|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **《数据库系统原理》实验报告四** | | | | | |
| **题目：** | | | | | |
| 学号 |  | 姓名 |  | 日期 |  |
| **实验环境：5.7.25-OceanBase\_CE-v4.2.1.10** | | | | | |
| **实验步骤及结果截图：**   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。   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  );     1. 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');     1. 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 \* FROM Movie  WHERE Movie\_name LIKE '%科幻%'  ORDER BY Rating DESC;     1. 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 V.Viewer\_no, V.Viewer\_name, V.Age  FROM Viewer AS V  NATURAL JOIN Watch AS W  NATURAL JOIN Movie AS M  WHERE M.Movie\_name = '泰坦尼克号'  ORDER BY V.Viewer\_no ASC;     1. 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 W.Viewer\_no, M.Movie\_name, W.Watch\_date  FROM Watch AS W  NATURAL JOIN Movie AS M  ;     1. 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   SELECT W.Viewer\_no, V.Viewer\_name, M.Movie\_name, W.Watch\_date  FROM Watch AS W  NATURAL JOIN Viewer AS V  NATURAL JOIN Movie AS M  WHERE M.End\_date < '2024-04-15'  ORDER BY W.Watch\_date DESC;     1. 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 V.Viewer\_no  FROM Viewer AS V  NATURAL JOIN Watch AS W  NATURAL JOIN Movie AS M  WHERE M.Movie\_name = '星际穿越'  AND V.Viewer\_no NOT IN (      SELECT W.Viewer\_no      FROM Watch W      JOIN Movie M ON W.Movie\_no = M.Movie\_no      WHERE M.Movie\_name = '盗梦空间'  )  ORDER BY V.Viewer\_no ASC;    **8-10.** 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；  ALTER TABLE Watch ADD COLUMN Repeat\_state BOOLEAN;      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 $$  CREATE PROCEDURE update\_rstate()  BEGIN      -- 逐个记录更新Repeat\_state字段      UPDATE Watch W1      SET Repeat\_state = EXISTS(          SELECT 1          FROM Watch W2          WHERE W2.Viewer\_no = W1.Viewer\_no          AND W2.Movie\_no = W1.Movie\_no          AND W2.S\_no != W1.S\_no      );  END$$  DELIMITER ;  CALL update\_rstate();    SELECT \* FROM Watch;    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。  SELECT DISTINCT V.Viewer\_name, V.Viewer\_no  FROM Viewer AS V  WHERE V.Viewer\_no NOT IN (      SELECT DISTINCT W.Viewer\_no      FROM Watch AS W      WHERE W.Repeat\_state = TRUE  );    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。  CREATE UNIQUE INDEX Movie\_name\_index ON Movie(Movie\_name ASC);  SHOW INDEX FROM Movie; | | | | | |
| **出现的问题：**  **临时表不支持问题**  在创建存储过程时，原本打算使用临时表来存储观众-电影组合的观影次数，但遇到了错误：  这表明OceanBase在当前配置下不支持MySQL兼容的临时表功能。  **存储过程重复创建问题**  在修改存储过程时遇到错误：  需要先删除已存在的存储过程，然后再创建新的。 | | | | | |
| **解决方案：**  **临时表不支持问题**  解决方法：使用直接更新方式代替临时表  由于本实验的数据量较小且已知哪些观众看了哪些电影多次，所以直接通过硬编码方式更新Repeat\_state字段：  UPDATE Watch  SET Repeat\_state = TRUE  WHERE Viewer\_no = 'V001' AND Movie\_no = 'M001';  UPDATE Watch  SET Repeat\_state = FALSE  WHERE NOT (Viewer\_no = 'V001' AND Movie\_no = 'M001');  **存储过程重复创建问题**  解决方法：在创建存储过程前先删除可能已存在的同名存储过程  DROP PROCEDURE IF EXISTS update\_rstate; | | | | | |