3.1 给出下面术语的定义：关系模式、关系数据库模式、域、关系实例、关系的基以及关系的度。
答：关系模式：关系模式对表（关系实例）的每个列进行描述，关系模式需要给出关系名、每个字段的名称，以及每个字段的域。
关系数据库模式：数据库中关系模式的集合。
域：字段的数据类型。
关系实例：关系中元组或记录的集合。
关系的基：该关系实例具有记录的数目。
关系的度：字段的数目。

3.2 如果一个关系实例的基为 22,那么该关系实例中有多少个不同的记录？
答：如果一个关系实例的基为 22,那么该关系实例中有 22 个不同的记录。

3.3 从编写 SQL 查询的角度来看,关系模型是否能够实现物理和逻辑数据的独立性？请解释。
答：进行 SQL 查询时，用户并不知道数据在底层的物理存储，所以可以实现物理数据的独立性；而用户可以通过定义视图隐藏数据在概念模式中的修改，所以可以实现逻辑数据的独立性。

3.4 关系的候选码和主码有什么区别？什么是超码？
答:候选码是指能够惟一确定每条记录的关系的字段的集合，主码是由数据库管理员从候选码中确定的一个码，候选码有多个，主码只有一个。超码是包含主码或候选码的字段集合。

3.5 对于图 3.1 给出的关系 Students 的实例：
（1）如果该实例是合法的，根据表中数据给出属性（或者属性集合）不是候选码的实例。
（2）如果该实例是合法的，根据表中数据能确定某属性（或者属性集合）是候选码吗？
答：（1）{name},{age}；
（2）不能，关系的候选码不能由有限基数的关系实例给出。

3.6 什么是外码约束？这种约束为什么很重要？什么是参照完整性？
答：有时存储在某个关系中的信息会和存储在其他关系中的信息发生关联，当修改其中一个关系的数据时，需要检查其他的关系，必要时还需要进行修改以保持数据的一致性，因此，我们需要事先指定关系间的完整性约束，使得 DBMS 能够自动进行检查，涉及两个关系的最普通的完整性约束是外码约束。
外码约束保证了数据库中数据的一致性。
参照完整性即参照的关系中的属性值必须能够在被参照关系找到或者取空值。

3.7 对于 1.5.2 节给出的关系 Students、Faculty、Course、Rooms、Enrolled、Teaches 和 Meets_In。
（1）列出这些关系中具有的所有外码约束。
（2）给出涉及这些关系的一个既不是主码约束又不是外码约束的约束实例。
答：(1)Enrolled 的 sid 和 cid 有外码约束,Teachers 的 fid 和 cid 有外码约束,Meets_In 的 cid 和 rno 有外码约束。
（2）对 sid、cid 和 fid 的名称标记或长度进行标准化。

3.14 考虑练习 2.4 的公司数据库以及设计出来的 ER 图，给出创建相应关系的 SQL 语句，并尽可能地定义需要的约束。如果不能定义某些约束，说明为什么？

答：

```
CREATE TABLE Employees (ssn CHAR(10),
                        sal INTEGER,
                        phone CHAR(13),
                        PRIMARY KEY (ssn))

CREATE TABLE Departments (  dno INTEGER,
                            budget INTEGER,
                            dname CHAR(20),
                            PRIMARY KEY (dno) )

CREATE TABLE Works in ( ssn CHAR(10),
                        dno INTEGER,
                        PRIMARY KEY (ssn, dno),
                        FOREIGN KEY (ssn) REFERENCES Employees,
                        FOREIGN KEY (dno) REFERENCES Departments )
CREATE TABLE Manages (  ssn CHAR(10),
                        dno INTEGER,
                        PRIMARY KEY (dno),
                        FOREIGN KEY (ssn) REFERENCES Employees,
                        FOREIGN KEY (dno) REFERENCES Departments )

CREATE TABLE Dependents (   ssn CHAR(10),
                            name CHAR(10),
                            age INTEGER,
                            PRIMARY KEY (ssn, name),
                            FOREIGN KEY (ssn) REFERENCES Employees,
                                    ON DELETE CASCADE )
```