

2005 年上半年软件设计师下午试题答案

答案:

【问题 1】

(1)起点: 学生住宿服务系统 终点: 房主

数据流名: 费用信息或交纳的费用或费用

(2)起点: 房屋文件 终点: 查询房屋或 4

【问题 2】

房主文件 学生文件

【问题 3】

姓名+现住址+电话号码+出生日期+性别

解析:

试题一分析

【问题 1】(6 分)

(1)题目中明确地说明了“房主向系统中加入 I=I『租用的房屋信息时,需交纳一定的费用,由系统自动给出费用信息”,但是在数据流图中却没有相关的数据流。所以,需要补齐的数据流为:

起点: 学生住宿服务系统 终点: 房主

数据流名: 费用信息或交纳的费用或费用

(2)查询房屋需要读取房屋文件,所以数据流的起点和终点为:

起点: 房屋文件 终点: 查询房屋或 4

【问题 2】(4 分)

题目中说明了“将会面信息通知学生和房主,会面信息包括会面时间、地点以及会面双方的基本信息,系统将记录会面信息”。此处要注意会面双方的基本信息也被包含在会面信息中了。所以,安排会面需要查询学生文件和房主文件以获得双方的基本信息。

【问题 3】(5 分)

根据数据流图 4-2 中的加工 3 以及数据流图 4—3 加工 3 的细化图可以看出,学生信息包含了登录信息和注册信息,登录信息为学生 ID 和密码,所以学生信息中除去登录信息就是注册信息了,因此,注册信息为: 姓名+现住址+电话号码+出生日期+性别

参考答案

【问题 1】

(1)起点：学生住宿服务系统终点：房主

数据流名：费用信息或交纳的费用或费用

(2)起点：房屋文件终点：查询房屋或 4

【问题 2】

房主文件学生文件

【问题 3】

姓名+现住址+电话号码+出生日期+性别

试题二答案:

【问题 1】(4 分)

(1)主键：职员号(1 分) 外键：部门号(1 分)

(2)主键：部门号，或部门名(1 分) 外键：负责人代码(1 分)

【问题 2】(4 分，(a)~(d)每空 1 分)

(a)PRIMARYKEY

(b)FOREIGN KEY(部门号) REFERENCES 部门

(c)count(*), Sum(月工资), Avg(月工资)

(d)GROUP by 部门号 HAVING

注：以上答案中的单词可以小写。

【问题 3】(3 分)

(1)该行不能插入“职员”关系，它违反了用户定义完整性中月工资的取值范围必须大于：等于 1000 元，小于等于 8000 元。(1 分)

(2)该行不能插入“职员”关系，因为职员号“60802”在表 2-1 中已存在，违反了实体完整性中主键必须唯一区分关系中的每一个属性。(1 分)

(3)该行可以插入“职员”关系，尽管部门号、电话和办公室为空，但是它表示该职员没有分配到某个部门。(1 分)

试题二分析:

【问题 1】

本试题中，“部门”关系的主键为部门号，“职员”关系的主键为职员号。在“部门”关系中，部门由于负责人也是来自职员关系，所以负责人代码是外键。在“职员”关系中，部门号是“部门”关系的主键，因此部门号是外键。根据题意，“职员”和“部门”的关系模式可表示如下：

职员(职员号，职员姓名，月工资，部门号，办公室，电话)

部门(部门号，部门名，负责人代码，任职时间)

【问题 2】

用 SQL 定义关系模式的一个非常重要的问题是完整性控制。完整性控制应具有三方面的功能：定义功能、检测功能、处理功能(一旦发现违背了完整性约束条件，采取相关的动作来保证数据的完整性)。数据库中最重要约束是声明一个或一组属性形成关系的键。键的约束在 SQL 的 CREATE TABLE 命令中声明。在关系系统中，最重要的完整性约束条件是实体完整性和参照完整性。

1. 实体完整性定义

在关系中只能有一个主键。声明主键有两种方法：

①将 PRIMARY KEY 保留字加在属性类型之后；

②在属性列表中引入一个新元素，该元素包含保留字 PRIMARY KEY 和用圆括号括起的形成该键的属性或属性组列表。

2. 参照完整性

参照完整性定义格式如下：

FOREIGN KEY (属性名) REFERENCES 表名 (属性名)

[ON DELETE [CASCADE|SET NULL]]

参照完整性通过使用保留字 FOREIGN KEY 定义哪些列为外码；REFERENCES 指明外键对应于哪个表的主键；ON DELETE CASCADE 指明删除被参照关系的元组时，同时删除参照关系中的元组；SETNULL 表示置为空值方式。

本试题中，职员关系的主键为职员号，部门关系的主键为部门号，这样，职员关系中的部门号是外键。其中，职员关系的主键职员号可采用如下两种方式定义：

职员号 CHAR (5) PRIMARYKEY 或者是 PRIMARYKEY (职员号)

根据分析问题 2(1)职员关系的 SQL 定义如下：

CreateTable 职员 (职员号 CHAR (5) PRIMARYKEY ,

职员姓名 CHAR (8),

月工资 NUMBER (4),

部门号 CHAR (1),

办公室 CHAR (20),

电话 CHAR (8),

问题 2(2)的关键在于要对职员关系采用分组语句按部门分类,并统计。如果统计的元组个大于等于 2,则在结果集中。根据分析,针对人数大于等于 2 的部门创建视图 D_View(Dept, D_num, D_Totals, D_AvgPay)如下:

```
Create View D_View(Dept, D_num, D_Totals, D_AvgPay)As
( Select 部门号, count(*) ,Sum(月工资), Avg(月工资)
from 职员
```

```
GROUP by 部门号 HAVING count(*)>=2 WHERE 部门号 IS NOT NULL):
```

【问题 3】

本题主要考查完整性定义的约束性。下表是待将插入的记录组

(1)	60811	芦峰	800	1	A 座 201	6883122
(2)	60802	李小楠	3500	2	B 座 202	6883123
(3)	60812	高亚南	2600			

(1)由于在职员表中的定义中,职员的月工资的取值范围必须大于等于 1000 元,小于等于 8000 元。该行不能插入“职员”关系,它违反了用户定义完整性,该条记录不能插入。

(2)该元组不能插入“职员”关系,因为职员号“60802”在职员表中已存在,违反了实体完整性中主键必须唯一区分关系中的每一个属性。

(3)该行可以插入“职员”关系,尽管部门号、电话和办公室为空,但是它表示该职员没有分配到某个部门。

【问题 4】

此题考察的是查询效率的问题。在涉及相关查询的某些情形中,构造临时关系可以提高查询效率。

原来的“职员”关系模式主要的问题是数据冗余和数据修改不一致问题。例如,某部门的职员人数有 100 个,其办公室和电话的属性值将要重复存入 100 次。如果某部门的办公室变了,可能会导致有些职员的办公室属性值修改了,另一些职员的办公室属性值未修改。根据

题意，每个部门有一名负责人、一间办公室、一部电话，因此，为了解决冗余和数据修改不一致的问题，应该将职员关系模式中的属性“办公室”和“电话”放到部门关系模式中，这样修改后的关系模式为：

职员(职员号，职员姓名，月工资，部门号) (1 分)

部门(部门号，部门名，负责人代码，任职时间，办公室，电话)(1 分)

试题三参考答案

(1)Grid[T.row, T.col]=1

(2)T=EndPos

(3)Q.insert(T)

(4)Found≠true

(5)CurPos←Q.delete()

分析

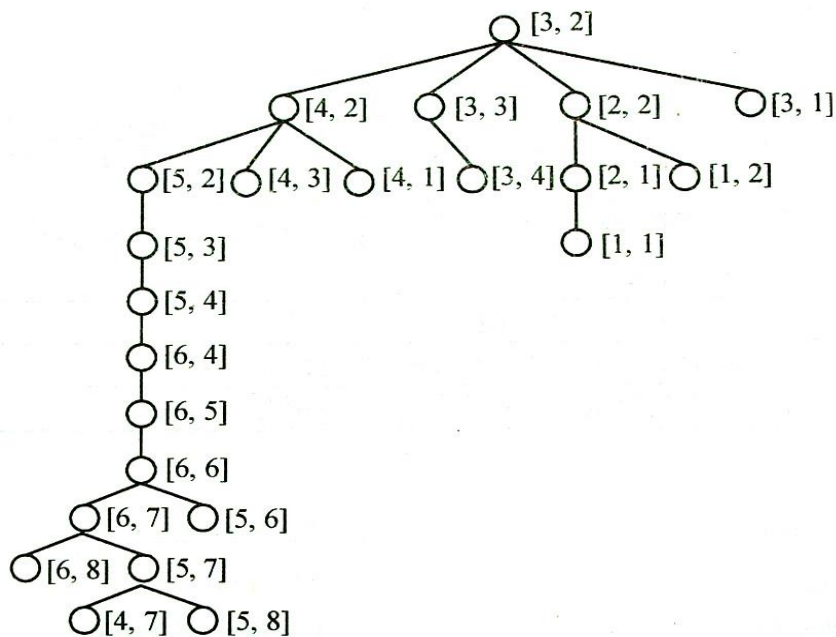
根据题目中的说明，设给定印刷电路板的起始方格 x 与目的方格 y 尚未布线，求这两个方格间最短布线方案的基本思路是：从起始方格 x 开始，先考查距离起始方格距离为 1 的可达方格并用一个路径长度值标记，然后依次考查距离为 2、3、...的可达方格，直到距离为 k 的某一个可达方格就是目标方格 y 时为止，或者由于不存在从 x 到 y 的布线方案而终止。布线区域中的每一个方格与其相邻的上、下、左、右四个方格之间的距离为 1，依次沿 F、右、上、左这四个方向考查，并用一个队列记录可达方格的位置。该方法体现了广度优先搜索策略，以题中的图 4.5 为例，根结点表示起始方格的位置([3, 2])，孩子结点表示一步可达的位置，其过程可用下图所示的树结构表示。

按照广度优先的策略，先将起始位置结点加入队列，此后在队列不为空的情况下，每次从队列中取出一个结点(元素出队列)，按照下、右、上、左的方向依次扩展并将扩展所得的结点加入队列，重复这个过程，直到目标位置结点出现，或队列为空还没有出现目标位置时为止。对于上例，根结点[3, 2]出队列后，扩展出结点[4, 2]、[3, 3]、[2, 2]、[3, 1]并依次加入队列，然后由[4, 2]扩展出[5, 2]、[4, 3]、[4, 1]，[3, 3]扩展出[3, 4]，[2, 2]扩展出[2, 1]、[1, 2]，依次类推，当扩展出目标结点[4, 7]时，路径长度为 10。

在流程图 3-4 描述的上述处理过程中，满足条件 $i < 4$ 且 Found=False 时处理沿 4 个方向进行考查并扩展结点的操作，即

T.rOW=CurPos.row+offset[i].r

T.col=CurPos.col+offset[i].c)



但是方格位置[T.row, T.col]有可能已经封锁(标记为-9)，所以在对可扩展结点(标记为-1)进行路径长度标记时应判断是否可以标记，因此空(1)处应对 Grid[T.row, T.col]的标志进行判断，根据流程中的处理逻辑，显然应填入“Grid[T row, T.col]=-1”。当得到一个扩展结点时 (Orid[T.row, T.col]←Grid[CurPos.row, CurPos.col]+1)，应判断目标结点是否出现，即扩展出的结点 T 是否等于目标结点 EndPos，若是，则可结束扩展操作(Found=True)，否则，将结点 T 加入队列，因此空(2)处填入“T=EndPos”、空(3)处填入“Q.insert(T)”。显然空(4)处表示找到目标方格时的结束条件，根据流程中的处理逻辑应填入“Found≠true”。当尚未找到目标位置结点而队列又不为空时，应从队列中取出一个新的结点作为当前结点进行考查和扩展，因此空(5)处填入“CurPos←Q.delete()”。

试题四参考答案

- (1) $k \geq N$ ，或 $k = N$ ，或其等价形式
- (2) $cost + c[k][i] < mincost$ ，或其等价形式
- (3) i
- (4) $k+1$
- (5) $worker[i] = 0$ ，或其等价形式

试题四分析

首先为函数 Plan()代码加上行号，以便说明。

```

1: void Plan(int k,unsigned int cost)
2: {int i;
3:   if( (1)   && COSt<mincost) {
4:     mLrlcost=cost;
5:     for(i=0: i<N; i++)   temp[i]=task[i];
6:   )
7: Else {
8:   for (i=0; i<N; i++)   /*分配任务 k* /
9:     if(worker[i]=0&& (2) )   {
10:      worker[i]=1; task[k]= (3) ;
11: Plan ( (4) , cost+c[k][i]);
12: (5) ; task[k]=o;
13:   } /*if*/
14: }/*else*/
15: }/*Plan*/

```

由注释可知，在 Plan(k, cost)中，以 k 表示任务编号、cost 表示费用。根据题目中的说明，程序用回溯法计算总费用最小的一种工作分配方案，因此在得到每一个分配方案时需要和先前已经得到的分配方案中的最小费用进行比较。由于需要将 N 个任务分配给 N 个工人，以任务为序时，最后一个任务(第 N-1 个任务)分配之后便得到一种方案，因此第 3 行代码的空(1)处填入“k>=N 或 k==N”。

显然，在分配任务 k 时，需要考查所有的工人(第 8 行代码)，此时若工人 i 尚未接收任务 (worker[i]==0)，并且将任务 k 分配给工人 i 不会超出前面某个方案的费用，则可将任务 k 分配给他(task[k]=i)，然后开始分配第 k+1 个任务。回溯时则需要将分配工人 i 的任务撤销，以便考查其他的分配方案。

试题五答案:

- (1) Shape
- (2) XCircle
- (3) displayIt()

(4) Shape*

(5) getShapelInstance (type)

(6) delete s

试题五分析

题目中明确要求 Circle 具有 Shape 接口，所以，第 1 空应填上 Shape。因为要重用 XCircle 类而不用从头开发一个新的 Circle 类，所以，凡是 Circle 类实现 Shape 的接口时都应调用相应的 XCircle 类提供的方法。因此第 2 空应填上 Xcircle，第 3 空应填上 displayItO。阅读主程序，第 5 空调用 factory 对象的方法，而类 Factory 类只有一个方法为 getShapelInstance，所以第 5 空应填入 getShapelInstance，参数为用户运行程序时指定的参数，程序中为 type 参数，表明需要生成哪一种类型的对象。同样，因为 s 是 Shape* 类型，所以，getShapelInstaace(type) 的返回值类型为 Shape*，因此第 4 空应填入 Shape*；程序退出前需要释放指针 s 所占用的内存空间，所以第 6 空应填写 delete s。

试题六参考答案

(1) implement Shape

(2) XCircle

(3) Xcircle()

(4) displayIt()

(5) Shape

(6) getShapelInstance (type)

试题六分析

题目中明确要求 Circle 具有 Shape 接口，所以，第 1 空应填上 implemems Shape。因为要重用 XCircle 类而不用从头开发一个新的 Circle 类，所以，凡是 Circle 类实现 Shape 的接口时都调用相应的 XCircle 类提供的方法。因此第 2 空应填上 Xcircle，第 3 空应填上 XcircleO 生成相应的 XCircle 的一个实例，第 4 空调用 displaylt()来实现 Shape 接口中的 display 方法。阅读主程序，第 6 空调用 factory 对象的方法，而类 Factory 类只有一个方法为 getShapelInstance，所以第 6 空为应填入 getShapeInstance(type)，参数为用户运行程序时指定的参数，程序中为 type 参数，表明需要生成哪一种类型的对象。同样，因为 s 是 Shape 类型的引用，所以，getShapelInstance 的返回值类型为 Shape，因此第 5 空应填入 Shape。

试题七答案:

- (1) Implements Ishape
- (2) Xcircle
- (3) New Xcircle
- (4) Get IShape_Color
- (5) LetIShape__Color
- (6) bridged.displayIt

试题七分析

在 VB 中, 抽象类模块只包含属性和方法的定义, 而不包括任何可执行代码。本题目中定义的抽象类 Ishape 中定义了一个属性 color、两个方法 draw 和 move。

在实现界面时, 需要在有 Ishape 界面的类模块的说明段中添加一个 Implements 键字, 因此空(1)处应填入“Implements IShape”。根据试题中的说明部分可知, 类 CCircle 中要使用 XCircle 提供的显示功能, 需声明 XCircle 的实例变量并生成和引用实例对象, 因此空(2)处应填入“XCircle”, 空(3)处填入“New XCircle”, 空(6)处填入“bridged. displayIt”。由“color=newColor”可知, 空(5)处所在的 Property 过程使 CCircle 的私有数据成员 color 取得外部设置的 newColor 值, 因此空(5)填入“LetIShape__Color 同理由“IShape_Color=color”可知, 空(4)处所在的 Property 过程使 IShape_Color 取得 CCircle 的私有数据成员 color 的值, 因此空(4)填入“Get IShape_Color”。