

上机内容

实验环境

- SQLServer 2017
- 客户端：
 - Navicat Premium 12.1.10 64-bit
 - Visual Studio Code 1.29.1

一、需求分析

用DDL语句创建习题中的SPJ数据库及基本表、约束、视图等数据库对象；用DML语句插入、删除、修改教材中的实例数据；选择完成作业中的查询（不少于10个）；编写存储过程，实现按供应商号查询该供应商的供应信息；结合教材中SPJ数据库，修改关系S，增加等级属性列（level），编写触发器，当更新SPJ表中的QTY列时，取该供应商的供应量总和除以100作为其对应等级。

信息要求

需要为 供应商（S）、零件表（P）、工程项目表（J）建立数据库以存储信息

- 供应商 需要记录：供应商代码（SNO）、供应商姓名（SNAME）、供应商状态（STATUS）、供应商所在城市（CITY）、供应商等级
- 零件信息有：零件代码（PNO）、零件名（PNAME）、颜色（COLOR）、重量（WEIGHT）
- 工程项目信息有：工程项目代码（JNO）、工程项目名（JNAME）、工程项目所在城市（CITY）
- 供应情况有：供应商代码（SNO）、零件代码（PNO）、工程项目代码（JNO）、供应数量（QTY）

处理要求

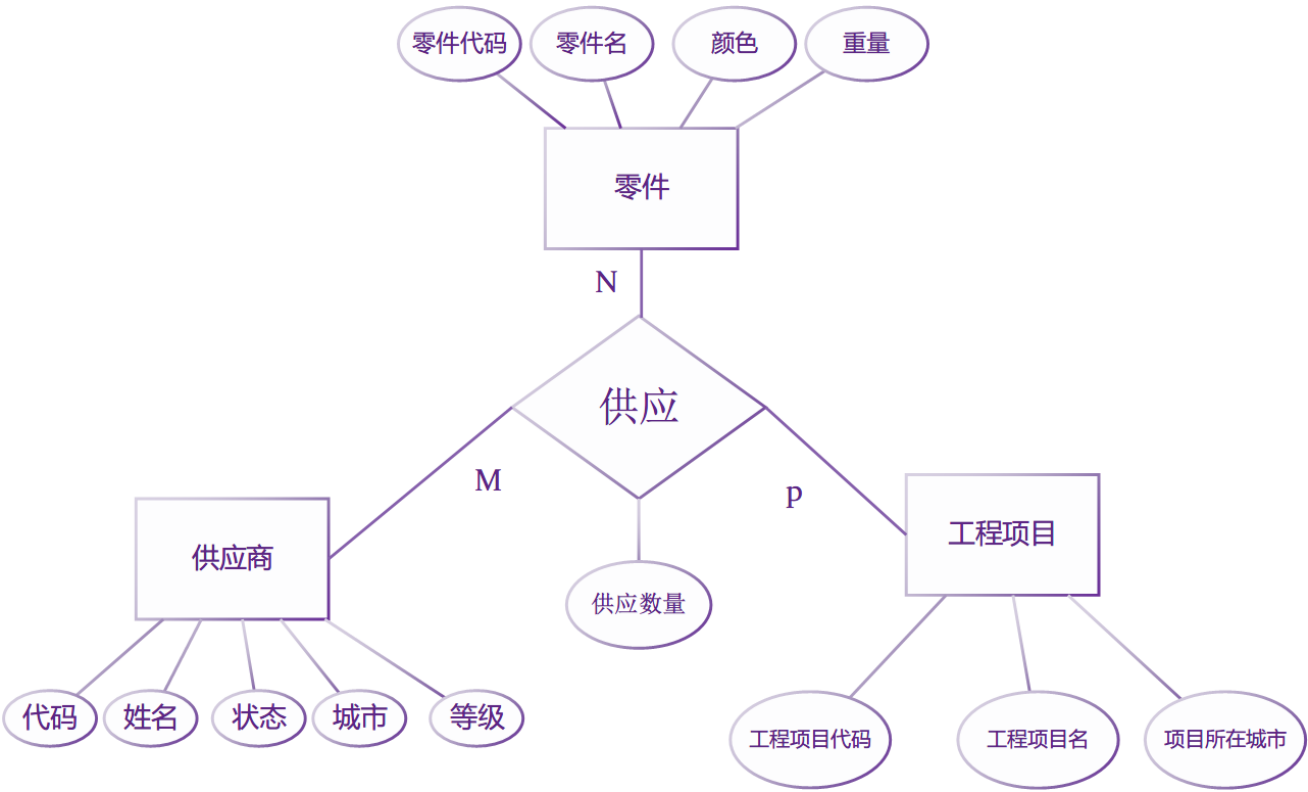
供应商等级需要根据供应商的供应量总和自动计算

安全性与完整性要求

- 供应商
 - 供应商代码唯一
 - 供应商代码、供应商姓名、供应商状态、供应商所在城市 非空
- 零件信息
 - 零件代码唯一
 - 零件名、颜色、重量 非空
- 工程项目信息
 - 工程项目代码非空唯一
 - 工程项目名、工程项目所在城市 非空
- 供应情况
 - 供应商代码、零件代码、工程项目代码、供应数量 非空

二、概念结构设计

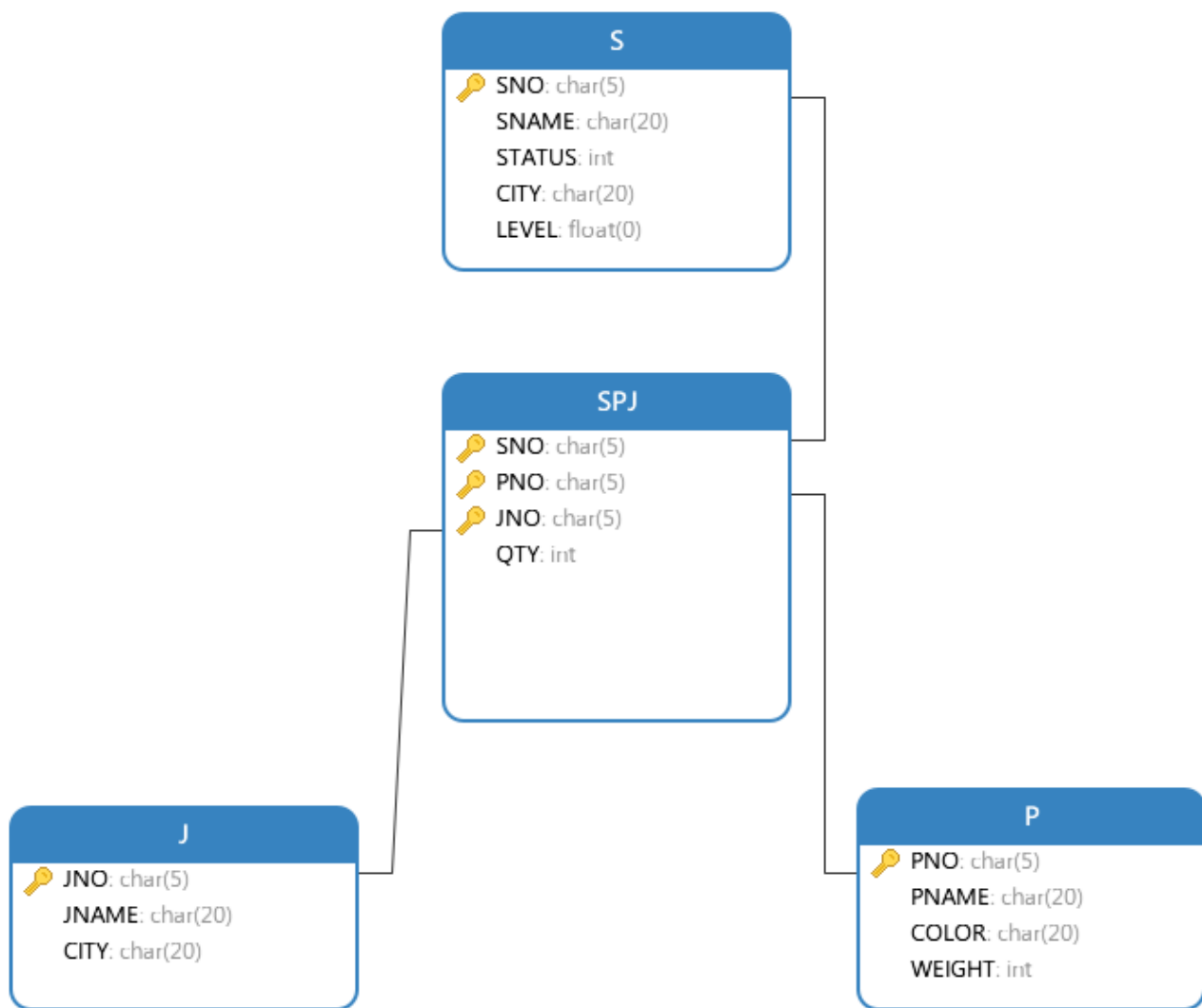
根据需求，设计 E-R图如下：



三、逻辑结构设计

根据 E-R 图，设计关系模型如下：

对每个实体定义的属性：



四、源代码以及详细实现

1.用DDL语句创建习题中的SPJ数据库及基本表、约束、视图等数据库对象；

```
1 CREATE TABLE S(  
2     SNO CHAR(5) PRIMARY KEY,  
3     SNAME CHAR(20),  
4     STATUS INTEGER,  
5     CITY CHAR(20),  
6 );  
7  
8 CREATE TABLE P(  
9     PNO CHAR(5) PRIMARY KEY,  
10    PNAME CHAR(20),  
11    COLOR CHAR(20),  
12    WEIGHT INTEGER,  
13 );  
14  
15 CREATE TABLE J(  
16     JNO CHAR(5) PRIMARY KEY,  
17     JNAME CHAR(20),
```

```

18     CITY CHAR(20),
19 );
20
21 CREATE TABLE SPJ(
22     SNO CHAR(5) NOT NULL,
23     PNO CHAR(5) NOT NULL,
24     JNO CHAR(5) NOT NULL,
25     QTY INTEGER,
26     PRIMARY KEY (SNO,PNO,JNO,QTY),
27     FOREIGN key (PNO) REFERENCES P(PNO),
28     FOREIGN key (JNO) REFERENCES J(JNO),
29     FOREIGN key (SNO) REFERENCES S(SNO),
30 );

```

对象
* 无标题 - 查询

保存
查询创建工具
美化 SQL
() 代码段

MS SQLserver
learnsql2
dbo
运行
停止
解释

```

1 CREATE TABLE S(
2     SNO CHAR(5) PRIMARY KEY,
3     SNAME CHAR(20),
4     STATUS INTEGER,
5     CITY CHAR(20),
6 );
7
8 CREATE TABLE P(
9     PNO CHAR(5) PRIMARY KEY,
10    PNAME CHAR(20),
11    COLOR CHAR(20)
12 );
13
14 CREATE TABLE J(
15     JNO CHAR(5) PRIMARY KEY,
16     JNAME CHAR(20),
17     CITY CHAR(20),
18 );
19
20 CREATE TABLE SPJ(
21     SNO CHAR(5) NOT NULL,
22     PNO CHAR(5) NOT NULL,
23     JNO CHAR(5) NOT NULL,
24     QTY INTEGER,
25     PRIMARY KEY (SNO,PNO,JNO,QTY),
26     FOREIGN key (PNO) REFERENCES P(PNO),
27     FOREIGN key (JNO) REFERENCES J(JNO),
28     FOREIGN key (SNO) REFERENCES S(SNO),
29 );

```

信息

```

> OK
> 时间: 0.004s

CREATE TABLE J(
    JNO CHAR(5) PRIMARY KEY,
    JNAME CHAR(20),
    CITY CHAR(20),
);
> OK
> 时间: 0.003s

CREATE TABLE SPJ(
    SNO CHAR(5) NOT NULL,
    PNO CHAR(5) NOT NULL,
    JNO CHAR(5) NOT NULL,
    QTY INTEGER,
    PRIMARY KEY (SNO,PNO,JNO,QTY),
    FOREIGN key (PNO) REFERENCES P(PNO),
    FOREIGN key (JNO) REFERENCES J(JNO),
    FOREIGN key (SNO) REFERENCES S(SNO),
);
> OK
> 时间: 0.006s

```

建立 `spj_detail` 视图, 实现 供应商供应情况详细信息的查询。

```

1 CREATE VIEW spj_detail
2 AS
3 SELECT J.JNO,J.JNAME,J.CITY as
   JCITY,P.PNO,P.PNAME,P.COLOR,P.WEIGHT,S.SNO,S.SNAME,S.CITY AS SCITY
4 FROM J,P,S,SPJ
5 WHERE J.JNO = SPJ.JNO and
6        P.PNO = SPJ.PNO and
7        S.SNO = SPJ.SNO
8 GO

```

执行结果:

信息

```

CREATE VIEW spj_detail
AS
SELECT J.JNO,J.JNAME,J.CITY as JCITY,P.PNO,P.PNAME,P.COLOR,P.WEIGHT,S.SNO,S.SNAME,S.CITY AS SCITY
      FROM J,P,S,SPJ
      WHERE J.JNO = SPJ.JNO and
            P.PNO = SPJ.PNO and
            S.SNO = S.SNO

> OK
> 时间: 0.028s

```

JNO	JNAME	JCITY	PNO	PNAME	COLOR	WEIGHT	SNO	SNAME	SCITY
J1	三建	北京	P1	螺母	红	12	S1	精益	天津
J1	三建	北京	P1	螺母	红	12	S2	盛锡	北京
J1	三建	北京	P1	螺母	红	12	S3	东方红	北京
J1	三建	北京	P1	螺母	红	12	S4	丰泰盛	天津
J1	三建	北京	P1	螺母	红	12	S5	为民	上海
J3	弹簧厂	天津	P1	螺母	红	12	S1	精益	天津
J3	弹簧厂	天津	P1	螺母	红	12	S2	盛锡	北京
J3	弹簧厂	天津	P1	螺母	红	12	S3	东方红	北京
J3	弹簧厂	天津	P1	螺母	红	12	S4	丰泰盛	天津
J3	弹簧厂	天津	P1	螺母	红	12	S5	为民	上海
J4	造船厂	天津	P1	螺母	红	12	S1	精益	天津
J4	造船厂	天津	P1	螺母	红	12	S2	盛锡	北京
J4	造船厂	天津	P1	螺母	红	12	S3	东方红	北京
J4	造船厂	天津	P1	螺母	红	12	S4	丰泰盛	天津
J4	造船厂	天津	P1	螺母	红	12	S5	为民	上海
J2	一汽	长春	P2	螺栓	绿	17	S1	精益	天津
J2	一汽	长春	P2	螺栓	绿	17	S2	盛锡	北京
J2	一汽	长春	P2	螺栓	绿	17	S3	东方红	北京
J2	一汽	长春	P2	螺栓	绿	17	S4	丰泰盛	天津
J2	一汽	长春	P2	螺栓	绿	17	S5	为民	上海
J1	三建	北京	P3	螺丝刀	蓝	14	S1	精益	天津
J1	三建	北京	P3	螺丝刀	蓝	14	S2	盛锡	北京
J1	三建	北京	P3	螺丝刀	蓝	14	S3	东方红	北京
J1	三建	北京	P3	螺丝刀	蓝	14	S4	丰泰盛	天津
J1	三建	北京	P3	螺丝刀	蓝	14	S5	为民	上海

2. 用DML语句插入、删除、修改教材中的实例数据;

```

1 INSERT
2 into [dbo].[S]
3 VALUES
4 ('S1','精益',20,'天津'),
5 ('S2','盛锡',10,'北京'),
6 ('S3','东方红',30,'北京'),
7 ('S4','丰泰盛',20,'天津'),
8 ('S5','为民',30,'上海');
9

```

```

10 INSERT
11 into [dbo].[P]
12 VALUES
13 ('P1','螺母','红',12),
14 ('P2','螺栓','绿',17),
15 ('P3','螺丝刀','蓝',14),
16 ('P4','螺丝刀','红',14),
17 ('P5','凸轮','蓝',40),
18 ('P6','齿轮','红',30);
19
20 INSERT
21 into [dbo].[J]
22 VALUES
23 ('J1','三建','北京'),
24 ('J2','一汽','长春'),
25 ('J3','弹簧厂','天津'),
26 ('J4','造船厂','天津'),
27 ('J5','机车厂','唐山'),
28 ('J6','无线电厂','常州'),
29 ('J7','半导体厂','南京');
30
31 INSERT
32 into SPJ
33 VALUES
34 ('S1','P1','J1',200),
35 ('S1','P1','J3',100),
36 ('S1','P1','J4',700),
37 ('S1','P2','J2',100),
38 ('S2','P3','J1',400),
39 ('S2','P3','J2',200),
40 ('S2','P3','J4',500),
41 ('S2','P3','J5',400),
42 ('S2','P5','J1',400),
43 ('S2','P5','J2',100),
44 ('S3','P1','J1',200),
45 ('S3','P3','J1',200),
46 ('S4','P5','J1',100),
47 ('S4','P6','J3',300),
48 ('S4','P6','J4',200),
49 ('S5','P2','J4',100),
50 ('S5','P3','J1',200),
51 ('S5','P6','J2',200),
52 ('S5','P6','J4',500);

```

实例数据插入后的数据表：

对象	J @learnsql.dbo (MS SQLser...	P @learnsql.dbo (MS SQLse...
开始事务	文本	筛选
JNO	JNAME	CITY
J1	三建	北京
J2	一汽	长春
J3	弹簧厂	天津
J4	造船厂	天津
J5	机车厂	唐山
J6	无线电厂	常州
J7	半导体厂	南京

对象	J @learnsql.dbo (MS SQLser...	P @learnsql.dbo (MS SQLse...
开始事务	文本	筛选
SNO	SNAME	STATUS
S1	精益	20 天津
S2	盛锡	10 北京
S3	东方红	30 北京
S4	丰泰盛	20 天津
S5	为民	30 上海

对象	J @learnsql.dbo (MS SQLser...	P @learnsql.dbo (MS SQLse...
开始事务	文本	筛选
SNO	PNO	JNO
S1	P1	J1
S1	P1	J3
S1	P1	J4
S1	P2	J2
S2	P3	J1
S2	P3	J2
S2	P3	J4
S2	P3	J5
S2	P5	J1
S2	P5	J2
S3	P1	J1
S3	P3	J1
S4	P5	J1
S4	P6	J3
S4	P6	J4
S5	P2	J4
S5	P3	J1
S5	P6	J2
S5	P6	J4

3.编写存储过程，实现按供应商号查询该供应商的供应信息

编写存储过程 `get_s_info`，其参数：`@sno CHAR(5)` 为待查询的供应商 `SNO`

```
1 ALTER PROC get_s_info
2 @sno CHAR(5)
3 AS
4 BEGIN
5 SELECT DISTINCT * FROM spj_detail WHERE Sno = @sno
6 END
```

执行存储过程

```
1 exec get_s_info 'S1'
```

执行存储过程结果如下，可以看到，存储过程成功返回了所查询的信息

定义	注释	信息	结果 1	SQL 预览						
JNO	JNAME	JCITY	PNO	PNAME	COLOR	WEIGHT	SNO	SNAME	SCITY	
J1	三建	北京	P1	螺母	红	12	S1	精益	天津	
J3	弹簧厂	天津	P1	螺母	红	12	S1	精益	天津	
J4	造船厂	天津	P1	螺母	红	12	S1	精益	天津	
J2	一汽	长春	P2	螺栓	绿	17	S1	精益	天津	

4. 结合教材中SPJ数据库，修改关系S，增加等级属性列（level），编写触发器，当更新SPJ表中的QTY列时，取该供应商的供应量总和除以100作为其对应等级。


信息要求

结合教材中SPJ数据库，修改关系S，增加等级属性列（level），编写触发器，当更新SPJ表中的QTY列时，取该供应商的供应量总和除以100作为其对应等级。

编写触发器 `make_level` 如下：

```
1  ---- 增加等级属性列
2  ALTER TABLE S
3      ADD level FLOAT NULL
4  GO
5
6
7  CREATE TRIGGER make_level
8  ON SPJ
9  AFTER UPDATE,INSERT,DELETE
10 AS
11 BEGIN
12     declare @sumqty int
13     -- select rows from a Table 'spj'
14     SELECT @sumqty = SUM(SPJ.QTY) FROM SPJ,inserted
15     WHERE SPJ.SNO = inserted.SNO
16     UPDATE S set level = @sumqty/100 FROM S, inserted
17     WHERE S.SNO = inserted.SNO
18 END
19 GO
```

执行sql 语句后，可以看到数据库中增加了 `make_level` 触发器：



The screenshot displays the SQL Server Enterprise Manager interface. The left pane shows the 'SPJ' table with a trigger named 'make_level' attached. The trigger is configured to fire 'After' updates, inserts, and deletes. The right pane shows the trigger's definition, which is as follows:

```
1 BEGIN
2     declare @sumqty int
3     -- Select rows from a Table or View 'spj'
4     SELECT @sumqty = SUM(SPJ.QTY) FROM SPJ,inserted
5     WHERE SPJ.SNO = inserted.SNO /* add search conditions here */
6     UPDATE S set level = @sumqty/100 FROM S, inserted
7     WHERE S.SNO = inserted.SNO
8 END
```

当修改 spj 表中 QTY列时，S 表的对应 `level` 字段会自动更新：

对象

< 1... P @learnsql.dbo (...)

S @learnsql.dbo (...)

开始事务

文本 筛选 排序

导入 导出

SNO	SNAME	STATUS	CITY	level
▶ S1	精益	20	天津	12
S2	盛锡	10	北京	20
S3	东方红	30	北京	4
S4	丰泰盛	20	天津	6
S5	为民	30	上海	10

5. 选择完成作业中的查询