    -- 获取产品价格
    SELECT Price INTO product_price
    FROM Products
    WHERE ProductID = NEW.ProductID;
    -- 设置 Subtotal = Price * Quantity
    SET NEW.Subtotal = product_price * NEW.Quantity;
END //
DELIMITER ;
```

```
obclient(root@sys)[exp1]> -- 触发器: 设置OrderItems.Subtotal = Products.Price * OrderItems.Quantity
Query OK, 0 rows affected (0.001 sec)

obclient(root@sys)[exp1]> DELIMITER //
obclient(root@sys)[exp1]>
obclient(root@sys)[exp1]> CREATE TRIGGER before_orderitems_insert
    -> BEFORE INSERT ON OrderItems
    -> FOR EACH ROW
    -> BEGIN
    ->     DECLARE product_price DECIMAL(10,2);
    ->
    ->     -- 获取产品价格
    ->     SELECT Price INTO product_price
    ->     FROM Products
    ->     WHERE ProductID = NEW.ProductID;
    ->
    ->     -- 设置 Subtotal = Price * Quantity
    ->     SET NEW.Subtotal = product_price * NEW.Quantity;
    -> END //
Query OK, 0 rows affected (0.110 sec)

obclient(root@sys)[exp1]>
obclient(root@sys)[exp1]> DELIMITER ;
```

建立完所有表后进行 show tables 展示所有表：

```
obclient(root@sys)[exp1]> show tables;
+----------------+
| Tables_in_exp1 |
+----------------+
| Categories     |
| Customers      |
| OrderItems     |
| Orders         |
| Products       |
+----------------+
5 rows in set (0.008 sec)
```

⑶表结构查询

desc Customers;

desc Products;

desc Categories;

desc Order;

desc OrderItems;

```
obclient(root@sys)[exp1]> desc Customers;
+------------------+-------------+------+------+-------------+----------------+
| Field            | Type        | Null | Key  | Default     | Extra          |
+------------------+-------------+------+------+-------------+----------------+
| CustomerID       | int(11)     | NO   | PRI  | NULL        | auto_increment |
| Name             | varchar(50) | NO   |      | NULL        |                |
| Email            | varchar(100)| NO   | UNI  | NULL        |                |
| Phone            | varchar(20) | NO   |      | NULL        |                |
| RegistrationDate | date        | NO   |      | date(now()) |                |
+------------------+-------------+------+------+-------------+----------------+
5 rows in set (0.022 sec)
```

```
obclient(root@sys)[exp1]> desc Products;
+---------------+---------------+------+------+---------+----------------+
| Field         | Type          | Null | Key  | Default | Extra          |
+---------------+---------------+------+------+---------+----------------+
| ProductID     | int(11)       | NO   | PRI  | NULL    | auto_increment |
| ProductName   | varchar(100)  | NO   |      | NULL    |                |
| Price         | decimal(10,2) | NO   |      | NULL    |                |
| StockQuantity | int(11)       | NO   |      | NULL    |                |
| CategoryID    | int(11)       | NO   |      | NULL    |                |
+---------------+---------------+------+------+---------+----------------+
5 rows in set (0.011 sec)
```

```
obclient(root@sys)[exp1]> desc Categories;
+--------------+-------------+------+------+---------+----------------+
| Field        | Type        | Null | Key  | Default | Extra          |
+--------------+-------------+------+------+---------+----------------+
| CategoryID   | int(11)     | NO   | PRI  | NULL    | auto_increment |
| CategoryName | varchar(50) | NO   | UNI  | NULL    |                |
+--------------+-------------+------+------+---------+----------------+
2 rows in set (0.016 sec)
```

```
obclient(root@sys)[exp1]> desc Orders;
+-------------+---------------+------+------+-------------------+----------------+
| Field       | Type          | Null | Key  | Default           | Extra          |
+-------------+---------------+------+------+-------------------+----------------+
| OrderID     | int(11)       | NO   | PRI  | NULL              | auto_increment |
| CustomerID  | int(11)       | NO   |      | NULL              |                |
| OrderDate   | datetime      | NO   |      | CURRENT_TIMESTAMP |                |
| TotalAmount | decimal(10,2) | NO   |      | NULL              |                |
+-------------+---------------+------+------+-------------------+----------------+
4 rows in set (0.010 sec)
```

```
obclient(root@sys)[exp1]> desc OrderItems;
+-------------+---------------+------+------+---------+----------------+
| Field       | Type          | Null | Key  | Default | Extra          |
+-------------+---------------+------+------+---------+----------------+
| OrderItemID | int(11)       | NO   | PRI  | NULL    | auto_increment |
| OrderID     | int(11)       | NO   |      | NULL    |                |
| ProductID   | int(11)       | NO   |      | NULL    |                |
| Quantity    | int(11)       | NO   |      | NULL    |                |
| Subtotal    | decimal(10,2) | NO   |      | NULL    |                |
+-------------+---------------+------+------+---------+----------------+
5 rows in set (0.012 sec)
```

⑷查询建立的约束

SELECT table_name, constraint_name, constraint_type

FROM information_schema.TABLE_CONSTRAINTS

WHERE table_name = 'Customers';

```
obclient(root@sys)[exp1]> SELECT table_name, constraint_name, constraint_type
    -> FROM information_schema.TABLE_CONSTRAINTS
    -> WHERE table_name = 'Customers';
+------------+-------------------------------------+-----------------+
| table_name | constraint_name                     | constraint_type |
+------------+-------------------------------------+-----------------+
| Customers  | PRIMARY                             | PRIMARY KEY     |
| Customers  | Email                               | UNIQUE          |
| Customers  | Customers_OBCHECK_1744206782705287 | CHECK           |
| Customers  | Customers_OBCHECK_1744206782705349 | CHECK           |
+------------+-------------------------------------+-----------------+
4 rows in set (0.081 sec)
```

SELECT table_name, constraint_name, constraint_type

FROM information_schema.TABLE_CONSTRAINTS

WHERE table_name = 'Products';

```
obclient(root@sys)[exp1]> SELECT table_name, constraint_name, constraint_type
    -> FROM information_schema.TABLE_CONSTRAINTS
    -> WHERE table_name = 'Products';
+------------+-------------------------------------+-----------------+
| table_name | constraint_name                     | constraint_type |
+------------+-------------------------------------+-----------------+
| Products   | PRIMARY                             | PRIMARY KEY     |
| Products   | Products_OBCHECK_1744208007839641 | CHECK           |
| Products   | Products_OBCHECK_1744208007839735 | CHECK           |
| Products   | Products_OBFK_1744208007840144    | FOREIGN KEY     |
+------------+-------------------------------------+-----------------+
4 rows in set (0.005 sec)
```

SELECT table_name, constraint_name, constraint_type

FROM information_schema.TABLE_CONSTRAINTS

WHERE table_name = 'Categories';

```
obclient(root@sys)[exp1]> SELECT table_name, constraint_name, constraint_type
    -> FROM information_schema.TABLE_CONSTRAINTS
    -> WHERE table_name = 'Categories';
+------------+-----------------+-----------------+
| table_name | constraint_name | constraint_type |
+------------+-----------------+-----------------+
| Categories | PRIMARY         | PRIMARY KEY     |
| Categories | CategoryName    | UNIQUE          |
+------------+-----------------+-----------------+
2 rows in set (0.004 sec)
```

```
SELECT table_name, constraint_name, constraint_type
FROM information_schema.TABLE_CONSTRAINTS
WHERE table_name = 'Orders';
```

```
obclient(root@sys)[exp1]> SELECT table_name, constraint_name, constraint_type
    -> FROM information_schema.TABLE_CONSTRAINTS
    -> WHERE table_name = 'Orders';
+------------+-----------------------------------+-----------------+
| table_name | constraint_name                   | constraint_type |
+------------+-----------------------------------+-----------------+
| Orders     | PRIMARY                           | PRIMARY KEY     |
| Orders     | Orders_OBCHECK_1744213010874996 | CHECK           |
| Orders     | Orders_OBFK_1744213010875157    | FOREIGN KEY     |
+------------+-----------------------------------+-----------------+
3 rows in set (0.003 sec)
```

```
SELECT table_name, constraint_name, constraint_type
FROM information_schema.TABLE_CONSTRAINTS
WHERE table_name = 'OrderItems';
```

```
obclient(root@sys)[exp1]> SELECT table_name, constraint_name, constraint_type
    -> FROM information_schema.TABLE_CONSTRAINTS
    -> WHERE table_name = 'OrderItems';
+------------+-------------------------------------+-----------------+
| table_name | constraint_name                     | constraint_type |
+------------+-------------------------------------+-----------------+
| OrderItems | PRIMARY                             | PRIMARY KEY     |
| OrderItems | OrderItems_OBCHECK_1744214062283249 | CHECK           |
| OrderItems | OrderItems_OBCHECK_1744214062283507 | CHECK           |
| OrderItems | OrderItems_OBFK_1744214062292714    | FOREIGN KEY     |
| OrderItems | OrderItems_OBFK_1744214062283745    | FOREIGN KEY     |
+------------+-------------------------------------+-----------------+
5 rows in set (0.004 sec)
```

⑸表结构修改与操作

①在 Customers 表中添加一个新字段 LastLoginDate，类型为 DATETIME，当客户每次登录时，该字段自动更新为当前时间。需创建触发器实现此功能。

第一步先进行表的修改，添加一个新字段：

```
ALTER TABLE Customers
ADD LastLoginDate DATETIME NOT NULL;
```

```
obclient(root@sys)[exp1]> ALTER TABLE Customers
    -> ADD LastLoginDate DATETIME NOT NULL;
Query OK, 0 rows affected (0.118 sec)

obclient(root@sys)[exp1]> desc Customers;
+------------------+-------------+------+-----+-------------+----------------+
| Field            | Type        | Null | Key | Default     | Extra          |
+------------------+-------------+------+-----+-------------+----------------+
| CustomerID       | int(11)     | NO   | PRI | NULL        | auto_increment |
| Name             | char(15)    | NO   |     | NULL        |                |
| Email            | varchar(100)| NO   | UNI | NULL        |                |
| Phone            | varchar(20) | NO   |     | NULL        |                |
| RegistrationDate | date        | NO   |     | date(now()) |                |
| LastLoginDate    | datetime    | NO   |     | NULL        |                |
+------------------+-------------+------+-----+-------------+----------------+
6 rows in set (0.022 sec)
```

第一个触发器会在向 Customers 表插入新记录（比如新用户注册）之前被触发。在插入新记录时，自动将新记录的 LastLoginDate 字段设置为当前时间：

```
DELIMITER //
CREATE TRIGGER before_customers_insert
BEFORE INSERT ON Customers
FOR EACH ROW
```

```
BEGIN
    -- 新用户注册时，设置 LastLoginDate 为当前时间
    SET NEW.LastLoginDate = CURRENT_TIMESTAMP;
END //
DELIMITER ;
```

第二个触发器会在 Customers 表的记录被更新之前触发。在更新记录，即用户登录时，自动将 LastLoginDate 字段设置为当前时间：

```
DELIMITER //
CREATE TRIGGER before_customers_update
BEFORE UPDATE ON Customers
FOR EACH ROW
BEGIN
    -- 在用户登录（更新）时，自动设置 LastLoginDate 为当前时间
    SET NEW.LastLoginDate = current_timestamp;
END //
DELIMITER ;
```

```
obclient(root@sys)[exp1]> DELIMITER //
obclient(root@sys)[exp1]> CREATE TRIGGER before_customers_insert
    -> BEFORE INSERT ON Customers
    -> FOR EACH ROW
    -> BEGIN
    ->     -- 新用户注册时，设置 LastLoginDate 为当前时间
    ->     SET NEW.LastLoginDate = CURRENT_TIMESTAMP;
    -> END //
Query OK, 0 rows affected (0.120 sec)

obclient(root@sys)[exp1]> DELIMITER ;
obclient(root@sys)[exp1]> DELIMITER //
obclient(root@sys)[exp1]> CREATE TRIGGER before_customers_update
    -> BEFORE UPDATE ON Customers
    -> FOR EACH ROW
    -> BEGIN
    ->     -- 在用户登录（更新）时，自动设置 LastLoginDate 为当前时间
    ->     SET NEW.LastLoginDate = current_timestamp;
    -> END //
Query OK, 0 rows affected (0.129 sec)

obclient(root@sys)[exp1]> DELIMITER ;
```

②将 Customers 中的 name 由 varchar 改为 char(15)。

对于 Name 列的值，如果它的长度（字符数）超过 15 个字符，就只保留前 15 个字符，为更改其数据类型做准备：

```
UPDATE Customers
SET Name = LEFT(Name, 15)
WHERE LENGTH(Name) > 15;
```

然后将 Name 列的数据类型改为 CHAR(15)：

```
ALTER TABLE Customers
MODIFY COLUMN Name CHAR(15) NOT NULL;
```

```
obclient(root@sys)[exp1]> UPDATE Customers
    -> SET Name = LEFT(Name, 15)
    -> WHERE LENGTH(Name) > 15;
Query OK, 0 rows affected (0.028 sec)
Rows matched: 0  Changed: 0  Warnings: 0

obclient(root@sys)[exp1]> ALTER TABLE Customers
    -> MODIFY COLUMN Name CHAR(15) NOT NULL;
Query OK, 0 rows affected (1.189 sec)

obclient(root@sys)[exp1]> desc Customers;
+------------------+--------------+------+------+-------------+----------------+
| Field            | Type         | Null | Key  | Default     | Extra          |
+------------------+--------------+------+------+-------------+----------------+
| CustomerID       | int(11)      | NO   | PRI  | NULL        | auto_increment |
| Name             | char(15)     | NO   |      | NULL        |                |
| Email            | varchar(100) | NO   | UNI  | NULL        |                |
| Phone            | varchar(20)  | NO   |      | NULL        |                |
| RegistrationDate | date         | NO   |      | date(now()) |                |
| LastLoginDate    | datetime     | NO   |      | NULL        |                |
+------------------+--------------+------+------+-------------+----------------+
6 rows in set (0.028 sec)
```

③修改 Products 表，添加一个 IsFeatured 字段，类型为 BOOLEAN，默认值为 FALSE。同时，创建一个索引，用于快速查询特色商品（IsFeatured 为 TRUE 的商品）。

向 Products 表添加一个新字段 IsFeatured，数据类型为 BOOLEAN：

```
ALTER TABLE Products
ADD IsFeatured BOOLEAN NOT NULL DEFAULT FALSE;
```

```
obclient(root@sys)[exp1]> ALTER TABLE Products
    -> ADD IsFeatured BOOLEAN NOT NULL DEFAULT FALSE;
Query OK, 0 rows affected (0.117 sec)

obclient(root@sys)[exp1]> desc Products;
+---------------+---------------+------+------+---------+----------------+
| Field         | Type          | Null | Key  | Default | Extra          |
+---------------+---------------+------+------+---------+----------------+
| ProductID     | int(11)       | NO   | PRI  | NULL    | auto_increment |
| ProductName   | varchar(100)  | NO   |      | NULL    |                |
| Price         | decimal(10,2) | NO   |      | NULL    |                |
| StockQuantity | int(11)       | NO   |      | NULL    |                |
| CategoryID    | int(11)       | NO   |      | NULL    |                |
| IsFeatured    | tinyint(1)    | NO   |      | 0       |                |
+---------------+---------------+------+------+---------+----------------+
6 rows in set (0.003 sec)
```

在 Products 表的 IsFeatured 列上创建一个索引，找到表中 IsFeatured 列为 TRUE 所有数据，并显示该索引：

```
CREATE INDEX idx_isfeatured ON Products(IsFeatured);
SHOW INDEX FROM Products;
```

```
obclient(root@sys)[exp1]> CREATE INDEX idx_isfeatured ON Products(IsFeatured);
Query OK, 0 rows affected (0.324 sec)

obclient(root@sys)[exp1]> SHOW INDEX FROM Products;
+----------+------------+---------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Table    | Non_unique | Key_name      | Seq_in_index | Column_name | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment | Index_comment | Visible | Expressio
n |
+----------+------------+---------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Products |          0 | PRIMARY       |            1 | ProductID   | A         |        NULL | NULL     | NULL   |      | BTREE      | available |               | YES     | NULL
  |
| Products |          1 | idx_isfeatured |            1 | IsFeatured  | A         |        NULL | NULL     | NULL   |      | BTREE      | available |               | YES     | NULL
  |
+----------+------------+---------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
2 rows in set (0.022 sec)
```

④删除 Categories 表中所有没有关联商品的分类记录，同时要保证删除操作符合数据库的参照完整性。需创建一个存储过程实现此功能。

存储过程 DeleteOrphanedCategories 的目的是清理 Categories 表中没有关联商品的记录。步骤是：定义变量 deleted_rows 记录删除行数；设置错误处理器，确保异常时回滚事务并抛出错误；开启事务，保证操作的原子性；使用 NOT EXISTS 子查询找出没有商品的分类并删除；记录删除的行数；提交事务，确认更改；返回删除的行数信息。

```
DELIMITER //
```

```
CREATE PROCEDURE DeleteOrphanedCategories()
BEGIN
    -- 声明变量用于错误处理和记录删除行数
    DECLARE deleted_rows INT DEFAULT 0;
    DECLARE EXIT HANDLER FOR SQLEXCEPTION
    BEGIN
        -- 错误发生时回滚事务并返回错误信息
        ROLLBACK;
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'An error occurred while deleting orphaned categories';
    END;
    -- 开启事务，确保操作原子性
    START TRANSACTION;
    -- 删除没有关联商品的分类
    DELETE FROM Categories
    WHERE NOT EXISTS (
        SELECT 1
        FROM Products
        WHERE Products.CategoryID = Categories.CategoryID
    );
    -- 获取删除的行数
    SET deleted_rows = ROW_COUNT();
    -- 提交事务
    COMMIT;
    -- 返回删除的行数，用于验证
    SELECT CONCAT('Deleted ', deleted_rows, ' orphaned categories') AS Result;
END //
DELIMITER ;
```

```
obclient(root@sys)[exp1]> DELIMITER //
obclient(root@sys)[exp1]> CREATE PROCEDURE DeleteOrphanedCategories()
    -> BEGIN
    ->     -- 声明变量用于错误处理和记录删除行数
    ->     DECLARE deleted_rows INT DEFAULT 0;
    ->     DECLARE EXIT HANDLER FOR SQLEXCEPTION
    ->     BEGIN
    ->         -- 错误发生时回滚事务并返回错误信息
    ->         ROLLBACK;
    ->         SIGNAL SQLSTATE '45000'
    ->         SET MESSAGE_TEXT = 'An error occurred while deleting orphaned categories';
    ->     END;
    ->
    ->     -- 开启事务，确保操作原子性
    ->     START TRANSACTION;
    ->
    ->     -- 删除没有关联商品的分类
    ->     DELETE FROM Categories
    ->     WHERE NOT EXISTS (
    ->         SELECT 1
    ->         FROM Products
    ->         WHERE Products.CategoryID = Categories.CategoryID
    ->     );
    ->
    ->     -- 获取删除的行数
    ->     SET deleted_rows = ROW_COUNT();
    ->
    ->     -- 提交事务
    ->     COMMIT;
    ->
    ->     -- 返回删除的行数，用于验证
    ->     SELECT CONCAT('Deleted ', deleted_rows, ' orphaned categories') AS Result;
    -> END //
Query OK, 0 rows affected (0.119 sec)

obclient(root@sys)[exp1]> DELIMITER ;
```

执行刚刚定义的存储过程，触发删除操作并返回结果：

CALL DeleteOrphanedCategories();

```
obclient(root@sys)[exp1]> CALL DeleteOrphanedCategories();
+------------------------------+
| Result                       |
+------------------------------+
| Deleted 0 orphaned categories |
+------------------------------+
1 row in set (0.178 sec)

Query OK, 0 rows affected (0.178 sec)
```

出现的问题：

①DATE 型变量设置默认值会报错

错误发生在 RegistrationDate DATE NOT NULL DEFAULT (date(current_timestamp))。该版本 oceanbase-ce 不支持直接在 DEFAULT 中使用 date(current_timestamp)这样的默认值表达式。

```
obclient(root@sys)[exp1]> CREATE TABLE Customers (
    ->     CustomerID INT AUTO_INCREMENT PRIMARY KEY,
    ->     Name VARCHAR(50) NOT NULL,
    ->     Email VARCHAR(100) NOT NULL UNIQUE,
    ->     Phone VARCHAR(20) NULL,
    ->     RegistrationDate DATE NOT NULL DEFAULT (date(current_timestamp)),
    ->     -- 检查邮箱是否包含 @ 和 .
    ->     CHECK (Email LIKE '%@%.%'),
    ->     -- 检查电话 (如果非空, 长度为 11 且全为数字)
    ->     CHECK (Phone IS NULL OR (LENGTH(Phone) = 11 AND Phone NOT LIKE '%[^0-9]%'))
    -> );
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your OceanBase version for the right syntax to use near '(date(current_timestamp)),
    -- ???????? @ ? .
    CHECK (Em' at line 6
[172.17.0.2:2882] [2025-04-09 13:34:30.051382] [YB42AC110002-000632586196BD05-0-0]
```

②列定义发生语法错误

错误发生在：

```
CREATE TABLE Categories (
    CategoryID INT AUTO_INCREMENT NOT NULL,
    CategoryName VARCHAR(50) NOT NULL UNIQUE,
    PRIMARY KEY (CategoryID),
);
```

试图创建 Categories 表，但由于语法错误导致失败。

```
obclient(root@sys)[exp1]> CREATE TABLE Categories (
    ->     CategoryID INT AUTO_INCREMENT NOT NULL,
    ->     CategoryName VARCHAR(50) NOT NULL UNIQUE,
    ->     PRIMARY KEY (CategoryID),
    -> );
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your OceanBase version for the right syntax to use near ')' at line 5
[172.17.0.2:2882] [2025-04-09 13:58:22.136977] [YB42AC110002-0006325896280601-0-0]
```

③查询语句约束信息错误

SQL 语句试图从 information_schema.TABLE_CONSTRAINTS 表中查询 Customers 表的约束信息，但数据库返回了错误信息，错误提示表明，在 WHERE 子句中，Customers 被当作列名处理，但实际上它是一个表名，导致查询失败。

```
obclient(root@sys)[exp1]> SELECT table_name, constraint_name, constraint_type
    -> FROM information_schema.TABLE_CONSTRAINTS
    -> WHERE table_name = Customers;
ERROR 1054 (42S22): Unknown column 'Customers' in 'where clause'
[172.17.0.2:2882] [2025-04-09 16:24:29.901536] [YB42AC110002-0006325898580603-0-0]
```

④创建索引失败

SQL 语句试图在 Products 表上创建名为 idx_isfeatured 的索引，但数据库返回了错误信息，错误发生在第 1 行，提示语法问题，具体是 Products 的使用不正确。

```
obclient(root@sys)[exp1]> CREATE INDEX idx_isfeatured ON Products;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your OceanBase version for the right syntax to use near 'Products' at line 1
[172.17.0.2:2882] [2025-04-10 10:08:05.096276] [YB42AC110002-00063269BBFDE5A9-0-0]
```

解决方案：

①DATE 型变量设置默认值会报错问题的解决

按照要求对 oceanbase-ce 进行升级，升到 4.3.5.1 后成功运行。

②列定义发生语法错误问题的解决

经检查发现，"PRIMARY KEY (CategoryID),"后面的逗号是多余的，正确的语句应该直接以)结束表定义。在 CREATE TABLE 语句中，列定义或约束列表的最后一个元素后不能有逗号。删去逗号后就可以成功创建。

③查询语句约束信息错误问题的解决

WHERE 子句试图筛选表名为 Customers 的记录。问题在于 Customers 没有用单引号或双引号括起来。在 SQL 中，字符串值（包括表名）需要用引号括起来，否则数据库会将其视为列名或标识符。这里，Customers 被错误地解析为列名，而 TABLE_CONSTRAINTS 表中没有名为 Customers 的列，因此报错。将 Customers 加上引号后结果就正确了。

④创建索引失败问题的解决

具体问题是语句缺少索引所基于的列名。在 SQL 中，CREATE INDEX 语句必须明确指定索引要应用于表的哪些列，例如 ON Products(IsFeatured)，改成这样后就可以成功创建。