1. 关系代数 Relational Algebra
   1. 一元运算符有选择select、投影project、重命名remane，其余的包括联合union、差积set diffetence、笛卡尔积cartesian product等为二元运算符。下面学习它们的基本使用。
      1. 选择  
         选择的操作符为小写sigma(σ)，比如选择物理系的教师，可以写为：  
         σdept\_name=”Physics”(instructor)（标准的写法中dept\_name=”Physics”属于σ的下标）。  
         还可以使用比较运算、与∧、或∨、非¬等，薪水大于90000的物理系教师：  
         σdept\_name=”Physics”∧salary>90000(instructor)  
         查询部门名称与所在办公楼同名的部分：  
         σdept\_name=building(department)  
         需要注意的是关系代数中的select并不与SQL中的select对应，与之对应的是SQL的where子句
      2. 投影  
         投影的操作符为大写pi（Π），可以选择性的显示关系的某些属性，SQL的select ID, name, salary from instructor 可以表示为：  
         ΠID, name, salary (instructor )
      3. 关系运算的组合  
         关系运算的结果还是关系，基于这一点，可以将关系运算组合使用，比如查找表达物理系老师的姓名，可以表示为：  
         Πname (σdept name = “Physics” (instructor))  
         这儿将选择运算的结果作为了投影运算的参数
      4. 并运算Union  
         并运算为二元运算符，用于合并两个相同结构的集合，之前查询过2009年秋季和2010年春季都开课的课程ID，可以表示为：  
         Πcourse id (semester = “Fall”∧ year=2009 (section))   
         ∪  
         Πcourse id (semester = “Spring”∧ year=2010 (section))  
         union运算的符号为∪，参加union运算的关系必须有相同数目的属性，而且对于合并位的属性类型必须相同。
      5. 集合差运算Set-Difference  
         集合差运算的符号位减号-，r-s的结果为存在于r中但不存在于s中的元组。集合差运算对集合的要求与union相同。
      6. 笛卡尔积Cartesian Product  
         笛卡尔积的符号为×，instructor×teaches的结果为教师与课程间所有可能的的组合情况，但这样往往没什么实用意义，除了教师与课程能对应起来的元组，这些元组可以筛选为：  
         σinstructor .ID = teaches.instcuctor\_ID (dept name =”Physics”(instructor × teaches))
      7. 重命名  
         重命名的符号为小写rho（ρ）：  
         ρx(E)表示将关系E重命名为x
   2. Formal definition of the relational algebra
   3. Additional附加运算  
      除了基本的关系操作，还有一些附加的操作，因为如果只用基本运算，有些表达式会比较繁琐，而附加操作是对某些基本表达的简化。
      1. 交集(Set-Intersection Operation)  
         交集运算符为∩，用于找出几个集合共有的数据，比如在2009秋季和2010春季都开课的课程

(semester = “Fall” ∧ year=2009 (section)

∩

(semester = “Spring” ∧ year=2010 (section))

交集的等价表达式为：r ∩ s = r - (r - s)

* + 1. Natural-Join Operation  
       在笛卡儿积的基础上，选择共有字段相等的元祖，并去除了重复的元祖。返回结果集中字段的显示顺序为：共有字段->左侧关系的字段->右侧关系的字段

查询所有教师的名称及其授课信息的ID表达式为：  
Πname, course\_id (instructor⋈teaches)  
连接的等价表达式为：  
r⋈s = R ∪ S (r.A1 = s.A1 ∧r.A2 = s.A2 ∧ ... ∧r.An = s.An (R × S))  
如果R和S没有值相同的属性，则r⋈s = r × s.   
如果要查询所有物理系教师名称及授课课程的名称则为：  
Πname,title (dept\_name = “Comp. Sci.” (instructor⋈teaches⋈course))

不管是(instructor⋈teaches)⋈course还是instructor⋈(teaches⋈course)，运算结果不受连接的顺序的影响

* + - 1. Theta join  
         θ-joint是笛卡尔积与选择的结合
    1. 赋值运算Assignment Operation  
       符号为←，与其他程序语言的赋值运算一样，←可以将一段表达式的值赋值给某个临时变量，避免嵌套太多难以理解。比如r⋈s的等价表达式也可以写为：  
       temp1←R × S  
       temps←σr.A1 = s.A1 ∧r.A2 = s.A2 ∧ ... ∧r.An = s.An (temp1)  
       result=ΠR ∪ S(temp2)  
       ←可以将复杂的逻辑简化为过程式的代码，而且←必须赋值给临时变量。
    2. 外连接Outer join  
       外连接是连接运算的扩展，可以用来处理缺失的信息，既自然连接中丢失的未匹配行；其中left outer join ⟕、right outer join ⟖、full outer join ⟗ 分别保留左侧、右侧、全部的未匹配行。  
       left outer join 的等价表达式为：  
       (r⋈s) ∪(r-ΠR(r⋈s))×{(null,...null)}   
       其中{(null,...null)}的模式是S-R后的schema
    3. 扩展运算
       1. 广义投影(Generalized Projection)  
          广义投影允许在投影运算的同时进行算术、字符串处理的操作，比如：  
          Πid,name,salary\*13(instructor)
       2. 聚集(aggregation)  
          聚集包括min、max、average、count等操作，聚集运算的输入为值的集合，输出为单一的值。教师的平均薪水可以表示为：  
          Gsum(salary)(instructor)。正确的G应该是calligraphic字体。  
          统计2010年春季的授课教师数量时，需要去重：  
          Gcount-distinct(ID)(σsemester=”spring”∧year=2010(teaches))  
          统计各部门的薪水平均值时，需要按部门分组，写为：  
          dept\_name G averge(salary)(instructor)