实验四：SQL 综合实验

一、实验目的

1. 熟练运用基础的 SQL 语句，包括数据库的创建，数据的插入、修改、删除、查询操作，数据库表的完整性，存储过程的创建和调用、数据库表索引的创建(*)；
2. 掌握 SQL 语句常见语法错误的调试方法。

二、实验要求说明

1. 本次实验使用 OceanBase 数据库完成，版本要求 ≥ 4.2.2.0；
2. 完成下列 SQL 综合练习的所有小题内容（序号后标有*的小题是选做题，不强制要求完成），要求提交的实验报告中包含每个小题的题干和相应的实验结果截图；
3. 实验截图中应该包含有完整的 SQL 执行语句和输出的实验结果；
4. 数据库表的表名、字段名要求与提供的表格和附录一中的代码片段保持一致，表内的字段类型自行设计，满足要求合理即可；
5. 插入数据请参考附录二中的代码片段。

三、实验内容

1. 建表（见附录一），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。

表名：Movie	
字段	**说明**
Movie_no	电影编号
Movie_name	电影名称
Director	导演
Rating	评分
End_date	电影停映日期，现实日期小于这个日期视为电影未下架，现实日期大于这个日期视为电影已下架

表名：Viewer	
字段	**说明**
Viewer_no	观众编号
Viewer_name	观众姓名
Age	年龄

P.S. Watch 表的 Repeat_state 字段（用灰色标注的）是第 9 小题需要额外添加的字段，此处建表时可暂时忽略；

表名：Watch	
字段	**说明**
S_no	电影场次编号
Viewer_no	观众编号，外键
Movie_no	电影编号，外键
Watch_date	观影日期
Repeat_state	重复观看状态（是否重复观看电影）

2. 插入样例数据（见附录二）。
3. 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。
4. 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。
5. 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。
6. 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。
7. 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。
8. 创建一个过程，使之能够实现如下功能：
9. 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；
10. 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。

    P. S. 创建存储过程的语法可参考如下代码片段，其中存储过程的名称、是否带参数、参数的名称类型自行决定，合理即可；

```
# 设置分隔符为$$，这样存储过程中出现的分号(;)不会被当成分隔符
delimiter $$

# 创建存储过程
create procedure update_rstate()
begin
```

```
# 使用分隔符$$表示语句的结束
end$$

# 重新设置分隔符为分号(;)
delimiter ;

# 调用存储过程
call update_rstate();
```

11. 在 8-10 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。
12. (*)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。

附录

1. 建表样例

```
create table Movie (
    Movie_no varchar(10) primary key,
    Movie_name varchar(50),
    Director varchar(30),
    Rating float,
    End_date datetime
);

create table Viewer (
    Viewer_no varchar(10) primary key,
    Viewer_name varchar(30),
    Age int
);

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
);
```

2. 插入数据样例

```
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');
```