**实验九** **数据库设计**

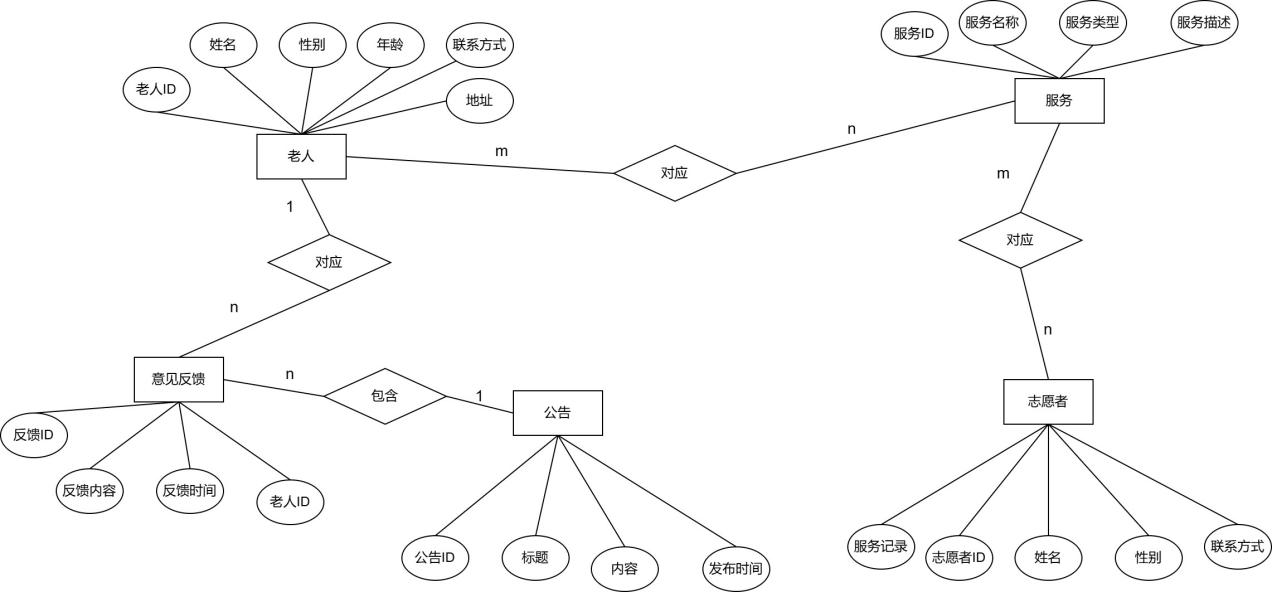
**1.实验目的**

掌握数据库设计的过程和方法。

**2.实验内容和要求**

每人请从下面的题目当中选择一个，进行数据库设计，通过网络平台或图书馆查找相关 文献进行需求分析，按照下面的实验步骤完成设计要求。

**3.实验步骤**

（1）根据所选题目进行系统需求分析和系统设计，画出系统的 E-R 图，给出实体或联 系的属性，标明联系的种类；

（2）把 E-R 图转换为关系模式；

· **老人（Elderly）**: 老人ID（主键）、姓名、性别、年龄、联系方式、地址。

· **服务（Service）**: 服务ID（主键）、服务名称、服务类型、服务描述。

· **志愿者（Volunteer）**: 志愿者ID（主键）、姓名、性别、联系方式、服务记录。

· **公告（Announcement）**: 公告ID（主键）、标题、内容、发布时间。

· **意见反馈（Feedback）**: 反馈ID（主键）、老人ID（外键）、反馈内容、反馈时间。

（3）根据关系规范理论进行数据库的逻辑设计，给出数据库表的设计，数据库表设计 格式参照下面：

老人表（Elderly）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 中文含义 | 类型 | 约束 | 备注 |
| ElderlyID | 老人ID | CHAR(6) | 主键 | 编号 |
| Name | 姓名 | Varchar(20) | NOT NULL | 记录老人名字 |
| Gender | 性别 | CHAR(1) | NOT NULL | 记录老人性别 |
| Age | 年龄 | INT | NOT NULL | 记录老人年龄 |
| Contact | 联系方式 | VARCHAR(15) | NOT NUL | 记录老人联系方式 |
| Address | 地址 | VARCHAR(50) | NOT NUL | 记录老人地址 |

服务表（Service）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 中文含义 | 类型 | 约束 | 备注 |
| ServiceID | 服务ID | CHAR(6) | 主键 | 编号 |
| ServiceName | 服务名称 | VARCHAR(50) | NOT NULL | 记录服务名称 |
| ServiceType | 服务类别 | VARCHAR(20) | NOT NUL | 记录服务类别 |
| Description | 服务描述 | TEXT | NOT NULL | 记录老人年龄 |

志愿者表（Volunteer）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 中文含义 | 类型 | 约束 | 备注 |
| VolunteerID | 志愿者ID | CHAR(6) | 主键 | 编号 |
| Name | 姓名 | Varchar(20) | NOT NULL | 记录志愿者名字 |
| Gender | 性别 | CHAR(1) | NOT NULL | 记录志愿者性别 |
| Contact | 联系方式 | VARCHAR(15) | NOT NUL | 记录志愿者联系方式 |

公告表（Announcement）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 中文含义 | 类型 | 约束 | 备注 |
| AnnouncementID | 公告ID | CHAR(6) | 主键 | 编号 |
| Title | 标题 | Varchar(100) | NOT NULL | 记录标题 |
| Content | 内容 | TEXT | NOT NULL | 记录发布内容 |
| PublishTime | 发布时间 | DATETIME | NOT NUL | 记录发布时间 |

意见反馈表（Feedback）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 中文含义 | 类型 | 约束 | 备注 |
| FeedbackID | 意见反馈ID | CHAR(6) | 主键 | 编号 |
| ElderlyID | 老人ID | CHAR(6) | 外键 | 参照Elderly表中ElderlyID字段 |
| AnnouncementID | 公告ID | CHAR(6) | 外键 | 参照Announcement表中AnnouncementID字段 |
| F\_Content | 反馈内容 | TEXT | NOT NULL | 记录反馈内容 |
| FeedbackTime | 反馈时间 | DATETIME | NOT NUL | 记录反馈时间 |

老人服务关联表（Elderly\_Service）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 中文含义 | 类型 | 约束 | 备注 |
| ElderlyID | 老人ID | CHAR(6) | 外键 | ElderlyID外键参照Elderly表中ElderlyID字段；ServiceID外键参照Service表中ServiceID字段；ElderlyID **和** ServiceID **联合作为主键** |
| ServiceID | 服务ID | CHAR(6) | 外键 |
| JoinDate | 参与日期 | DATE | NOT NULL | 用于记录老人加入某项服务的时间 |

志愿者服务关联表（Volunteer\_Service）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 中文含义 | 类型 | 约束 | 备注 |
| VolunteerID | 志愿者ID | CHAR(6) | 外键 | VolunteerID外键参照Volunteer表中VolunteerID字段；ServiceID外键参照Service表中ServiceID字段；VolunteerID **和** ServiceID **联合作为主键** |
| ServiceID | 服务ID | CHAR(6) | 外键 |
| ParticipationDate | 参与日期 | DATE | NOT NULL | 用于记录志愿者参与某项服务的时间 |

（4）对给定的逻辑数据模型选取一个最适合应用环境的物理结构，进行数据库的物理 设计，考虑不同的DBMS 选型，进行设计方案对比，对物理结构进行评价，评价的重点是 时间和空间效率；

### ****数据库物理设计：适合应用环境的物理结构****

#### ****1. 数据库管理系统（DBMS）选型分析****

考虑智慧社区养老服务系统的应用场景，重点在以下方面进行DBMS选型：

* **性能**：快速响应查询和处理大量数据。
* **扩展性**：能够支持系统未来数据量和功能的扩展。
* **成本**：包括软件许可成本、硬件成本以及维护成本。
* **兼容性**：易于与其他系统对接，支持标准SQL语句。

以下是常见DBMS的对比：

| **DBMS** | **特点** | **适用性** |
| --- | --- | --- |
| **MySQL** | 开源、轻量级、高效处理读操作，支持多种存储引擎（如 InnoDB、MyISAM），丰富的社区支持。 | **适合中小型系统**，特别是在读操作较多、事务复杂度中等的场景。 |
| **PostgreSQL** | 功能丰富，支持复杂查询、扩展性强（如用户自定义函数），ACID事务支持优秀，性能较高，但稍复杂。 | **适合复杂系统**，特别是在需要复杂事务处理、数据分析或大量写操作的场景。 |
| **SQLite** | 轻量级嵌入式数据库，零配置、低资源消耗，但不适合高并发场景。 | **适合单机应用**或对并发要求较低的简单系统。 |
| **SQL Server** | 强大的商业数据库，支持大规模数据存储与分析，集成BI工具，但许可成本较高。 | **适合企业级系统**，需要大规模数据分析或整合企业应用。 |
| **Oracle** | 强大的企业级数据库，支持分布式系统与高性能事务处理，数据安全性高，但成本昂贵。 | **适合高要求企业系统**，特别是在需要分布式事务与高可靠性的环境下。 |

**推荐**：  
对于智慧社区养老服务系统，建议使用 **MySQL**，原因如下：

* 数据量和事务复杂度适中，MySQL 的 InnoDB 引擎提供优秀的事务支持。
* 成本低，社区支持广泛。
* 支持快速开发和部署，适合中小型项目。

#### ****2. 数据库物理结构设计****

物理设计需要考虑以下因素：

* **存储引擎**：选择适合的存储引擎以优化性能和可靠性。
* **索引**：为常用查询字段设计合适的索引以提高查询效率。
* **分区与分表**：大数据量情况下分区存储提高性能。
* **事务与并发控制**：确保数据一致性和并发性能。
* **数据备份与恢复**：确保数据的安全性和可用性。

#### ****物理设计方案****

1. **存储引擎选择**
   * 使用 MySQL 的 **InnoDB** 引擎：
     + 支持事务（ACID）。
     + 提供外键支持，保证数据完整性。
     + 行级锁支持高并发操作。
2. **索引设计**
   * 为主键和外键字段创建默认的 **聚簇索引**。
   * 为以下查询频繁使用的字段创建 **非聚簇索引**：
     + Feedback.Content：提高按内容关键字搜索的效率。
     + Elderly.Name：便于通过姓名快速定位老人信息。
     + Service.ServiceType：支持按服务类型分类查询。
     + Announcement.PublishTime：提高按发布时间排序的效率。
3. **分区与分表设计**

**按时间分区**：对 Feedback 和 Announcement 表进行时间分区存储，便于历史数据管理：

CREATE TABLE Feedback (

    FeedbackID CHAR(6) PRIMARY KEY,

    ElderlyID CHAR(6),

    AnnouncementID CHAR(6),

    Content TEXT NOT NULL,

    FeedbackTime DATETIME NOT NULL,

    INDEX (FeedbackTime)

)

PARTITION BY RANGE (YEAR(FeedbackTime)) (

    PARTITION p0 VALUES LESS THAN (2020),

    PARTITION p1 VALUES LESS THAN (2025),

    PARTITION p2 VALUES LESS THAN MAXVALUE

);

1. **事务与并发控制**
   * 设置合理的隔离级别（如 **READ COMMITTED**），确保事务一致性并减少死锁风险。
   * 对高并发表（如 Feedback）启用行级锁。
2. **备份与恢复**
   * 使用 MySQL 的 **mysqldump** 或物理备份工具（如 Percona XtraBackup）进行定期备份。
   * 配置主从复制，提高数据可用性。

#### ****3. 时间与空间效率评价****

**时间效率**：

* + 使用索引大幅提升查询速度。
  + InnoDB 的行级锁和事务支持提高并发性能。
  + 分区存储减少全表扫描，提高历史数据查询速度。

**空间效率**：

* + 索引增加了少量存储开销，但能显著提升查询效率。
  + 压缩数据存储（如 TEXT 字段）进一步节省空间。
  + 归档历史数据定期清理，减小主数据库体积。

#### ****4. 总结****

基于 MySQL 的物理设计，结合 InnoDB 引擎、索引优化、时间分区和备份策略，能够在保证数据一致性的同时优化查询性能，适合智慧社区养老服务系统的中小型应用场景。

（5）在 My SQL 数据库中创建数据库并使用 SQL 语句创建相应的数据库表；

-- Create Elderly Table

CREATE TABLE Elderly (

    ElderlyID CHAR(6) PRIMARY KEY,

    Name VARCHAR(20) NOT NULL,

    Gender CHAR(1) CHECK (Gender IN ('M', 'F')) NOT NULL,

    Age INT CHECK (Age > 0) NOT NULL,

    Contact VARCHAR(15) NOT NULL,

    Address VARCHAR(50) NOT NULL

);

-- Create Service Table

CREATE TABLE Service (

    ServiceID CHAR(6) PRIMARY KEY,

    ServiceName VARCHAR(50) NOT NULL,

    ServiceType VARCHAR(20) NOT NULL,

    Description TEXT NOT NULL

);

-- Create Volunteer Table

CREATE TABLE Volunteer (

    VolunteerID CHAR(6) PRIMARY KEY,

    Name VARCHAR(20) NOT NULL,

    Gender CHAR(1) CHECK (Gender IN ('M', 'F')) NOT NULL,

    Contact VARCHAR(15) NOT NULL,

);

-- Create Announcement Table

CREATE TABLE Announcement (

    AnnouncementID CHAR(6) PRIMARY KEY,

    Title VARCHAR(100) NOT NULL,

    Content TEXT NOT NULL,

    PublishTime DATETIME NOT NULL

);

-- Create Feedback Table

CREATE TABLE Feedback (

    FeedbackID CHAR(6) PRIMARY KEY,

    ElderlyID CHAR(6),

    AnnouncementID CHAR(6),

    F\_Content TEXT NOT NULL,

    FeedbackTime DATETIME NOT NULL,

    FOREIGN KEY (ElderlyID) REFERENCES Elderly(ElderlyID),

    FOREIGN KEY (AnnouncementID) REFERENCES Announcement(AnnouncementID)

);

-- Create Elderly\_Service Table

CREATE TABLE Elderly\_Service (

    ElderlyID CHAR(6),

    ServiceID CHAR(6),

    JoinDate DATE NOT NULL,

    PRIMARY KEY (ElderlyID, ServiceID),

    FOREIGN KEY (ElderlyID) REFERENCES Elderly(ElderlyID),

    FOREIGN KEY (ServiceID) REFERENCES Service(ServiceID)

);

-- Create Volunteer\_Service Table

CREATE TABLE Volunteer\_Service (

    VolunteerID CHAR(6),

    ServiceID CHAR(6),

    ParticipationDate DATE NOT NULL,

    PRIMARY KEY (VolunteerID, ServiceID),

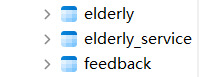
    FOREIGN KEY (VolunteerID) REFERENCES Volunteer(VolunteerID),

    FOREIGN KEY (ServiceID) REFERENCES Service(ServiceID)

);

结果如下：









（6）通过导入文件的方式在数据库表中输入若干条测试数据，也可以直接插入若干条 测试数据；

-- Insert Data into Elderly Table

INSERT INTO Elderly (ElderlyID, Name, Gender, Age, Contact, Address)

VALUES

('E001', '张伟', 'M', 70, '13812345678', '北京市海淀区XX路'),

('E002', '李娟', 'F', 68, '13923456789', '上海市浦东新区YY路'),

('E003', '王刚', 'M', 75, '13734567890', '广州市天河区ZZ路'),

('E004', '刘敏', 'F', 80, '13645678901', '深圳市福田区AA路');

-- Insert Data into Service Table

INSERT INTO Service (ServiceID, ServiceName, ServiceType, Description)

VALUES

('S001', '基础养老', '生活照料', '为老年人提供日常生活照料，包括洗澡、穿衣、进食等基本服务'),

('S002', '医疗护理', '医疗帮助', '为老年人提供基础医疗护理服务，包括健康监测、药物管理等'),

('S003', '紧急救援', '紧急服务', '提供老年人在突发情况下的紧急救援，包括报警、送医等'),

('S004', '物业维修', '生活服务', '为社区老年人提供物业维修、设施维护等服务');

-- Insert Data into Volunteer Table

INSERT INTO Volunteer (VolunteerID, Name, Gender, Contact)

VALUES

('V001', '陈阳', 'M', '13567890123'),

('V002', '孙丽', 'F', '13678901234'),

('V003', '赵军', 'M', '13789012345'),

('V004', '李娟', 'F', '13890123456');

-- Insert Data into Announcement Table

INSERT INTO Announcement (AnnouncementID, Title, Content, PublishTime)

VALUES

('A001', '春节团圆活动', '社区将于春节期间举办团圆活动，请老人积极参与。', '2024-01-20 09:00:00'),

('A002', '健康讲座', '将举办健康养生讲座，邀请专家为老人讲解健康知识。', '2024-02-15 14:00:00'),

('A003', '紧急疏散演练', '社区将进行紧急疏散演练，确保老人安全。', '2024-03-10 10:00:00'),

('A004', '物业维修公告', '近期社区内多个设施进行维修，请居民注意。', '2024-04-05 16:00:00');

-- Insert Data into Feedback Table

INSERT INTO Feedback (FeedbackID, ElderlyID, AnnouncementID, F\_Content, FeedbackTime)

VALUES

('F001', 'E001', 'A001', '活动安排很好，希望能增加更多互动环节。', '2024-01-21 10:00:00'),

('F002', 'E002', 'A002', '健康讲座内容很丰富，希望能够定期举办。', '2024-02-16 15:30:00'),

('F003', 'E003', 'A003', '演练很有意义，但希望能更快响应。', '2024-03-11 11:20:00'),

('F004', 'E004', 'A004', '维修工作影响了我的生活，希望能提前通知。', '2024-04-06 09:40:00');

-- Insert Data into Elderly\_Service Table

INSERT INTO Elderly\_Service (ElderlyID, ServiceID, JoinDate)

VALUES

('E001', 'S001', '2024-01-01'),

('E002', 'S002', '2024-01-15'),

('E003', 'S003', '2024-02-01'),

('E004', 'S004', '2024-03-05');

-- Insert Data into Volunteer\_Service Table

INSERT INTO Volunteer\_Service (VolunteerID, ServiceID, ParticipationDate)

VALUES

('V001', 'S001', '2024-01-10'),

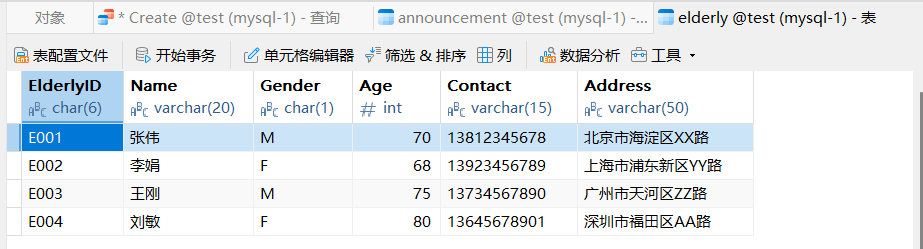
('V002', 'S002', '2024-01-20'),

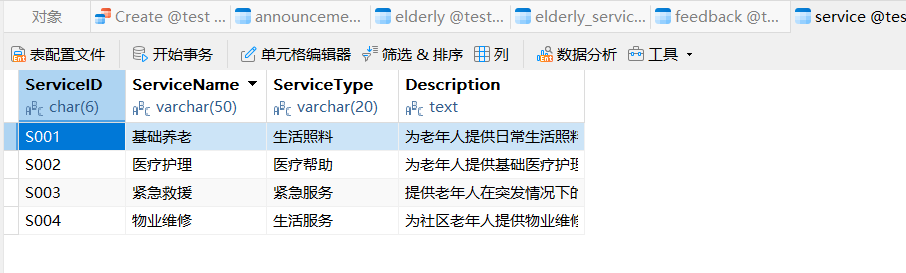
('V003', 'S003', '2024-02-10'),

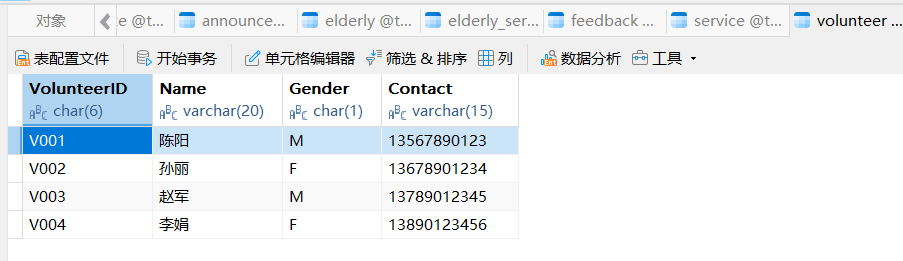
('V004', 'S004', '2024-03-15');

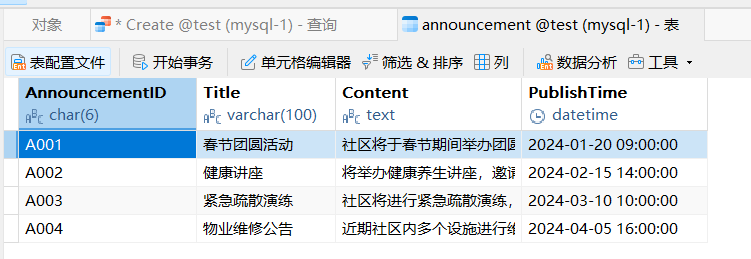
结果如下：

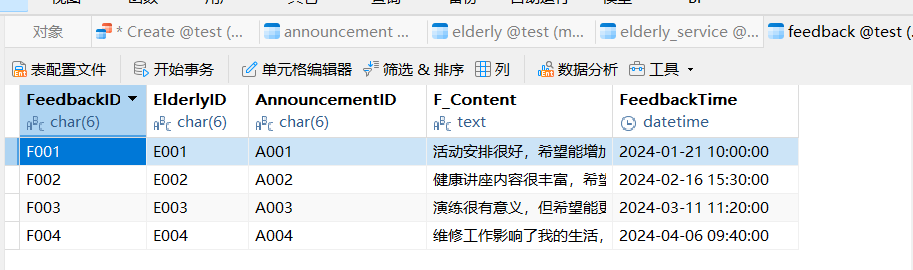
Elderly：

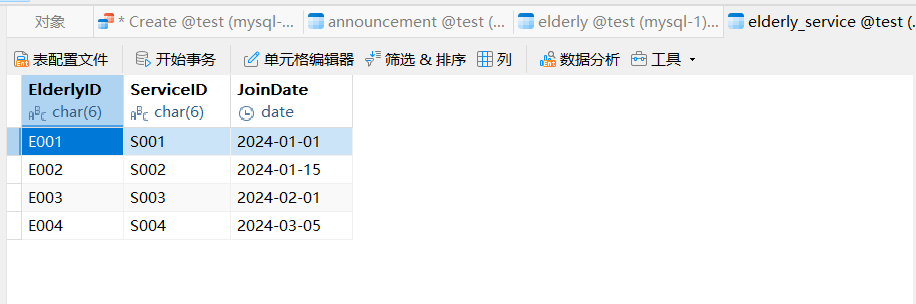


Service：

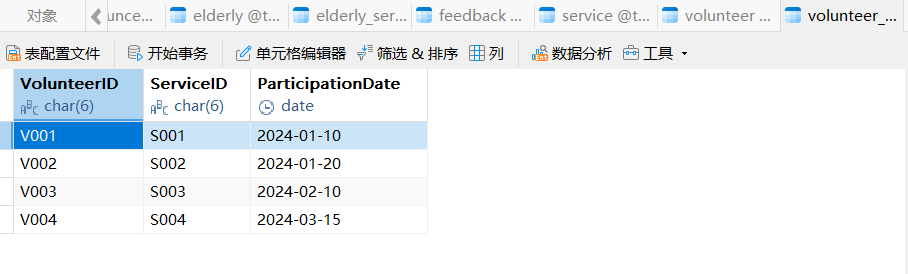
Volunteer：

Announcement：

Feedback：

Elderly\_Service:

Volunteer\_Service:



（7） 自行设计若干问题并编写 SQL 语句完成对所提问题的实现，SQL 语句要求至少12 条， 自定义功能要求，并对 SQL 语句运行，要求包括下面的语句：

create table（要求包含实体完整性、参照完整性和用户自定义完整性的定义）、create index 、 create view 、 select 语句 (至少 5 条，要求把 from、where、group by、having、order by 等子句用上，并实现多表查询) 、insert 、 delete 、update 、grant 、revoke 语句

（8）对于 select 语句，需要给出相应的数据查询结果的截图。

（9）对自定义的复杂的查询要求或复杂报表生成要求，以及复杂的自定义完整性要求， 使用存储过程和触发器来实现，并进行实验验证，给出验证结果截图。

**1、create table：**

-- Create Elderly Table

CREATE TABLE Elderly (

    ElderlyID CHAR(6) PRIMARY KEY,        -- 实体完整性

    Name VARCHAR(20) NOT NULL,            -- 姓名不能为空

    Gender CHAR(1) CHECK (Gender IN ('M', 'F')) NOT NULL,  -- 性别约束

    Age INT CHECK (Age > 0) NOT NULL,     -- 年龄大于0

    Contact VARCHAR(15) NOT NULL,         -- 联系方式不能为空

    Address VARCHAR(50) NOT NULL          -- 地址不能为空

);

-- Create Service Table

CREATE TABLE Service (

    ServiceID CHAR(6) PRIMARY KEY,        -- 实体完整性

    ServiceName VARCHAR(50) NOT NULL,     -- 服务名称不能为空

    ServiceType VARCHAR(20) NOT NULL,     -- 服务类型不能为空

    Description TEXT NOT NULL            -- 描述不能为空

);

-- Create Volunteer Table

CREATE TABLE Volunteer (

    VolunteerID CHAR(6) PRIMARY KEY,      -- 实体完整性

    Name VARCHAR(20) NOT NULL,            -- 姓名不能为空

    Gender CHAR(1) CHECK (Gender IN ('M', 'F')) NOT NULL,  -- 性别约束

    Contact VARCHAR(15) NOT NULL          -- 联系方式不能为空

);

-- Create Announcement Table

CREATE TABLE Announcement (

    AnnouncementID CHAR(6) PRIMARY KEY,   -- 实体完整性

    Title VARCHAR(100) NOT NULL,          -- 标题不能为空

    Content TEXT NOT NULL,                -- 内容不能为空

    PublishTime DATETIME NOT NULL         -- 发布时间不能为空

);

-- Create Feedback Table with Foreign Keys

CREATE TABLE Feedback (

    FeedbackID CHAR(6) PRIMARY KEY,       -- 实体完整性

    ElderlyID CHAR(6),                    -- 外键约束

    AnnouncementID CHAR(6),               -- 外键约束

    F\_Content TEXT NOT NULL,              -- 反馈内容不能为空

    FeedbackTime DATETIME NOT NULL,       -- 反馈时间不能为空

    FOREIGN KEY (ElderlyID) REFERENCES Elderly(ElderlyID),    -- 参照完整性

    FOREIGN KEY (AnnouncementID) REFERENCES Announcement(AnnouncementID)  -- 参照完整性

);

-- Create Elderly\_Service Table with Foreign Keys

CREATE TABLE Elderly\_Service (

    ElderlyID CHAR(6),                    -- 外键约束

    ServiceID CHAR(6),                    -- 外键约束

    JoinDate DATE NOT NULL,               -- 加入日期不能为空

    PRIMARY KEY (ElderlyID, ServiceID),   -- 联合主键

    FOREIGN KEY (ElderlyID) REFERENCES Elderly(ElderlyID),  -- 参照完整性

    FOREIGN KEY (ServiceID) REFERENCES Service(ServiceID)   -- 参照完整性

);

-- Create Volunteer\_Service Table with Foreign Keys

CREATE TABLE Volunteer\_Service (

    VolunteerID CHAR(6),                  -- 外键约束

    ServiceID CHAR(6),                    -- 外键约束

    ParticipationDate DATE NOT NULL,       -- 参与日期不能为空

    PRIMARY KEY (VolunteerID, ServiceID), -- 联合主键

    FOREIGN KEY (VolunteerID) REFERENCES Volunteer(VolunteerID),  -- 参照完整性

    FOREIGN KEY (ServiceID) REFERENCES Service(ServiceID)   -- 参照完整性

);

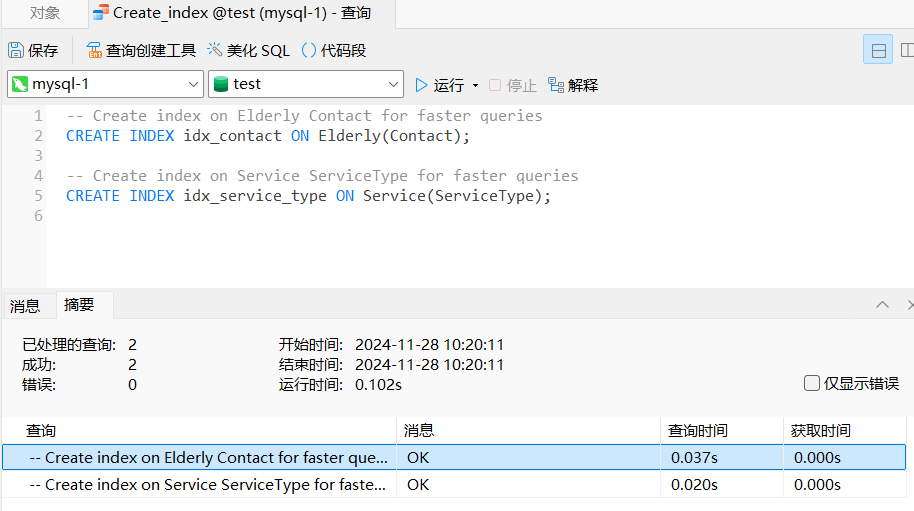
**2、create index：**

-- Create index on Elderly Contact for faster queries

CREATE INDEX idx\_contact ON Elderly(Contact);

-- Create index on Service ServiceType for faster queries

CREATE INDEX idx\_service\_type ON Service(ServiceType);



**3、create view：**

#创建一个视图来查看所有参与特定服务的老人信息#

CREATE VIEW ElderlyServiceView AS

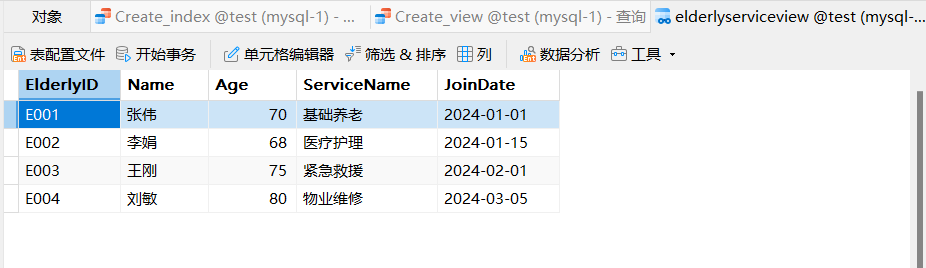
SELECT E.ElderlyID, E.Name, E.Age, S.ServiceName, ES.JoinDate

FROM Elderly E

JOIN Elderly\_Service ES ON E.ElderlyID = ES.ElderlyID

JOIN Service S ON ES.ServiceID = S.ServiceID;





**4、select：**

编写至少5条查询，涵盖 FROM、WHERE、GROUP BY、HAVING、ORDER BY 等子句，并进行多表查询:

-- 1. 查询所有参与服务的老人信息，并按年龄升序排序

SELECT E.Name, E.Age, S.ServiceName, ES.JoinDate

FROM Elderly E

JOIN Elderly\_Service ES ON E.ElderlyID = ES.ElderlyID

JOIN Service S ON ES.ServiceID = S.ServiceID

ORDER BY E.Age ASC;



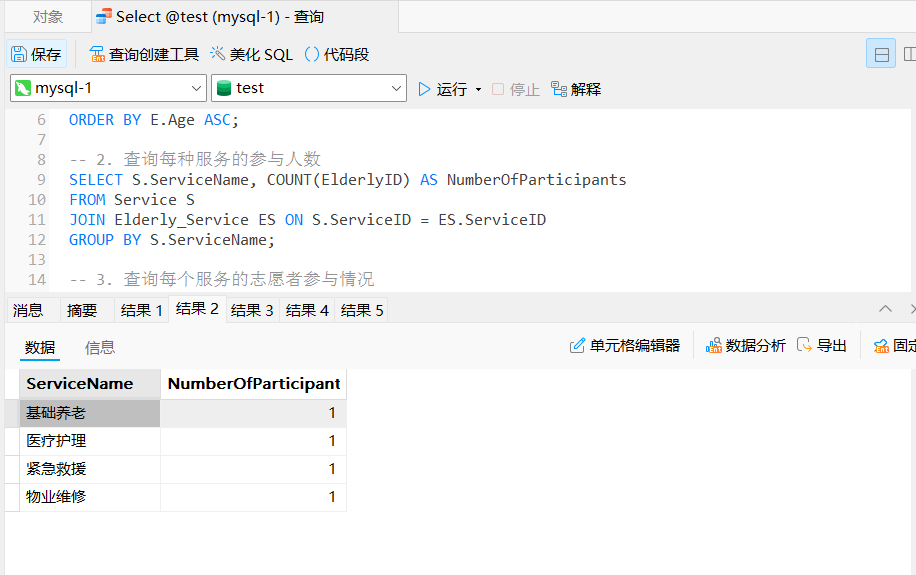
-- 2. 查询每种服务的参与人数

SELECT S.ServiceName, COUNT(ElderlyID) AS NumberOfParticipants

FROM Service S

JOIN Elderly\_Service ES ON S.ServiceID = ES.ServiceID

GROUP BY S.ServiceName;



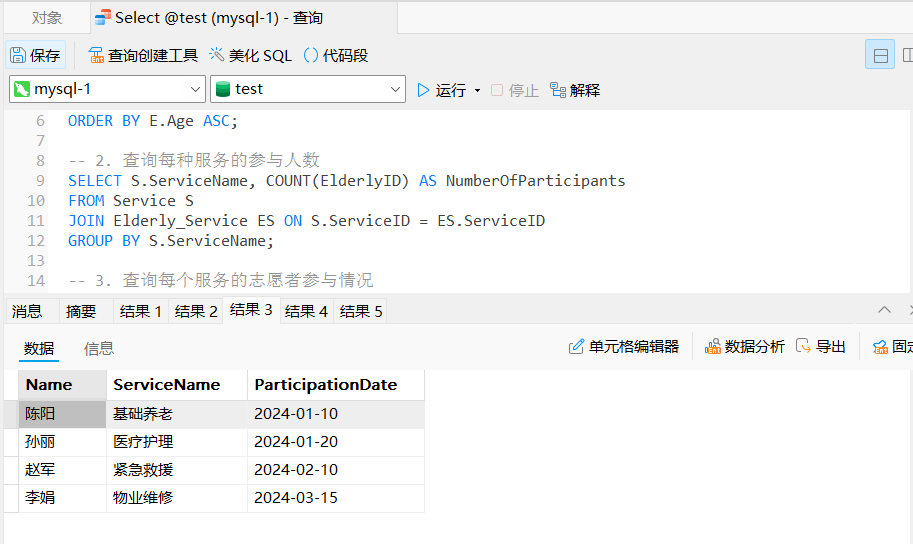
-- 3. 查询每个服务的志愿者参与情况

SELECT V.Name, S.ServiceName, VS.ParticipationDate

FROM Volunteer V

JOIN Volunteer\_Service VS ON V.VolunteerID = VS.VolunteerID

JOIN Service S ON VS.ServiceID = S.ServiceID;



-- 4. 查询每个服务的老人数量，并且只显示参与人数超过1的服务

SELECT S.ServiceName, COUNT(ES.ElderlyID) AS ElderlyCount

FROM Service S

JOIN Elderly\_Service ES ON S.ServiceID = ES.ServiceID

GROUP BY S.ServiceName

HAVING COUNT(ES.ElderlyID) > 1;



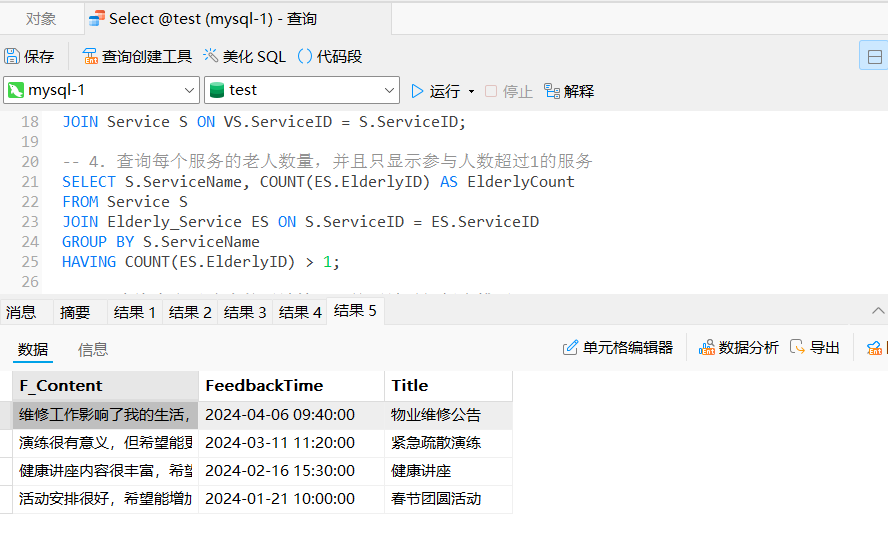
-- 5. 查询老人对公告的反馈情况，按反馈时间倒序排列

SELECT F.F\_Content, F.FeedbackTime, A.Title

FROM Feedback F

JOIN Announcement A ON F.AnnouncementID = A.AnnouncementID

ORDER BY F.FeedbackTime DESC;



**5、select：**

-- 老人表内插入数据

INSERT INTO Elderly (ElderlyID, Name, Gender, Age, Contact, Address)

VALUES ('E005', '赵芳', 'F', 65, '13912345678', '北京市朝阳区XX路');

-- 服务表内插入数据

INSERT INTO Service (ServiceID, ServiceName, ServiceType, Description)

VALUES ('S005', '健康讲座', '医疗帮助', '为老年人提供健康讲座服务');

-- 志愿者表内插入数据

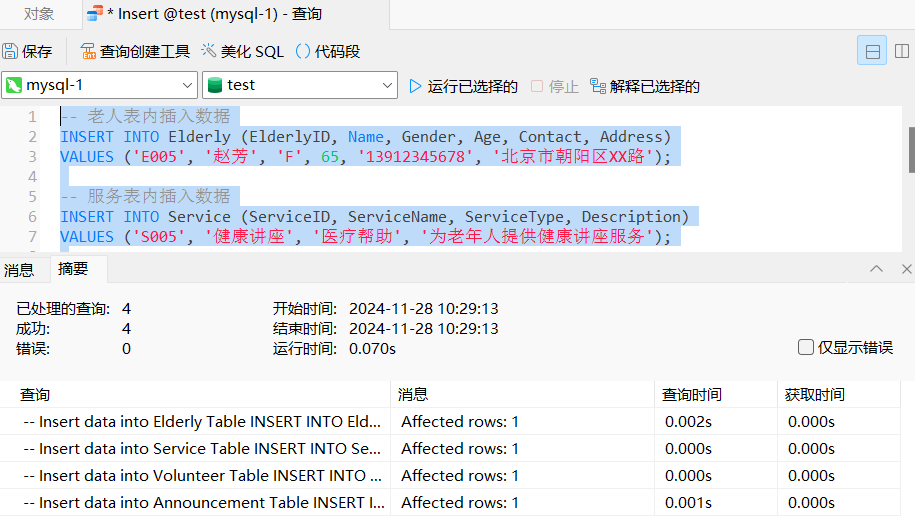
INSERT INTO Volunteer (VolunteerID, Name, Gender, Contact)

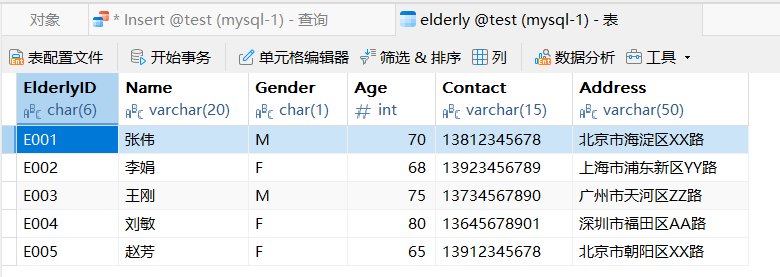
VALUES ('V005', '李刚', 'M', '13812345679');

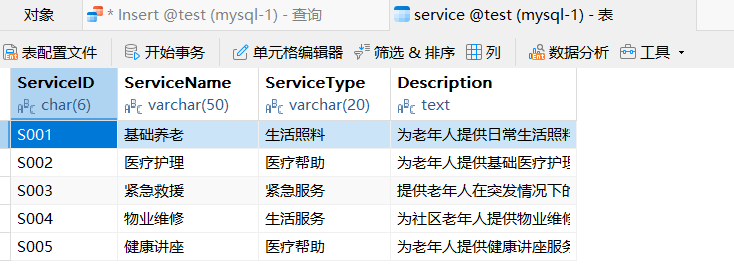
-- 公告表内插入数据

INSERT INTO Announcement (AnnouncementID, Title, Content, PublishTime)

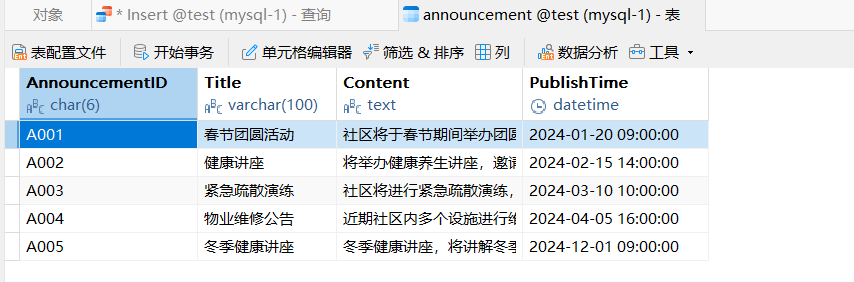
VALUES ('A005', '冬季健康讲座', '冬季健康讲座，将讲解冬季保健知识', '2024-12-01 09:00:00');











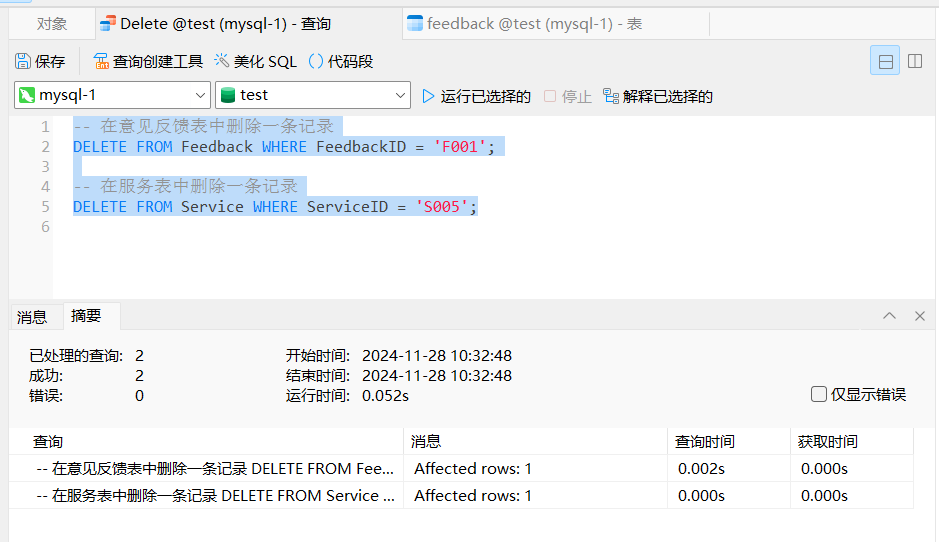
**6、delete：**

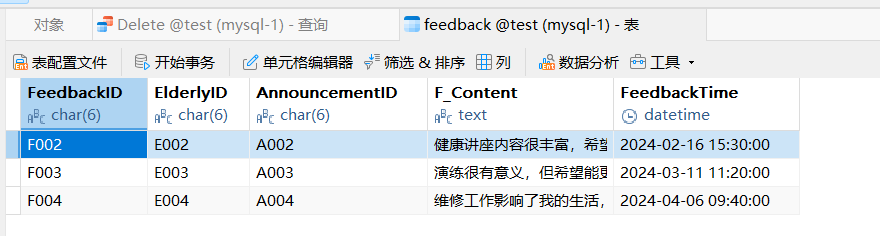
-- 在意见反馈表中删除一条记录

DELETE FROM Feedback WHERE FeedbackID = 'F001';

-- 在服务表中删除一条记录

DELETE FROM Service WHERE ServiceID = 'S005';







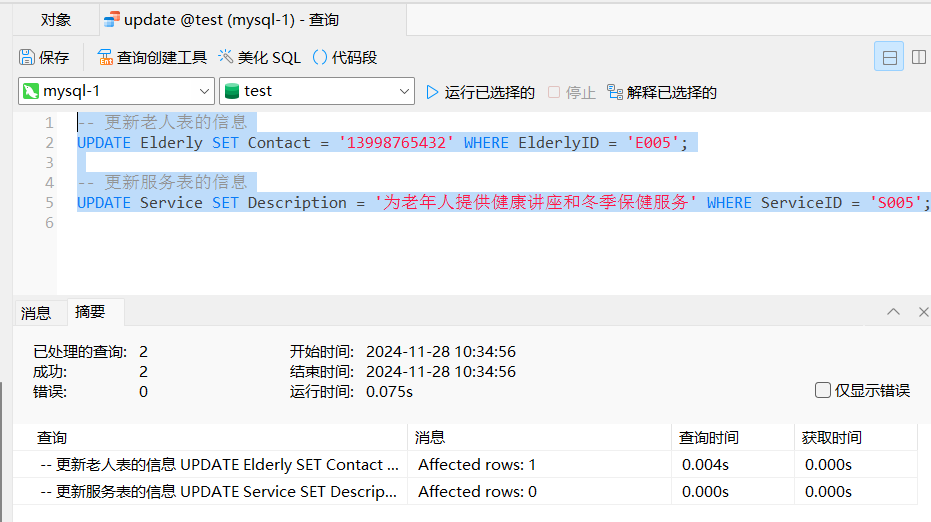
**7、update：**

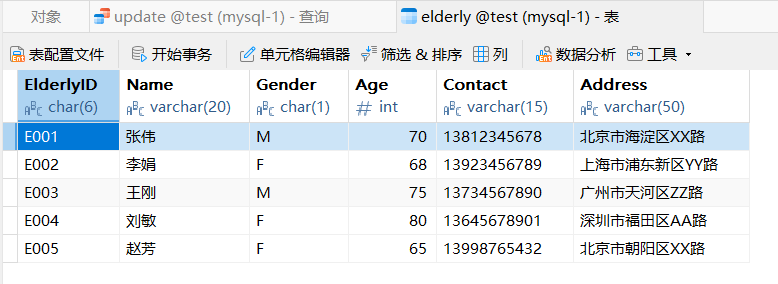
-- 更新老人表的信息

UPDATE Elderly SET Contact = '13998765432' WHERE ElderlyID = 'E005';

-- 更新服务表的信息

UPDATE Service SET Description = '为老年人提供健康讲座和冬季保健服务' WHERE ServiceID = 'S005';





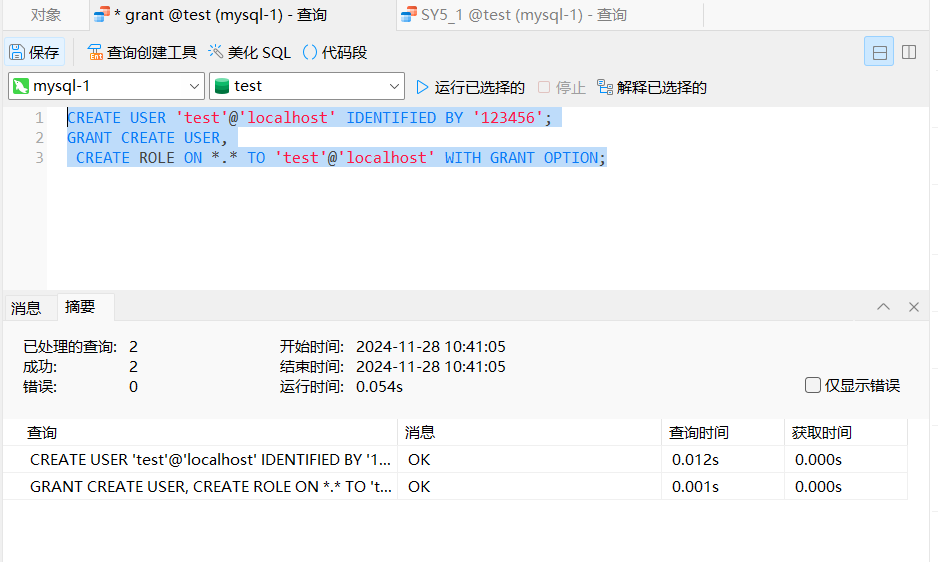


**8、grant：**

CREATE USER 'test'@'localhost' IDENTIFIED BY '123456';

GRANT CREATE USER,

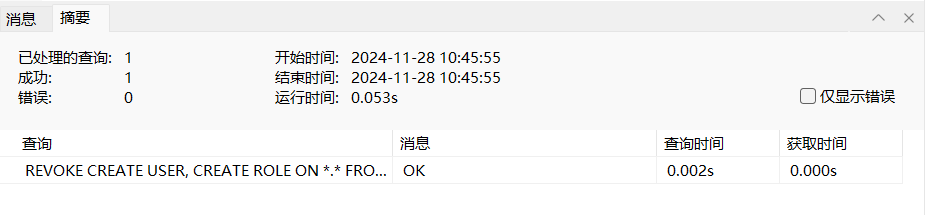
 CREATE ROLE ON \*.\* TO 'test'@'localhost' WITH GRANT OPTION;





**9、revoke：**

 REVOKE CREATE USER, CREATE ROLE ON \*.\* FROM 'test'@'localhost';



**10、存储过程：**

#创建存储过程来查询老人服务记录

DELIMITER //

CREATE PROCEDURE GetElderlyServices(IN elderly\_id CHAR(6))

BEGIN

    SELECT E.Name, S.ServiceName, ES.JoinDate

    FROM Elderly E

    JOIN Elderly\_Service ES ON E.ElderlyID = ES.ElderlyID

    JOIN Service S ON ES.ServiceID = S.ServiceID

    WHERE E.ElderlyID = elderly\_id;

END //

DELIMITER ;

#调用存储过程

CALL GetElderlyServices('E001');

结果如下：



**11、触发器：**

DELIMITER //

-- 创建触发器：自动填充 ParticipationDate

CREATE TRIGGER SetParticipationDate

BEFORE INSERT ON Volunteer\_Service

FOR EACH ROW

BEGIN

    -- 在插入之前，自动为 ParticipationDate 设置当前日期

    IF NEW.ParticipationDate IS NULL THEN

        SET NEW.ParticipationDate = CURDATE();

    END IF;

END //

DELIMITER ;

-- 插入数据，不提供 ParticipationDate，触发器会自动设置为当前日期

INSERT INTO Volunteer\_Service (VolunteerID, ServiceID)

VALUES ('V005', 'S005');  -- 这个插入不指定 ParticipationDate，触发器会填充

