《数据库系统原理》实验报告（4）					
题目：SQL 综合实验					
学号	2352018	姓名	刘彦	日期	2025.4.30

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建立数据库，在数据库中建表，表内字段的类型可以自行定义（合理即可），注意建表时不要忽略各表的主键约束和表间的外键约束

```
CREATE TABLE Movie (
    Movie_no VARCHAR(10),
    Movie_name VARCHAR(50),
    Director VARCHAR(30),
    Rating DECIMAL(3,1),
    End_date DATETIME,
    PRIMARY KEY (Movie_no)
);
```

```
obclient(root@sys)[exp4]> CREATE TABLE Viewer (
    ->     Viewer_no VARCHAR(10),
    ->     Viewer_name VARCHAR(30),
    ->     Age INT,
    ->     PRIMARY KEY (Viewer_no)
    -> );
Query OK, 0 rows affected (0.083 sec)

obclient(root@sys)[exp4]> desc Viewer;
+-------------+-------------+------+------+---------+-------+
| Field       | Type        | Null | Key  | Default | Extra |
+-------------+-------------+------+------+---------+-------+
| Viewer_no   | varchar(10) | NO   | PRI  | NULL    |       |
| Viewer_name | varchar(30) | YES  |      | NULL    |       |
| Age         | int(11)     | YES  |      | NULL    |       |
+-------------+-------------+------+------+---------+-------+
3 rows in set (0.014 sec)
```

```
CREATE TABLE Viewer (
    Viewer_no VARCHAR(10),
    Viewer_name VARCHAR(30),
    Age INT,
    PRIMARY KEY (Viewer_no)
);
```

```
obclient(root@sys)[exp4]> CREATE TABLE Movie (
    ->     Movie_no VARCHAR(10),
    ->     Movie_name VARCHAR(50),
    ->     Director VARCHAR(30),
    ->     Rating DECIMAL(3,1),
    ->     End_date DATETIME,
    ->     PRIMARY KEY (Movie_no)
    -> );
Query OK, 0 rows affected (0.098 sec)

obclient(root@sys)[exp4]> desc Movie;
+------------+--------------+------+------+---------+-------+
| Field      | Type         | Null | Key  | Default | Extra |
+------------+--------------+------+------+---------+-------+
| Movie_no   | varchar(10)  | NO   | PRI  | NULL    |       |
| Movie_name | varchar(50)  | YES  |      | NULL    |       |
| Director   | varchar(30)  | YES  |      | NULL    |       |
| Rating     | decimal(3,1) | YES  |      | NULL    |       |
| End_date   | datetime     | YES  |      | NULL    |       |
+------------+--------------+------+------+---------+-------+
5 rows in set (0.014 sec)
```

```
CREATE TABLE Watch (
    S_no VARCHAR(10),
    Viewer_no VARCHAR(10),
    Movie_no VARCHAR(10),
    Watch_date DATETIME,
    PRIMARY KEY (S_no, Viewer_no, Movie_no),
    FOREIGN KEY (Viewer_no) REFERENCES Viewer(Viewer_no) ON DELETE CASCADE,
    FOREIGN KEY (Movie_no) REFERENCES Movie(Movie_no) ON DELETE CASCADE
);
```

```
obclient(root@sys)[exp4]> CREATE TABLE Watch (
    ->     S_no VARCHAR(10),
    ->     Viewer_no VARCHAR(10),
    ->     Movie_no VARCHAR(10),
    ->     Watch_date DATETIME,
    ->     PRIMARY KEY (S_no, Viewer_no, Movie_no),
    ->     FOREIGN KEY (Viewer_no) REFERENCES Viewer(Viewer_no) ON DELETE CASCADE,
    ->     FOREIGN KEY (Movie_no) REFERENCES Movie(Movie_no) ON DELETE CASCADE
    -> );
Query OK, 0 rows affected (0.115 sec)

obclient(root@sys)[exp4]> desc Watch;
+------------+-------------+------+------+---------+-------+
| Field      | Type        | Null | Key  | Default | Extra |
+------------+-------------+------+------+---------+-------+
| S_no       | varchar(10) | NO   | PRI  | NULL    |       |
| Viewer_no  | varchar(10) | NO   | PRI  | NULL    |       |
| Movie_no   | varchar(10) | NO   | PRI  | NULL    |       |
| Watch_date | datetime    | YES  |      | NULL    |       |
+------------+-------------+------+------+---------+-------+
4 rows in set (0.015 sec)
```

⑵插入样例数据

```
insert into Movie values
('M001', '星际穿越', '克里斯托弗·诺兰', 9.3, '2024-05-01'),
('M002', '泰坦尼克号', '詹姆斯·卡梅隆', 9.1, '2024-04-10'),
('M003', '盗梦空间', '克里斯托弗·诺兰', 8.8, '2024-04-20'),
('M004', '科幻冒险之旅', '张三', 7.5, '2024-04-18'),
('M005', '爱情故事', '李四', 7.0, '2024-04-25');
insert into Viewer values
('V001', '李明', 25),
('V002', '王红', 30),
```

('V003', '张磊', 22),
('V004', '赵颖', 28),
('V005', '孙阳', 35);
insert into Watch values
('1', 'V001', 'M001', '2024-03-15'),
('2', 'V001', 'M001', '2024-03-20'),
('2', 'V001', 'M002', '2024-03-20'),
('3', 'V002', 'M002', '2024-03-25'),
('1', 'V002', 'M003', '2024-04-01'),
('2', 'V003', 'M001', '2024-04-05'),
('2', 'V004', 'M002', '2024-04-12'),
('1', 'V005', 'M003', '2024-04-14');

```
obclient(root@sys)[exp4]> insert into Movie values
    -> ('M001', '星际穿越', '克里斯托弗·诺兰', 9.3, '2024-05-01'),
    -> ('M002', '泰坦尼克号', '詹姆斯·卡梅隆', 9.1, '2024-04-10'),
    -> ('M003', '盗梦空间', '克里斯托弗·诺兰', 8.8, '2024-04-20'),
    -> ('M004', '科幻冒险之旅', '张三', 7.5, '2024-04-18'),
    -> ('M005', '爱情故事', '李四', 7.0, '2024-04-25');
Query OK, 5 rows affected (0.016 sec)
Records: 5  Duplicates: 0  Warnings: 0

obclient(root@sys)[exp4]> insert into Viewer values
    -> ('V001', '李明', 25),
    -> ('V002', '王红', 30),
    -> ('V003', '张磊', 22),
    -> ('V004', '赵颖', 28),
    -> ('V005', '孙阳', 35);
Query OK, 5 rows affected (0.018 sec)
Records: 5  Duplicates: 0  Warnings: 0

obclient(root@sys)[exp4]> insert into Watch values
    -> ('1', 'V001', 'M001', '2024-03-15'),
    -> ('2', 'V001', 'M001', '2024-03-20'),
    -> ('2', 'V001', 'M002', '2024-03-20'),
    -> ('3', 'V002', 'M002', '2024-03-25'),
    -> ('1', 'V002', 'M003', '2024-04-01'),
    -> ('2', 'V003', 'M001', '2024-04-05'),
    -> ('2', 'V004', 'M002', '2024-04-12'),
    -> ('1', 'V005', 'M003', '2024-04-14');
Query OK, 8 rows affected (0.006 sec)
Records: 8  Duplicates: 0  Warnings: 0
```

⑶查询电影名称中包含“科幻”的电影信息，输出所有信息（包括电影名称、电影编号、导演、评分、电影停映日期），并按照评分降序排列

SELECT Movie_name, Movie_no, Director, Rating, End_date
FROM Movie
WHERE Movie_name LIKE '%科幻%'
ORDER BY Rating DESC;

```
obclient(root@sys)[exp4]> SELECT Movie_name, Movie_no, Director, Rating, End_date
    -> FROM Movie
    -> WHERE Movie_name LIKE '%科幻%'
    -> ORDER BY Rating DESC;
+--------------------+----------+----------+--------+---------------------+
| Movie_name         | Movie_no | Director | Rating | End_date            |
+--------------------+----------+----------+--------+---------------------+
| 科幻冒险之旅 | M004     | 张三   |    7.5 | 2024-04-18 00:00:00 |
+--------------------+----------+----------+--------+---------------------+
1 row in set (0.004 sec)
```

⑷查询观看了电影名为“泰坦尼克号”的观众信息，输出该观众的编号、姓名和年龄，并按照观众编号升序排列

```
SELECT DISTINCT v.Viewer_no, v.Viewer_name, v.Age
FROM Viewer v
JOIN Watch w ON v.Viewer_no = w.Viewer_no
JOIN Movie m ON w.Movie_no = m.Movie_no
WHERE m.Movie_name = '泰坦尼克号'
ORDER BY v.Viewer_no ASC;
```

```
obclient(root@sys)[exp4]> SELECT DISTINCT v.Viewer_no, v.Viewer_name, v.Age
    -> FROM Viewer v
    -> JOIN Watch w ON v.Viewer_no = w.Viewer_no
    -> JOIN Movie m ON w.Movie_no = m.Movie_no
    -> WHERE m.Movie_name = '泰坦尼克号'
    -> ORDER BY v.Viewer_no ASC;
+-----------+-------------+------+
| Viewer_no | Viewer_name | Age  |
+-----------+-------------+------+
| V001      | 李明      |   25 |
| V002      | 王红      |   30 |
| V004      | 赵颖      |   28 |
+-----------+-------------+------+
3 rows in set (0.020 sec)
```

⑸统计每个观众的观影信息，输出每个观众的编号、观看的电影名称和观看日期

```
SELECT v.Viewer_no, m.Movie_name, w.Watch_date
FROM Viewer v
JOIN Watch w ON v.Viewer_no = w.Viewer_no
JOIN Movie m ON w.Movie_no = m.Movie_no
ORDER BY v.Viewer_no, w.Watch_date;
```

```
obclient(root@sys)[exp4]> SELECT v.Viewer_no, m.Movie_name, w.Watch_date
    -> FROM Viewer v
    -> JOIN Watch w ON v.Viewer_no = w.Viewer_no
    -> JOIN Movie m ON w.Movie_no = m.Movie_no
    -> ORDER BY v.Viewer_no, w.Watch_date;
+-----------+-----------------+---------------------+
| Viewer_no | Movie_name      | Watch_date          |
+-----------+-----------------+---------------------+
| V001      | 星际穿越        | 2024-03-15 00:00:00 |
| V001      | 星际穿越        | 2024-03-20 00:00:00 |
| V001      | 泰坦尼克号      | 2024-03-20 00:00:00 |
| V002      | 泰坦尼克号      | 2024-03-25 00:00:00 |
| V002      | 盗梦空间        | 2024-04-01 00:00:00 |
| V003      | 星际穿越        | 2024-04-05 00:00:00 |
| V004      | 泰坦尼克号      | 2024-04-12 00:00:00 |
| V005      | 盗梦空间        | 2024-04-14 00:00:00 |
+-----------+-----------------+---------------------+
8 rows in set (0.003 sec)
```

⑹查询所有已停映电影的信息，输出观众编号、姓名、电影名称和观看日期，并按观看日期降序排列

P.S.已停映电影指的是“现实日期”大于电影停映日期字段的电影，“现实日期”以 4 月 15 日为例。

```
SELECT v.Viewer_no, v.Viewer_name, m.Movie_name, w.Watch_date
FROM Viewer v
JOIN Watch w ON v.Viewer_no = w.Viewer_no
JOIN Movie m ON w.Movie_no = m.Movie_no
WHERE m.End_date < '2025-04-15'
ORDER BY w.Watch_date DESC;
```

```
obclient(root@sys)[exp4]> SELECT v.Viewer_no, v.Viewer_name, m.Movie_name, w.Watch_date
    -> FROM Viewer v
    -> JOIN Watch w ON v.Viewer_no = w.Viewer_no
    -> JOIN Movie m ON w.Movie_no = m.Movie_no
    -> WHERE m.End_date < '2025-04-15'
    -> ORDER BY w.Watch_date DESC;
+-----------+-------------+-----------------+---------------------+
| Viewer_no | Viewer_name | Movie_name      | Watch_date          |
+-----------+-------------+-----------------+---------------------+
| V005      | 孙阳        | 盗梦空间        | 2024-04-14 00:00:00 |
| V004      | 赵颖        | 泰坦尼克号      | 2024-04-12 00:00:00 |
| V003      | 张磊        | 星际穿越        | 2024-04-05 00:00:00 |
| V002      | 王红        | 盗梦空间        | 2024-04-01 00:00:00 |
| V002      | 王红        | 泰坦尼克号      | 2024-03-25 00:00:00 |
| V001      | 李明        | 星际穿越        | 2024-03-20 00:00:00 |
| V001      | 李明        | 泰坦尼克号      | 2024-03-20 00:00:00 |
| V001      | 李明        | 星际穿越        | 2024-03-15 00:00:00 |
+-----------+-------------+-----------------+---------------------+
8 rows in set (0.009 sec)
```

⑺查询观看了“星际穿越”但没有观看“盗梦空间”的观众信息，输出这些观众的编号，并按照编号升序排列。

```
SELECT DISTINCT w1.Viewer_no
FROM Watch w1
JOIN Movie m1 ON w1.Movie_no = m1.Movie_no
LEFT JOIN (
    SELECT w2.Viewer_no
    FROM Watch w2
    JOIN Movie m2 ON w2.Movie_no = m2.Movie_no
```

```
    WHERE m2.Movie_name = '盗梦空间'
) w2 ON w1.Viewer_no = w2.Viewer_no
WHERE m1.Movie_name = '星际穿越'
AND w2.Viewer_no IS NULL
ORDER BY w1.Viewer_no ASC;
```

```
obclient(root@sys)[exp4]> SELECT DISTINCT w1.Viewer_no
    -> FROM Watch w1
    -> JOIN Movie m1 ON w1.Movie_no = m1.Movie_no
    -> LEFT JOIN (
    ->     SELECT w2.Viewer_no
    ->     FROM Watch w2
    ->     JOIN Movie m2 ON w2.Movie_no = m2.Movie_no
    ->     WHERE m2.Movie_name = '盗梦空间'
    -> ) w2 ON w1.Viewer_no = w2.Viewer_no
    -> WHERE m1.Movie_name = '星际穿越'
    -> AND w2.Viewer_no IS NULL
    -> ORDER BY w1.Viewer_no ASC;
+-----------+
| Viewer_no |
+-----------+
| V001      |
| V003      |
+-----------+
2 rows in set (0.011 sec)
```

⑻创建一个过程，使之能够实现如下功能

修改观影表，增加字段”重复观看状态”（字段名为“Repeat_state”），字段含义为表示某观众是否多次观看某电影；

并根据表中已有数据为该字段赋值（所赋的值与表定义时的数据类型保持一致即可，比如可以定义多次观看某电影的“重复观看状态”为 True，只看过一次某电影的“重复观看状态”为 False），要求使用 if 语句进行条件判断。

```
DELIMITER $$
CREATE PROCEDURE update_repeat_state()
BEGIN
    -- 1. 修改 Watch 表，增加 Repeat_state 字段，类型为 BOOLEAN
    ALTER TABLE Watch
    ADD COLUMN Repeat_state BOOLEAN DEFAULT FALSE;

    -- 2. 更新 Repeat_state 字段
    -- 使用子查询和 IF 语句判断每个 Viewer_no 和 Movie_no 组合的观看次数
    UPDATE Watch w
    SET Repeat_state = (
        SELECT IF(COUNT(*) > 1, TRUE, FALSE)
        FROM Watch w2
        WHERE w2.Viewer_no = w.Viewer_no
        AND w2.Movie_no = w.Movie_no
    );
END$$
DELIMITER ;
```

CALL update_repeat_state();

```
obclient(root@sys)[exp4]> DELIMITER $$
obclient(root@sys)[exp4]> CREATE PROCEDURE update_repeat_state()
    -> BEGIN
    ->     -- 1. 修改 Watch 表，增加 Repeat_state 字段，类型为 BOOLEAN
    ->     ALTER TABLE Watch
    ->     ADD COLUMN Repeat_state BOOLEAN DEFAULT FALSE;
    ->
    ->     -- 2. 更新 Repeat_state 字段
    ->     -- 使用子查询和 IF 语句判断每个 Viewer_no 和 Movie_no 组合的观看次数
    ->     UPDATE Watch w
    ->     SET Repeat_state = (
    ->         SELECT IF(COUNT(*) > 1, TRUE, FALSE)
    ->         FROM Watch w2
    ->         WHERE w2.Viewer_no = w.Viewer_no
    ->         AND w2.Movie_no = w.Movie_no
    ->     );
    -> END$$
Query OK, 0 rows affected (0.049 sec)

obclient(root@sys)[exp4]> DELIMITER ;
obclient(root@sys)[exp4]> CALL update_repeat_state();
Query OK, 8 rows affected (0.113 sec)
```

⑼在上题的基础上，查询没有重复观看过电影的观众信息，输出观众姓名和编号

SELECT v.Viewer_name, v.Viewer_no

FROM Viewer v

JOIN Watch w ON v.Viewer_no = w.Viewer_no

GROUP BY v.Viewer_no, v.Viewer_name

HAVING SUM(CASE WHEN w.Repeat_state = TRUE THEN 1 ELSE 0 END) = 0

ORDER BY v.Viewer_no;

```
obclient(root@sys)[exp4]> SELECT v.Viewer_name, v.Viewer_no
    -> FROM Viewer v
    -> JOIN Watch w ON v.Viewer_no = w.Viewer_no
    -> GROUP BY v.Viewer_no, v.Viewer_name
    -> HAVING SUM(CASE WHEN w.Repeat_state = TRUE THEN 1 ELSE 0 END) = 0
    -> ORDER BY v.Viewer_no;
+-------------+-----------+
| Viewer_name | Viewer_no |
+-------------+-----------+
| 王红        | V002      |
| 张磊        | V003      |
| 赵颖        | V004      |
| 孙阳        | V005      |
+-------------+-----------+
4 rows in set (0.029 sec)
```

⑽修改电影表，在 Movie_name 列上增加唯一性索引 Movie_name_index，并按 Movie_name 升序排列。

CREATE UNIQUE INDEX Movie_name_index

ON Movie (Movie_name DESC);

```
obclient(root@sys)[exp4]> CREATE UNIQUE INDEX Movie_name_index
    -> ON Movie (Movie_name DESC);
Query OK, 0 rows affected (0.386 sec)

obclient(root@sys)[exp4]> SHOW INDEX FROM Movie;
+-------+------------+----------------+--------------+------------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Table | Non_unique | Key_name       | Seq_in_index | Column_name | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment | Index_comment | Visible | Expression |
+-------+------------+----------------+--------------+------------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Movie |          0 | PRIMARY        |            1 | Movie_no   | A           |        NULL | NULL   | NULL |      | BTREE      |         | available     | YES     | NULL       |
| Movie |          0 | Movie_name_index |          1 | Movie_name | A           |        NULL | NULL   | NULL | YES  | BTREE      |         | available     | YES     | NULL       |
+-------+------------+----------------+--------------+------------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
2 rows in set (0.030 sec)
```

```
obclient(root@sys)[exp4]> CREATE UNIQUE INDEX Movie_name_index
    -> ON Movie (Movie_name DESC);
Query OK, 0 rows affected (0.386 sec)
```

出现的问题：

使用过程增加字段“重复观看状态”时失败

在调用 update_repeat_state() 过程时，SQL 语句出现列名重复的问题，显示 repeat_state 这个列名可能被重复引用了，无法正常运行。

```
obclient(root@sys)[exp4]> CALL update_repeat_state();
ERROR 1060 (42S21): Duplicate column name 'Repeat_state'
[172.17.0.2:2882] [2025-04-30 15:39:10.444253] [YB42AC110002-00063400B7472744-0-0]
```

过程重复执行问题

如果多次调用 update_repeat_state()，而没有检查 Repeat_state 列是否已存在，会导致列名重复的错误

解决方案：

使用过程增加字段“重复观看状态”时失败

经检查，Watch 表中已经存在 repeat_state 列，可能是因为之前创建的错误的过程，虽然没有全部正确运行，但是加上了这一列，删去着一列后重新运行过程，即可成功。

```
obclient(root@sys)[exp4]> desc Watch;
+--------------+-------------+------+------+---------+-------+
| Field        | Type        | Null | Key  | Default | Extra |
+--------------+-------------+------+------+---------+-------+
| S_no         | varchar(10) | NO   | PRI  | NULL    |       |
| Viewer_no    | varchar(10) | NO   | PRI  | NULL    |       |
| Movie_no     | varchar(10) | NO   | PRI  | NULL    |       |
| Watch_date   | datetime    | YES  |      | NULL    |       |
| Repeat_state | tinyint(1)  | YES  |      | 0       |       |
+--------------+-------------+------+------+---------+-------+
5 rows in set (0.002 sec)
```

过程重复执行问题的解决

修改后的存储过程，添加列存在性检查。

```
IF column_exists = 0 THEN
    ALTER TABLE Watch
    ADD COLUMN Repeat_state BOOLEAN DEFAULT FALSE;
END IF;
```