题型：

选择题20分

填空10空，10分

名词解释2个，10分

简答题2个，10分

编程题10个（7个SQL语句，3个关系代数表达式），30分

综合体4问，20分

1. 绪论
   1. 数据库的4个基本概念
      * 1. 数据，信息的载体，数据不一定有价值，其中蕴含的信息有价值；
        2. 数据库，长期存储、有组织并且可共享的大量数据集合；
        3. 数据库管理系统，用户和操作系统之间的管理软件，和操作系统一样是计算机的基础软件，也是一个大型复杂的软件系统。用于科学的组织和存储数据，高效地获取和维护数据。

主要功能：（1）数据定义（DDL）（2）数据组织、存储和管理（3）数据操纵功能（DML）（4）数据库的事务管理和运行管理（5）数据库的建立和维护功能；

* + - 1. 数据库系统，由数据库、数据库管理系统（及其应用开发工具）、应用程序和数据库管理人员（DBA）组成的存储、管理、处理和维护数据的系统。
  1. 数据库发展的三个阶段：人工管理阶段，文件系统阶段和数据库系统阶段（最重要）。
  2. 数据库相较于人工管理和文件系统的优点：
     + 1. 数据整体结构化，这是与文件系统的本质区别。在文件系统中也有数据结构化，但是文件系统实现的记录内的结构化，但是记录之间的关系需要程序员去维护，数据库中的数据结构不再仅限某一应用，而面向整个企业或组织，数据之间都是有联系的（数据库针对集合，文件系统针对记录）；
       2. 数据的共享性高、冗余度低且易扩充：数据共享可以大大减少数据冗余，节约存储空间，还能避免数据之间的不相容性与不一致性；
       3. 数据独立性高，分为物理独立性和逻辑独立性分别保证应用程序与数据库的物理结和逻辑结构相互独立，由数据库管理系统的两级映像来保证；
       4. 数据有限数据库管理系统统一管理和控制：（1）数据安全性（防止非法访问操作致使数据泄露或破坏）（2）数据完整性（保证数据正确。有效和相容）（3）并发控制（4）数据库恢复。
  3. 数据模型的组成要素（数据结构、数据操作数据完整性约束）
     + 1. 数据结构描述数据库的组成对象以及对象之间的联系，对象类型的集合，对系统静态特性的描述；
       2. 数据操作是指对数据库中各种对象（型）的实例（值）允许执行的操作的集合，包括操作及有关的操作规则，对系统动态特性的描述；
       3. 数据的完整性约束条件是一组完整性规则。
  4. 数据库系统的三级模式
     + 1. 模式也称逻辑模式，是数据库中全体数据的逻辑结构的描述，是所有用户的公共数据视图。一个数据库只有一个模式。
       2. 外模式也称子模式或用户模式，它是数据库用户能够看见和使用的局部数据的逻辑结构和特征的描述，是数据库用户的数据视图，是与某一应用有关的数据逻辑表示。
       3. 内模式也称存储模式，一个数据库只有一个内模式。他是数据物理结构和存储方式的描述，是数据在数据库内部的组织方式。
  5. 数据库系统的二级映像功能与数据独立性
     + 1. 外模式/模式映像，当模式改变时，由数据库管理员对各个外模式/模式映像作相应改变，可以使外模式保持不变。应用程序是依据数据的外模式编写的，从而应用程序不必修改，保证了数据与程序的逻辑独立性，简称数据的物理独立性。
       2. 模式/内模式映像，当数据库的存储结构改变时，由数据库管理员对各个模式/内模式映像作相应改变，可以使模式保持不变，从而应用程序不必修改，保证了数据与程序的物理独立性，简称数据的物理独立性。

1. 关系数据库
   1. 关系模式（最后一个综合大题，需要写出关系模式）

关系的描述称为关系模式，是型，关系是值。关系模式可以表示为R(U,D,DOM,F)，R为关系名。通常也简记为R(U)。例如：学生（学号，姓名，性别，出生日期，所在系）其中主码用下划线标识，外码用波浪线标识。

* 1. 实体完整性：关系模型的完整性规则是对关系的某种约束条件（UNIQUE，CHECK约束等）

关系模型中有三类完整性约束：实体完整性、参照完整性和用户定义完整性。其中实体完整性规定关系必须有码且非空。

* 1. 专门的关系运算（选择、投影、连接和除运算，除不考）
     + 1. 选择又称限制，在关系R种选择满足给定条件的诸元组，记作σF(R)={t|t∈R^F(t)=真}
       2. 投影是一种无条件查询，从关系R中选择出若干属性列组成新的关系。ΠA(t[A]|t∈R)
       3. 连接也称θ连接，是从两个关系的笛卡尔积中选取属性间满足一定条件的元组。其中θ为=的连接称为等值连接。自然连接要求两个关系中进行比较的分量必须是同名的属性组，结果中把重复的属性去掉。

并、差、笛卡尔积、选择、投影这五种运算为基本的运算。关系代数中，这些运算经有限次复合后形成的表达式称为关系代数表达式。

1. 关系数据库标准语言SQL
   1. SQL的特点
      * 1. 综合统一：SQL集数据定义语言DDL（CREATE，DROP，ALTER）、数据操作语言DML（INSERT，UPDATE，DELETE）、数据控制语言DCL（GRANT，REVOKE）于一体；
        2. 高度非过程化：只要提出做什么，而无需指明怎么做，无需了解存取路径，这些都由系统自动完成；
        3. 面向集合的操作方式；
        4. 以同一种语法结构提供多种使用方式：SQL既是独立的语言，有事嵌入式语言；
        5. 语言简洁，易学易用。
   2. 基本表的定义、删除和修改
      * 1. 定义基本表及约束，基本格式如下：

CREATE TABLE <表名> (<列名><数据类型> [列级完整性约束条件]

[,<列名><数据类型> [列级完整性约束条件]

……

[,<表级完整性约束>]);

(常用约束有，PRIMARY KEY，UNIQUE，FOREIGN KEY...REFERENCES <表名>(属性或属性组)，NOT NULL等，以上约束可以在前加CONSTRAINT来为定义的约束进行命名)

* + - 1. 修改基本表

ALTER TABLE <表名>

[ADD [COLUMN] <新列名><数据类型> [完整性约束]]

[ADD<表级完整性约束>]

[DROP [COLUMN] <列名> [CASCADE|RESTRICT]]

[DROP CONSTRAINT <完整性约束名> [RESTRICT|CASCADE]]

[ALTER COLUMN <列名><数据类型>];

无论基本表中原来是否已有数据，新增加的列一律为空值。

* + - 1. 删除基本表

DROP TABLE <表名> [RESTRICT|CASCEDE];

* 1. 数据查询（如果结合难以实现那么就分步做）
     + 1. 一般格式：

SELECT [ALL|DISTINCT] <目标列表达式> [,<目标列表达式>]...

FROM <表名或视图名> [,<表名或视图名>]|(<SELECT语句>)[AS] <别名>

[WHERE<条件表达式>]

[GROUP BY <列名1> [HAVING <条件表达式>]]

[ORDER BY <列名2> [ASC(升序)|DESC(降序)]];

SELECT子句的<列目标表达式>不仅可以使表中的属性列，也可以是表达式，还可以是字符串常量，函数等。可以通过指定别名来改变查询结果的列标题。

* + - 1. WHERE子句

字符匹配用LIKE，LIKE适用于所有情况，但是=只能用于已经确定的字符。LIKE可以用于不确定的字符，\_代表一个任意单个字符，%代表任意长度任意字符，ESCAPE 后跟一个字符代表这个字符是换码字符，例如ESCAPE '\'。检验空值用IS [NOT]NULL，不能用等号或者其他字符。确定范围的[NOT]BETWEEN AND包含边界。

* + - 1. ORDER BY子句

ASC升序，DESC降序。对于空值，排序是现实的次序由具体系统实现来决定。

* + - 1. 聚集函数

COUNT(\*|[DISTINCT|ALL] <列名>) 统计元组|一列中值得个数

SUM([DISTINCT|ALL] <列名>) 计算一列值的总和

AVG([DISTINCT|ALL] <列名>) 计算一列值的平均值（同SUM，此列必须是数值型）

MAX([DISTINCT|ALL] <列名>) 求一列值中的最大值

MIN([DISTINCT|ALL] <列名>) 求一列值中的最小值

当聚集函数遇到空值时，除COUNT(\*)外，都跳过空值而只处理非空值。聚集函数只能用于SELECT子句和HAVING短语中。

* + - 1. GROUP BY子句会将属性列值相等的元组作为一组，通常会在每组中作用聚集函数，若其后还带有HAVING短语，则只有满足指定条件的组才予以输出。目的是为了细化聚集函数的作用对象，若无分组聚集函数将作用于整个查询结果，分组后聚集函数将作用于每一个组而不是整个查询结果，即每一组都有一个函数值。
      2. 连接查询，其中比较常用的是自然连接，一般在WHERE子句中写入，一般格式如下：

[<表名1>.] <列名1><比较运算符> [<表名2>.]<列名2>

[<表名1>.] <列名1>BETWEEN [<表名2>.]<列名2> AND [<表名3>.]<列名3>

如果属性名在参加连接的各表中是唯一的，则可以省略表名前缀。

也可用INNER JOIN，例如SELECT...FROM Stu INNER JOIN SC ON（条件表达式）

* + - 1. 嵌套查询

子查询的SELECT语句中不能使用ORDER BY子句。一般使用IN谓词。

* 1. 视图
     + 1. 建立视图，一般格式为：

CREATE VIEW<视图名> [(<列名> [,<列名>]...)]

AS <子查询>

[WITH CHECK OPTION]（对视图进行操作时要保证操作的行满足视图定义中的谓词条件）

* + - 1. 查询视图，首先进行有效性检查，然后转化为等价的对基本表的查询，最后执行。这个过程叫做视图消解。
      2. 需要区分的是派生表，派生表是在查询语句执行的时候生成的临时表，语句执行完后立马被删除，而视图只是存储了查询语句，在查询的时候调用。派生表用法如下：

SELECT ...

FROM （SELECT ...

FROM ...

[GROUP BY]）[AS] <表名>(属性)

WHERE ...

* + - 1. 视图的作用：（简答题）
         1. 视图能够简化用户的操作；
         2. 视图使用户能以多种角度看待同一数据；
         3. 视图对重构数据库提供了一定程度的逻辑性；
         4. 视图能够对机密数据提供安全保护；
         5. 适当利用视图可以更清晰地表达查询。

1. 数据库安全性
   1. 存取控制（主要包括定义用户权限和合法权限检查两部分）
      * 1. 首先了解一个概念，数据字典。数据字典是关系数据库管理系统内部的一组系统表，其中记录了所有定义信息。
        2. 分类，自主存取控制和强制存取控制。
           1. 自主存取控制（DAC），不同的用户对于不同的数据库对象具有不同的存取权限，用户还可以将权限转授给其他用户，十分灵活。收回一个角色权限时，该用户授予其他用户的有关权限也会被收回。
           2. 强制存取控制（MAC），每一个数据库对象被标以一定的密级，每一个用户可以被授予某一个级别的许可证。只有具有合法许可证的用户才可以存取。
   2. 自主存取控制方法
      * 1. 用户权限由两个要素组成：数据库对象和操作类型。其中存取控制的对象不仅有数据本身（SELECT，INSERT等），还有数据库模式（CREATE TABLE等）。拥有权限在对表进行操作时也需要遵守相应的完整性约束。
   3. 授权：授予和收回（GRANT和REVOKE）
      * 1. GRANT语句一般格式如下：（不能循环授权）

GRANT <权限>[,<权限>]...

ON <对象类型><对象名>[,<对象类型><对象名>]...

TO <用户>[,<用户>]...

[WITH GRANT OPTION];（被授权的用户还可以把该权限赋予其他人）

* + - 1. REVOKE语句一般格式如下：

REVOKE <权限>[,<权限>]...

ON <对象类型><对象名>[,<对象类型><对象名>]...

FROM <用户>[,<用户>]...[CASCADE|RESTRICT]（级联|拒绝）

* + - 1. CREATE USER <username> [WITH] [DBA|RESOURCE|CONNECT];（只有超级用户才有权创建）
         1. 默认为CONNECT，不能创建用户，模式，基本表，只能登陆数据库；
         2. RESOURCE，能创建基本表和视图，不能创建模式和用户；
         3. DBA，超级用户。
      2. 数据库角色，是被命名的一组与数据库操作相关的权限，角色是权限的集合。
         1. 创建，CREATE ROLE <角色名>
         2. 授权，GRANT <权限>[,<权限>]... ON <对象类型>对象名 TO <角色>[,<角色>]...
         3. 将角色授予角色或用户

GRANT <角色1>[,<角色2>]...

TO <角色3>[,<用户1>]...

[WITH ADMIN OPTION]

* + - * 1. 权限收回（只是角色的权限，已经授予的用户或其他角色不能收回）

REVOKE <权限>[,<权限>]...

ON <对象类型><对象名>

FROM <角色>[,<角色>]...

* 1. 强制存取控制
     + 1. 主题是系统中的活动实体，包括实际用户和各进程；客体是系统中的被动实体，是受主体操纵的，包括文件、基本表等。数据库管理系统为主体和客体每一个实例都指派一个敏感度标记，主体的敏感度标记称为许可证级别，客体的敏感度标记称为密级。
          1. 仅当主体的许可证级别大于或等于客体的密级时，该主体才能读取相应的客体；
          2. 仅当主体的许可证级别小于或等于客体的密级时，该主体才能写相应的客体。

强制存取控制是对数据本身进行密级标记，无论数据如何复制，标记与数据是不可分的整体。对通过自主存取控制检查的允许存取的数据库对象再由系统自动进行强制存取控制检查。具有强制存取控制的一定也有自主存取控制，也就是说强制存取控制包含自主存取控制。

1. 数据库完整性
   1. 数据库的完整性是指数据的正确性（复合现实语义、反映当前实际状况）和相容性（数据库同一对象在不同关系表中的数据符合逻辑）。数据库管理系统必须能实现如下功能：
      * 1. 提供完整性约束条件（也称完整性规则）的机制；
        2. 提供完整性检查的方法；
        3. 进行违约处理。（NO ACTION和CASCADE）
   2. 实体完整性
      * 1. 定义实体完整性：主码不为空且唯一（PRIMARY KEY等）。
        2. 实体完整性检查和违约处理
           1. 检查主码值是否唯一，如果不唯一则拒绝插入或修改；
           2. 检查主码的各个属性是否为空，只要有一个为空就拒绝插入或修改。
   3. 参照完整性
      * 1. 定义参照完整性：FOREIGN KEY [(<列名1>)] REFERENCES <表名>(<列名2>)

参照表中的属性参照被参照表中的主属性（组）。

* + - 1. 参照完整性检查和违约处理
         1. 拒绝执行（NO ACTION）一般被设置为默认策略；
         2. 级联操作（CASCADE）：当删除或修改被参照表的一个元组导致与参照表不一致时，删除或修改参照表中的所有不一致的元组。
         3. 设置为空值：当删除或修改被参照表的一个元组造成了不一致，则将参照表中的所有造成不一致的元组对应属性设置为空值。
  1. 用户定义的完整性，针对某一具体应用的数据必须满足的予以要求。
     + 1. 属性上的约束
          1. 属性上约束条件的定义

不允许取空值（NOT NULL）②列值唯一（UNIQUE）③CHECK短语指定列值应该满足的条件

* + - * 1. 属性上约束条件的检查和违约处理：拒绝执行。
      1. 元组上的约束
         1. 元组上约束条件的定义，CHECK短语，可设置不同属性之间的取值；
         2. 元组上约束条件的检查和违约处理：拒绝执行。
  1. 完整性约束命名子句
     + 1. 完整性约束命名子句：CONSTRAINT <完整性约束条件名><完整性约束条件>，也可以在修改表的语句中定义：ALTER TABLE ADD/DROP CONSTRAINT...
  2. 断言，用以指定更具一般性的约束。
     + 1. 可以定义涉及多个表或聚集操作的比较复杂的完整性约束。任何对断言中所涉及关系的操作都会触发关系数据库管理系统对断言的检查，任何使断言不为真的操作都会被拒绝执行。
       2. 如果断言很复杂，则系统在检测和维护断言上的开销较高。
  3. 触发器，用户定义在关系表上的一类由事件驱动的特殊过程。（只考概念？差不多看看）
     + 1. 定义触发器，触发器又叫做事件-条件-动作规则。建立触发器的一般格式如下：

CREATE TRIGGER <触发器名>

{BEFORE|AFTER} <触发事件> ON <表名>

REFERENCING NEW|OLD ROW AS<变量>

FOR EACH{ROW|STATEMENT}(每行都执行一次|只执行一次)

[WHEN<触发条件>] <触发动作体>

1. 只有表的拥有者能在一个表上定义一定数量的触发器；
2. 触发器名和表名必须在统一模式下；
3. 只能定义在基本表上，但是不能定义在是视图上；
4. 触发事件可以是INSERT、DELETE或UPDATE（这些都是DML定义在数据库上，DDL是定义在服务器上的），也可以是这几个事件的组合，用OR连接。还可以是UPDATE OF <触发列,...>，AFTER/BEFORE是触发的时机。
5. 触发动作体：如果是行级触发器那么使用NEWROW或OLDROW来表示修改前后的行；如果是表级触发器可以使用OLDTABLE，OLDTABLE来表示原来的表和变化后的部分。
6. 关系数据理论
   1. 范式：符合某一种级别的关系模式的集合。一个低一级范式通过模式分解可以转换为若干个高一级范式的关系模式的集合，这种过程就叫做规范化。（最后一个大题要尽量的规范化为较高级别的范式）
   2. 第一范式1NF：每一个分量必须是不可分的数据项。
   3. 2NF：R∈1NF，且每一个非主属性完全依赖于任何一个候选码。（无部分依赖）
   4. 3NF：R∈1NF，R中不存在这样的码X，属性组Y及非主属性Z（Z⊉Y）使得X→Y，Y→Z成立。（无传递依赖）
   5. BCNF：关系模式R<U,F>∈1NF，若X→Y且Y⊈X时，X必含有码。也就是说，对于关系模式R，每一个决定因素都包含码。满足BCNF的关系模式有：
      * 1. 所有非主属性对每一个码都是完全函数依赖；
        2. 所有主属性对每一个不包含它的码也是完全函数依赖；
        3. 没有任何属性完全函数依赖于非码的任何一组属性。
7. 数据库设计
   1. 数据库设计是指对于一个给定的应用环境，构造（设计）优化的数据库逻辑模式和物理结构，并据此建立数据库及应用系统，使之能够有效地存储和管理数据，满足各种用户的应用需求，包括信息管理要求和数据操作要求。（了解定义）
      * 1. 数据库设计的基本步骤（我们只关心前四个，其中第2、3个最重要）

需求分析；概念结构设计；逻辑结构设计；物理结构设计；数据库实施；数据库运行和维护

* 1. 需求分析
     + 1. 需求分析的任务：
          1. 信息要求：致用户从数据库中获得信息的内容与性质；
          2. 处理要求：指用户要完成的数据处理功能，对处理性能的要求；
          3. 安全性与完整性。
       2. 数据字典，关于数据库中数据的描述，即元数据，而不是数据本身。数据字典是在需求分析阶段监理，在数据库设计过程中不断修改、充实、完善的。它在数据库设计中占有很重要的地位。
  2. 概念结构设计
     + 1. 概念模型，E-R图，包括实体、属性、实体之间的联系等。（必考）
       2. 实体与属性的划分原则：属性不能再具有需要描述的性质；不能与其他实体具有联系。
  3. 概念结构设计（必考）把概念结构设计阶段设计好的基本E-R图转换为选用数据库管理系统产品所支持的数据模型相符合的逻辑结构。
     + 1. E-R图向关系模型的转换，一般的一个实体型转换为一个关系模式，
          1. 一个1:1联系可以与任意一端应的关系模式合并；
          2. 一个1:n的联系可以与n端对应的关系模式合并；
          3. 一个m:n联系转换为一个关系模式；
          4. 具有相同码的关系可合并。
       2. 关系模式的优化

数据库逻辑设计的结果不是唯一的，也并不是规范化程度越高的关系就越优。可以进行必要的分解：水平和垂直分解，但是要保证分解后的关系具有无损连接性和保持函数依赖性。

* 1. 物理结构设计，为一个给定的逻辑数据模型选取一个最适合应用要求的物理结构的过程。
     + 1. 分为两步：确定数据库的物理结构；对物理结构进行评价。
       2. 关系模式存取方法选择：索引法和聚簇法（聚簇每个表只有一个）。

1. 数据库编程（只考概念）
   1. 嵌入式SQL
      * 1. 嵌入式SQL的处理过程（被嵌入的程序设计语言称为宿主语言或主语言），对于嵌入式SQL，数据库管理系统一般采用预编译方法处理，即将其转换为主语言调用语句。为了能快速区分SQL语句与主语言，所有SQL语句都必须加前缀，C语言：EXEC SQL。
        2. 嵌入式SQL语句与主语言之间的通信，主要包括：向主语言传递SQL语句的状态信息，使主语言能够据此信息控制程序流程，主要用SQL通信区；主语言向SQL语句提供参数，主要用主变量实现；将SQL语句查询数据库的结果交主语言处理，主要用主语言和游标实现。
           1. SQL缓冲区，主要包括描述系统当前工作状态和运行环境的各种参数EXEC SQL INCLUEDE SQLCA。其中有SQLCODE，用来放每次执行SQL语句后的返回值。
           2. 主变量，分为输入主变量和输出主变量。声明在BEGIN DECLARE SECTION和END DECLARE SECTION之间，SQL语句中的主变量名和指示变量前要加冒号作为标志。
           3. 游标是系统为用户开设的一个数据缓冲区，存放SQL语句的执行结果。
        3. 使用游标的SQL语句：查询结果为多条记录的SELECT语句、CURRENT形式的UPDATE和DELETE语句。
           1. 查询结果为多条记录的SELECT语句

说明游标：EXEC SQL DECLARE <游标名>CURSOR FOR <SELECT语句>;定义游标仅仅是一条说明性语句，这时的关系数据库管理系统并不执行SELECT语句。

打开游标：EXEC SQL OPEN<游标名>;可打开多次，每次都执行SQL语句。

推进游标指针并取当前记录：EXEC SQL OPEN<游标名> INTO <主变量>[<指示变量>]..;

关闭游标：EXEC SQL CLOSE <游标名>;

* + - * 1. CURRENT形式的UPDATE和DELETE语句

在UPDATE和DELETE语句中要用子句：WHERE CURRENT OF <游标名>来表示修改删除的是最近一次取出的记录，即游标指针指向的记录。当游标定义中的SELECT语句带有UNION或ORDER BY子句，或者该语句相当于定义了一个不可更新的视图时，不能使用CURRENT。

1. 关系处理和查询优化
   1. 关系数据库的查询处理
      * 1. 查询处理步骤
           1. 查询分析：对语句进行扫描、词法分析和语法分析；
           2. 查询检查，对合法的查询语句进行语义检查，即根据数据字典中有关的模式定义查询语句的数据库对象以及约束。这时的完整性检查是初步的、静态的检查。然后将SQL查询语句转换成关系代数表达式，一般都利用查询树，也称语法分析树；
           3. 查询优化（核心功能）分为代数优化和物理优化；
           4. 查询执行：由代码生成器生成执行代码然后执行返回结果。
        2. 选择操作的实现，一般情况下，当选择率较低时，基于索引的选择算法更优；选择率较高，例如查找元素均匀地分布在查找的表中时，全表扫描更优。
        3. 连接操作的实现（最耗时的操作之一）：
           1. 嵌套循环算法，最简单可行；
           2. 排序合并算法，等值连接最常用的算法，尤其是和参与连接的主表已经排好序的情况。
           3. 索引连接算法；
           4. Hash join算法：第一步：划分（创建）阶段；第二步：试探（连接）阶段。
   2. 关系数据库系统的查询优化
      * 1. 查询优化概述：查询优化的优点不仅在于用户不必考虑如何最好的表达查询以获得较高的效率，而且在于系统可以比用户程序的“优化”做的更好。优点如下：
           1. 优化器可以从数据字典中获取许多统计信息；
           2. 如果数据库的物理统计信息改变了，系统可以自动对查询进行重新优化以选择相应执行计划；
           3. 优化器可以考虑百种不同的执行计划；
           4. 优化其中包括了很复杂的优化技术。

优化的总代价=I/O代价+CPU代价+内存代价+通信代价。（集中式只有前三个，后者分布式需要考虑）计算查询代价时一般用查询处理读写的块数作为衡量单位。

* 1. 代数优化
     + 1. 关系代数表达式等价变换规则，代数优化策略是通过对关系代数表达式的等价变换来提高查询效率的：（了解，课本P283，公式真的太难写了）
          1. 连接、笛卡尔积的交换律；
          2. 连接、笛卡尔积的结合律；
          3. 投影的串接定律；
          4. 选择的串接定律；
          5. 选择与投影的交换律；
          6. 选择与笛卡尔积的交换律；
          7. 选择与并的分配率；
          8. 选择与差的分配率；
          9. 选择与自然连接的分配率；
          10. 投影与笛卡尔积的分配率；
          11. 投影与并的分配率。
       2. 查询树的启发式优化
          1. 选择运算应尽可能先做；（最重要、最基本的一条）
          2. 把投影运算和选择运算同时进行；
          3. 把投影其前或后的双目运算结合起来；（会使一些投影消失）
          4. 把某些选择同在它前面要执行的笛卡尔积结合起来称为一个连接运算；
          5. 找出公共子表达式。（把一个投影分裂为两个）
  2. 物理优化（这个老师并没有提及）选择的方法
     + 1. 基于规则的启发式优化；
       2. 基于代价估算的优化；
       3. 两者结合的优化方式。