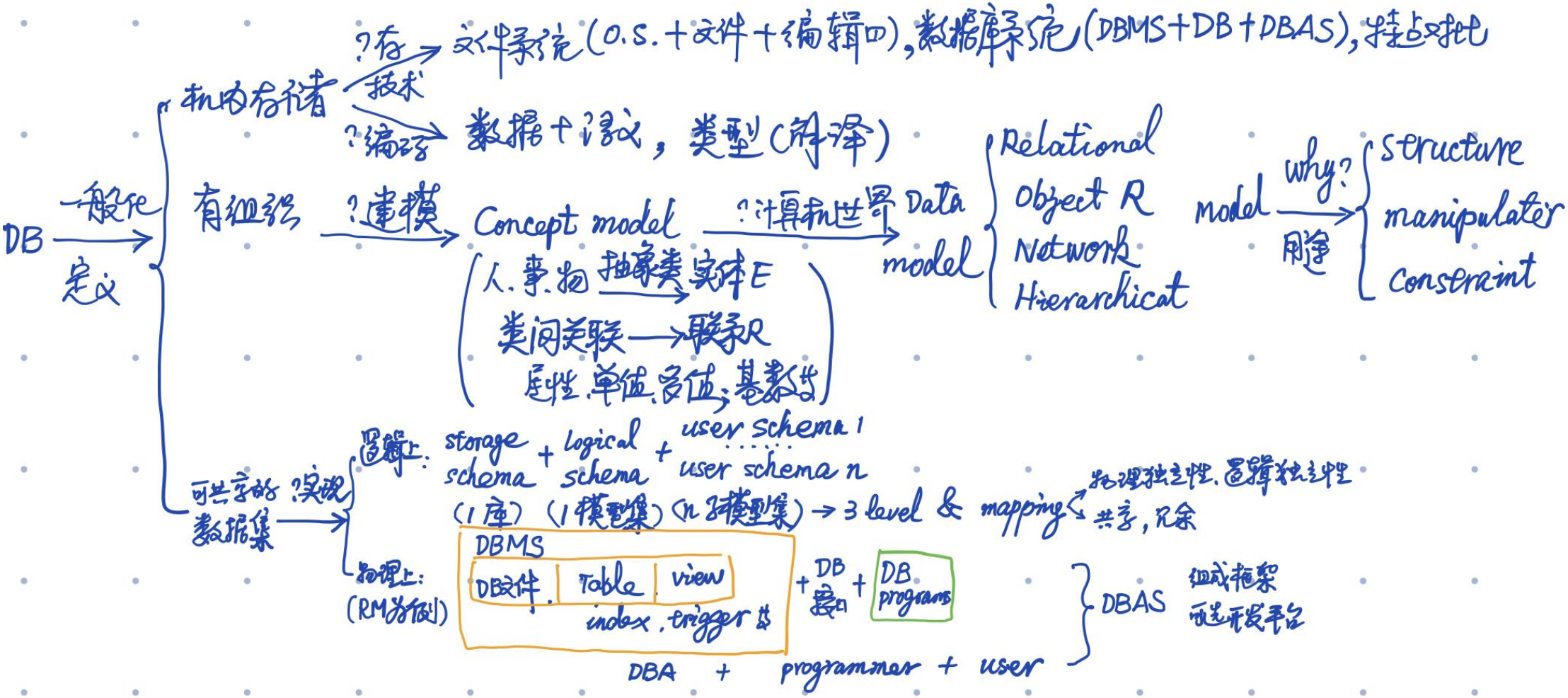
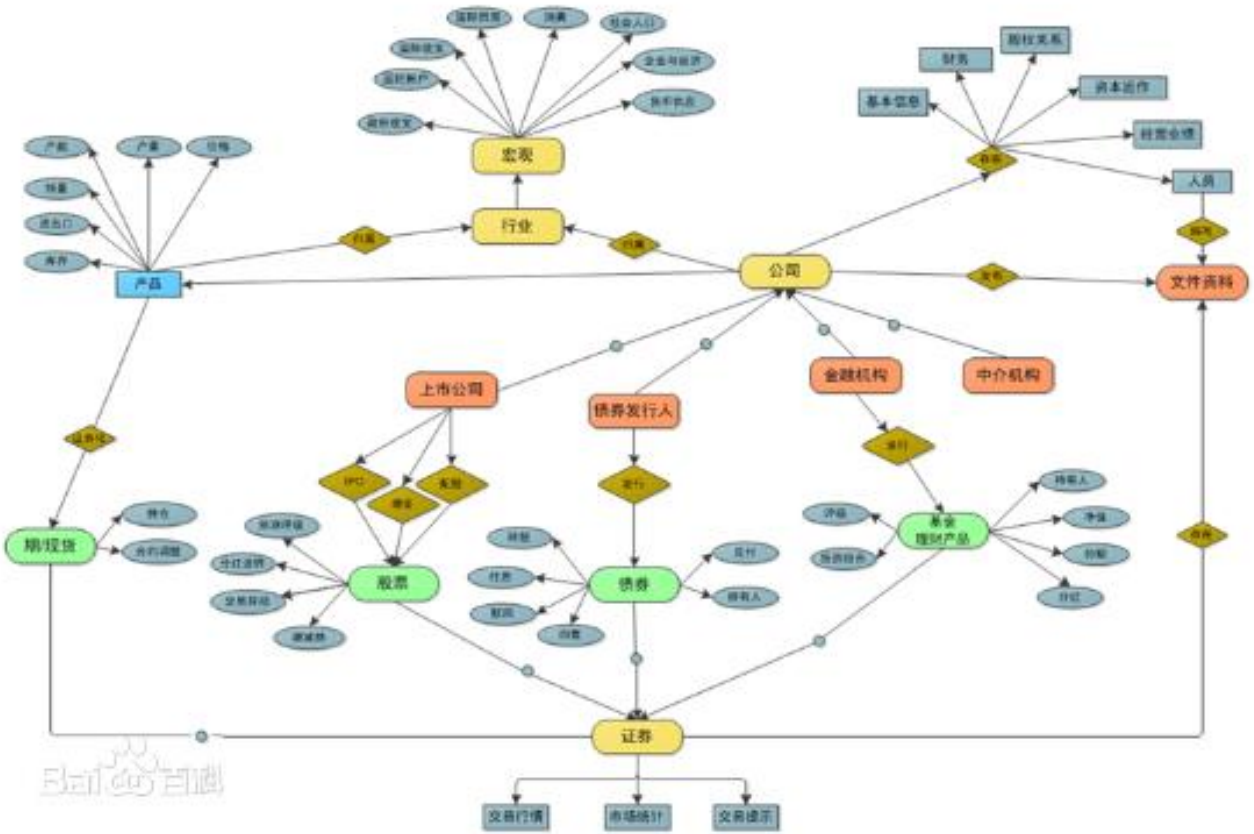
一. 数据库概念及组成：


DB
一般化定义
机内存储者
?存 技术
文件系统(O.S.+文件+编辑口),数据库系统(DBMS+DB+DBAS),特点对比
?编码
数据+说明,类型(解释)
有组织
?建模
Concept model
?计算机世界
Data model
Relational
Object R
Network
Hierarchical
model
why? 用途
structure
manipulater
constraint
人.事.物 抽象类 实体E
类间关联 → 联系R
原性.单值.多值;基本的
可共享的数据集
?实现
逻辑上: storage schema + logical schema + user schema 1 ... user schema n
(1库) (1模型集) (n子模型集) → 3 level & mapping
物理独立性.逻辑独立性
共享,冗余
物理上: (RM为例)
DBMS
DB文件 | Table | view
index, trigger等
+ DB接口 + DB programs
DBAS
组成框架
可选开发平台
DBA + programmer + user

二. 数据模型

CA隔离
- RM：
  - 一类E 转换→关系表，转换规则（结构）
  - 关系代数 交、并、差、积、除 连接等（操作使用）
  - 属性间取值以函数依赖→主属性，主键，外键，完整性（约束），范式（1NF，2NF，3NF等）
    - 实体完整性，参照……，用户自定义……
    - trigger，index
- ORM：SR区别—含子结构、含多值
  - 示例：姓名、爱好
- HM：树状结构，示例：学院、系、班、学生
- NM：网状结构，示例

概念模型：

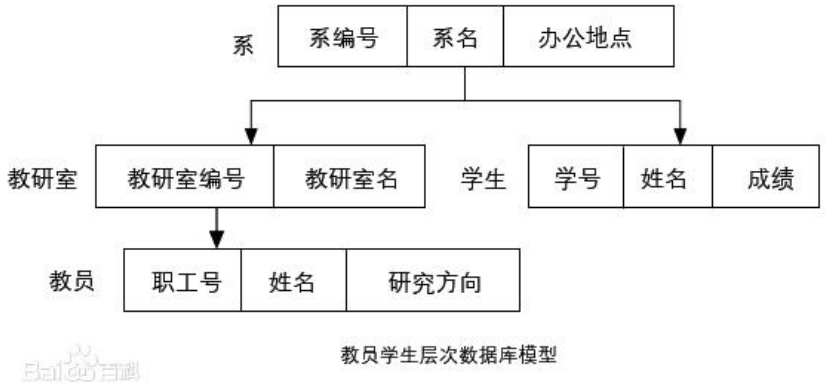


关系模型：

学号	姓名	性别	年龄	图书证号	所在系
S3001	张明	男	22	B20050101	外语
S3002	李静	女	21	B20050102	外语
S4001	赵丽	女	21	B20050301	管理

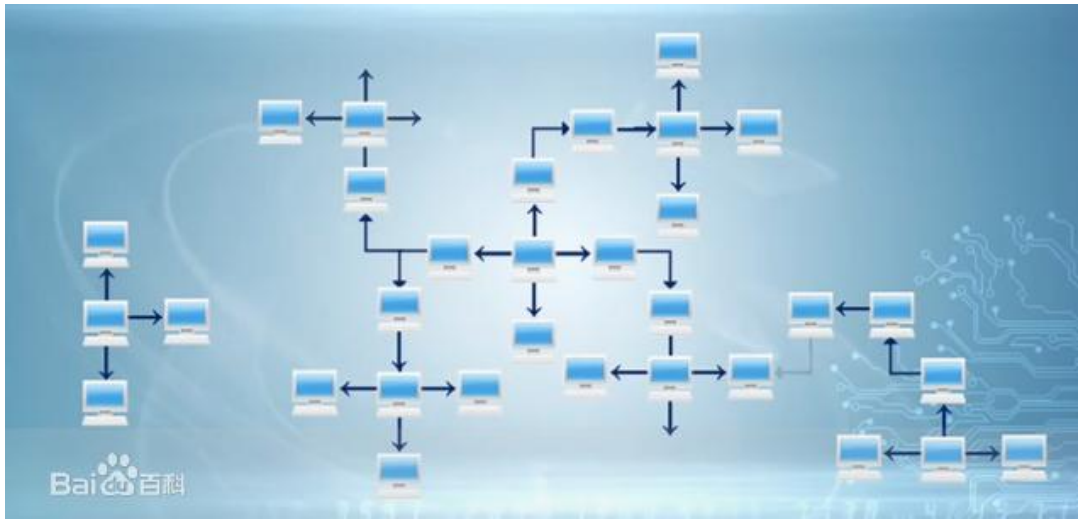
学号	课程号
S3001	C1
S3001	C2
S3002	C1
S4001	C3

层次模型：

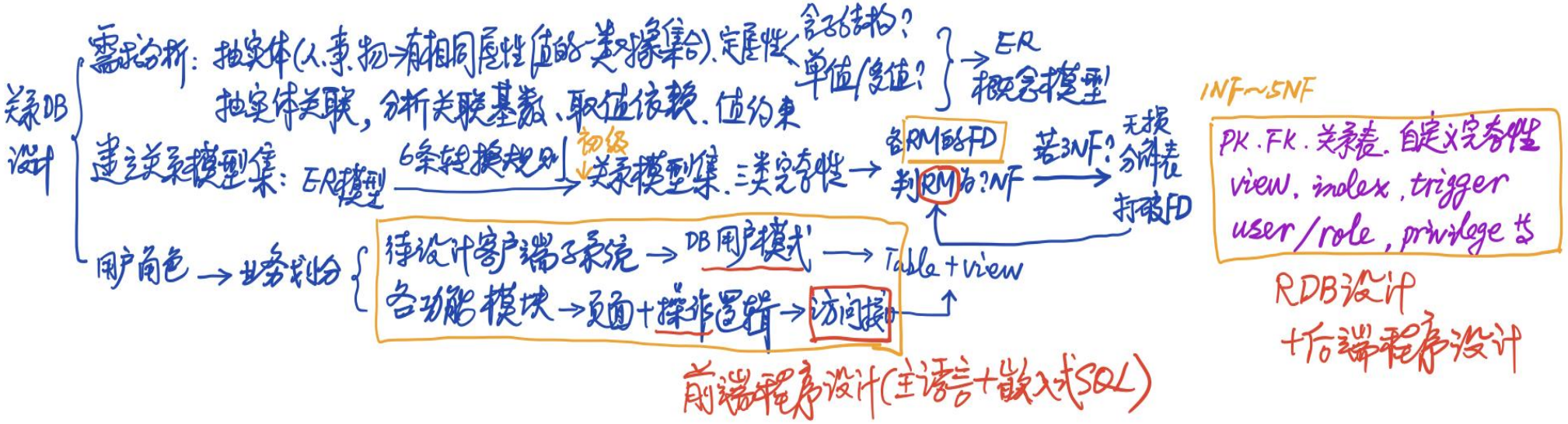


教员学生层次数据库模型

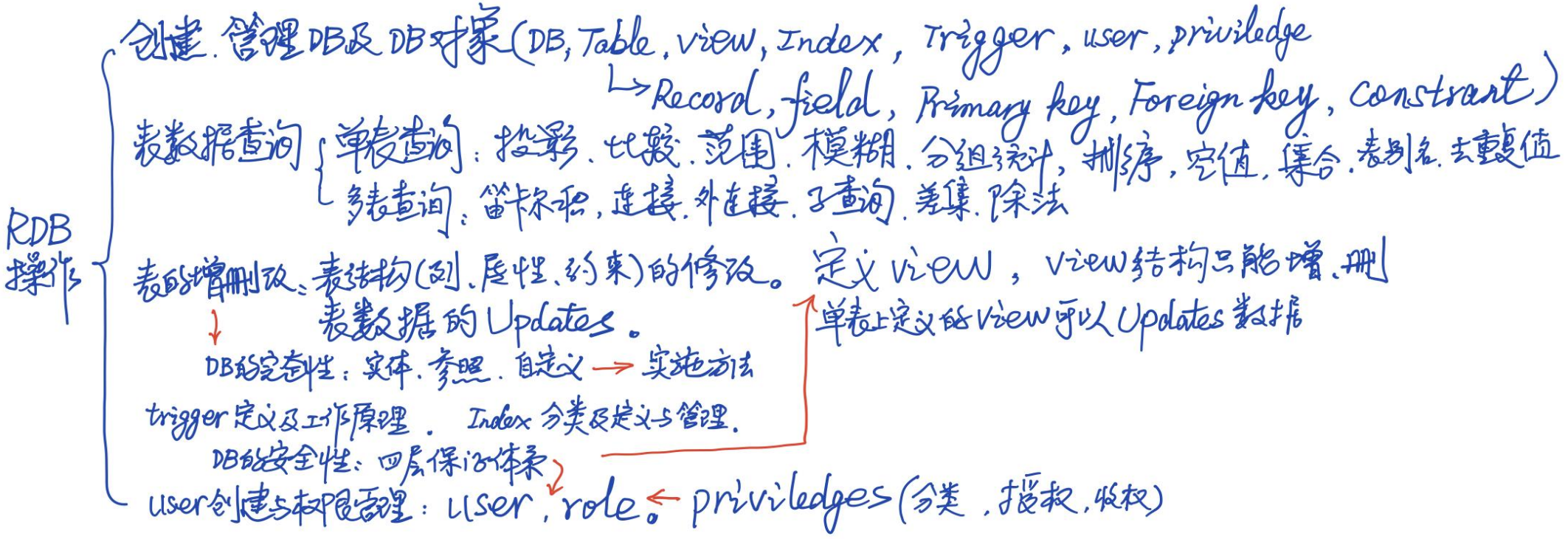
网状模型：



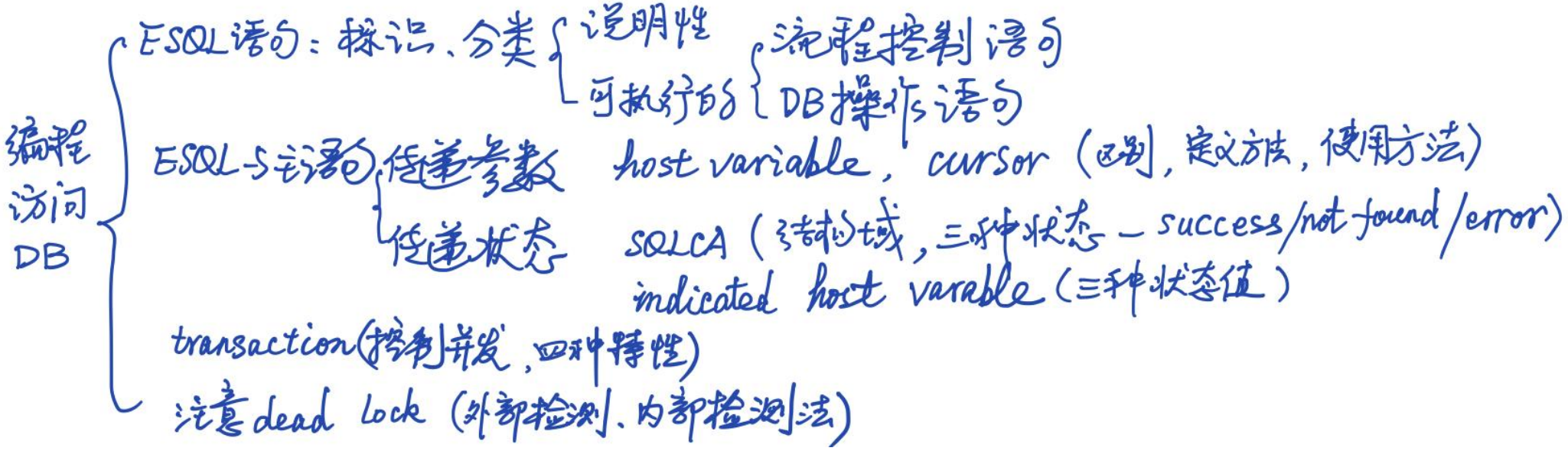
三. 关系数据库设计：



四. 关系数据库(直接)操作：



五. 关系数据库编程访问：

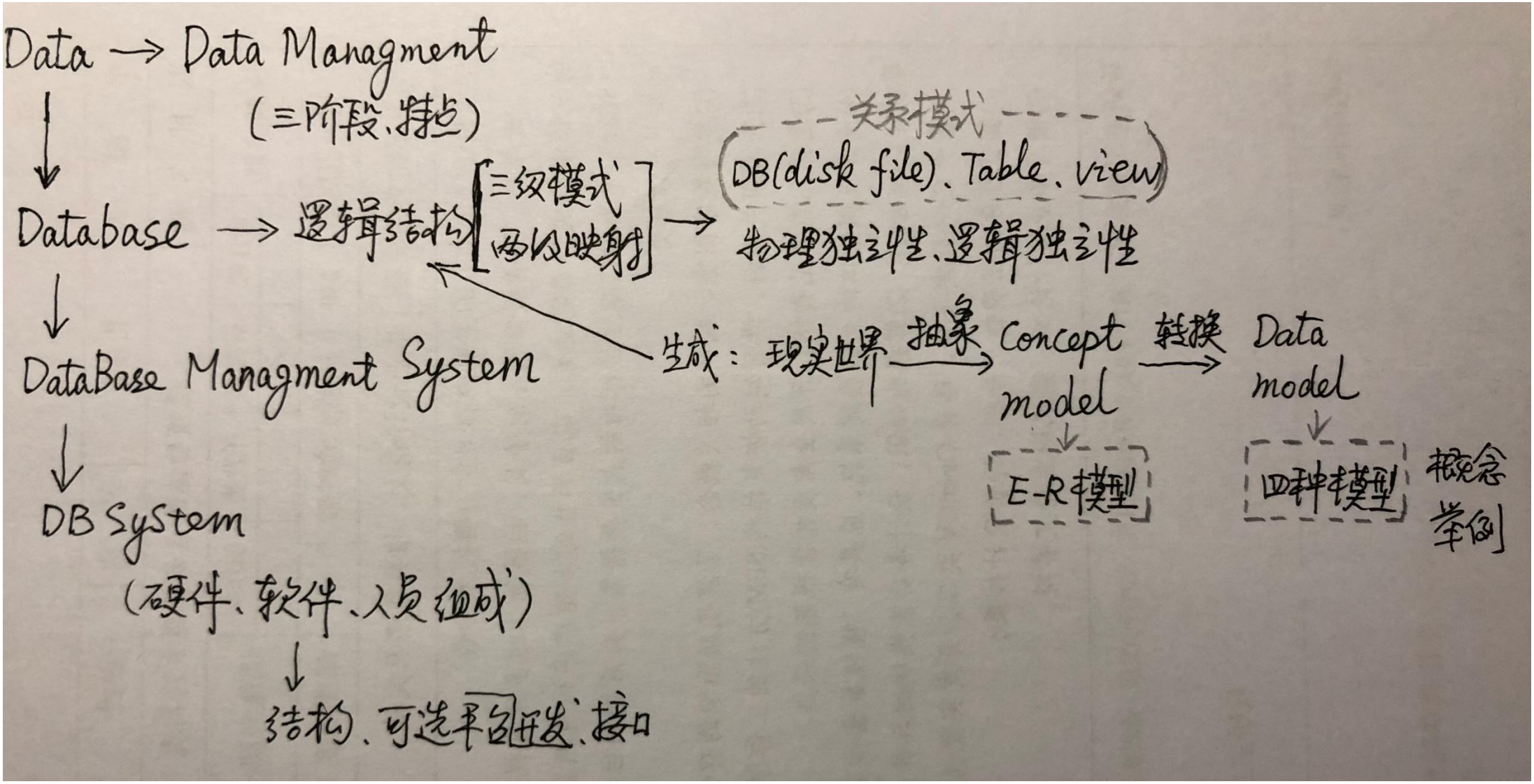


考察设计能力

- 概念理解、框架认知
- 数据库设计（需求案例分析、概念模型建立；关系模型设计与优化，物理实现）
- 数据库直接使用与管理

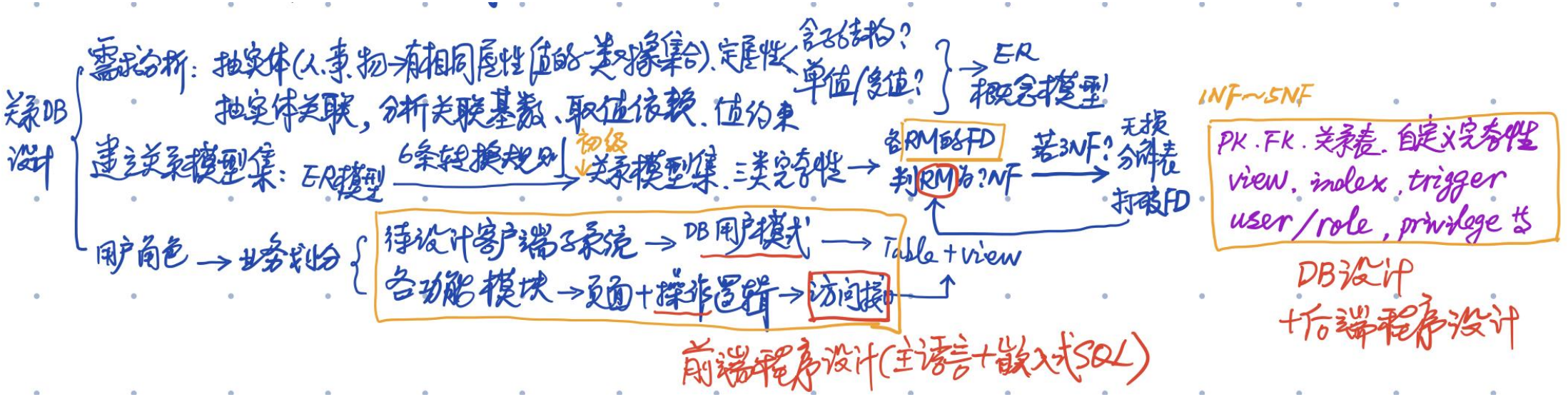
  关系代数下的：关系模型定义，完整性约束定义和作用，关系代数运算
  SQL 语言下的：关系表定义，表的数据操作(查询，增删改，各种完整性约束实施，触发器，视图，权限，索引)
- 数据库编程访问（C 与 E-SQL 编程，主变量(含指示变量)，SQLCA，Cursor，并发访问 DB(事务、锁)）

Chapter 1 Introduction

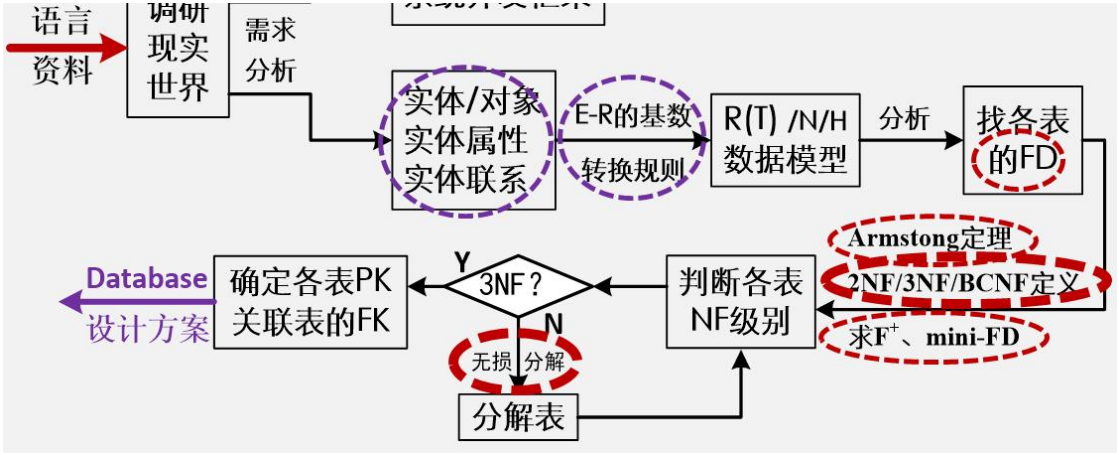


1.各项概念的理解
2.数据管理的发展历史，各阶段特点
3.DB、DBMS、DBS/DBAS 的组成框架结构，每层结构的作用 （包括人员）
4.设计数据库的步骤与方法，其中形成的各种模型 （与第 **6** 章联合复习）

chapter 6 Database Design（关系模式设计+数据库应用程序设计）



chapter 6 Database Design



1.完整软件(含数据库)应用系统的设计步骤、各步骤的设计内容；

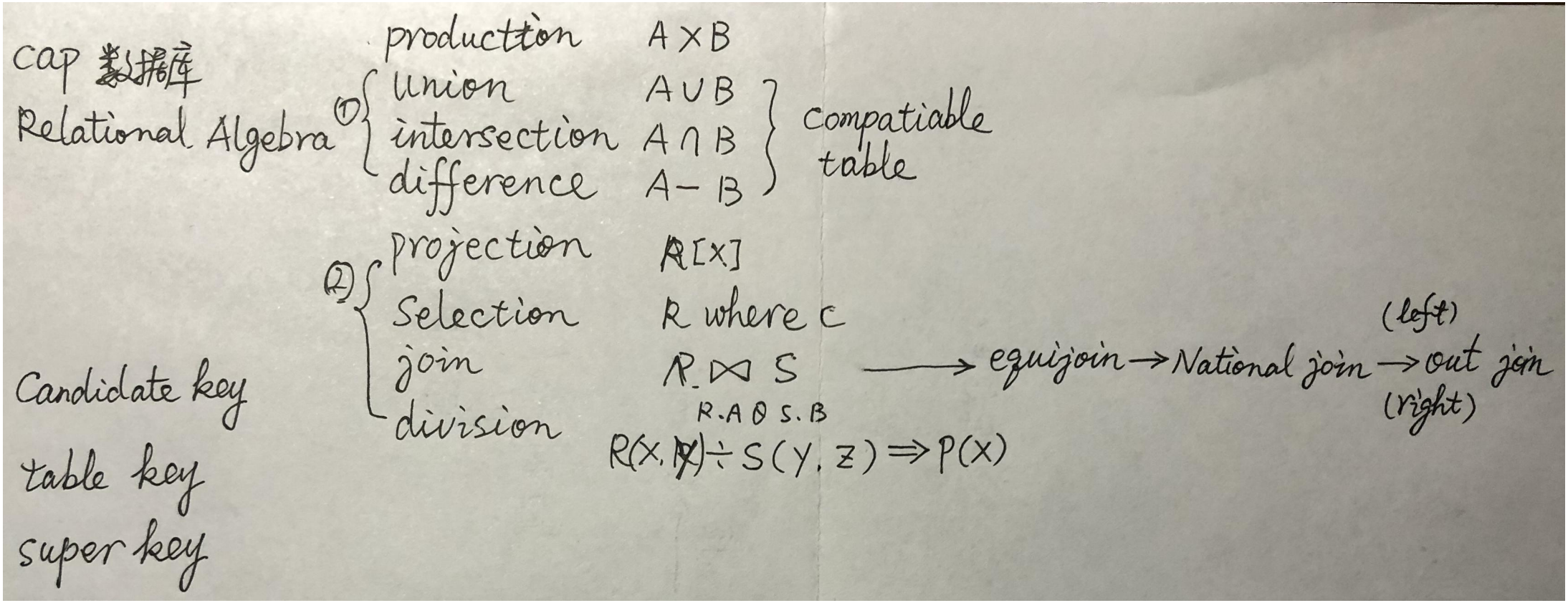
2.重点厘清“数据(库)结构”的设计步骤、各步骤设计内容，掌握其中的专有名词、模型、定义、规则

现实世界调研、需求分析 → **Concept model** (Entity-Relationship model) →→→ **Data Model**

Network model, Heretical model, Object-Relational model

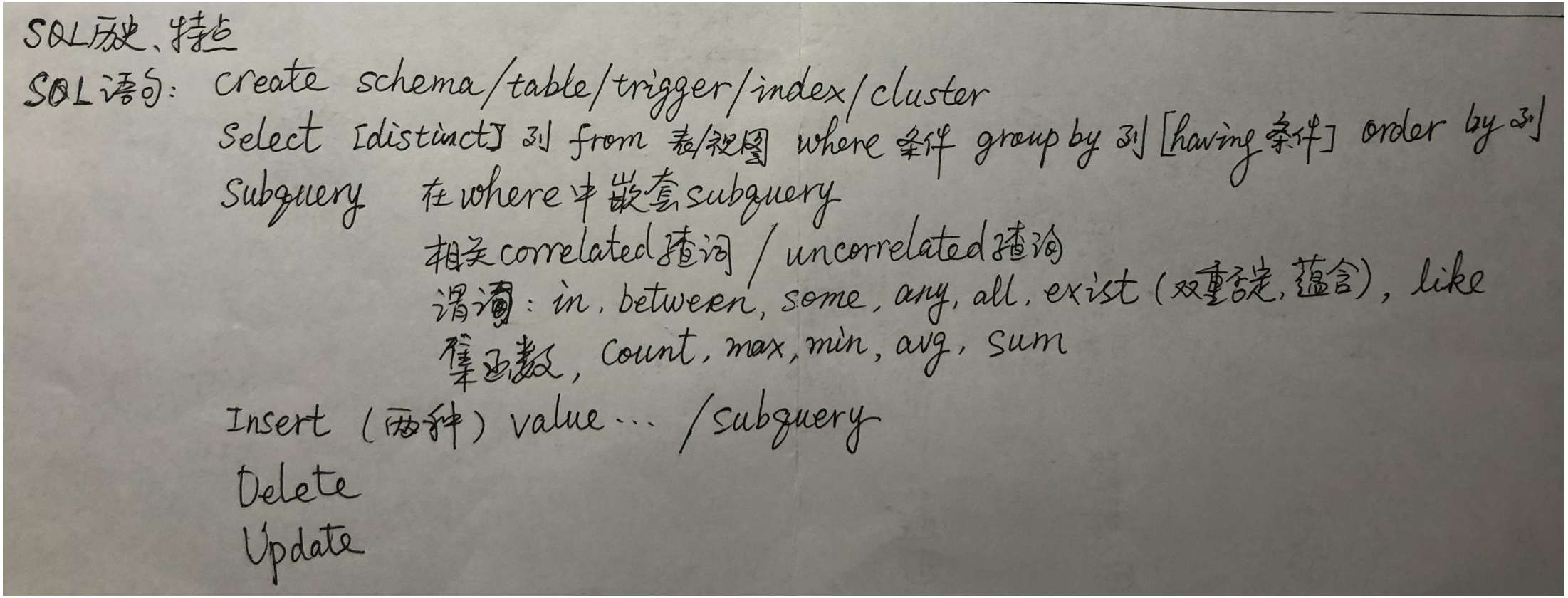
(Transform rules) →→→ **Relational model** →→→ ..... →→→ 数据(库)结构

chapter 2　Relational Model



1.关系(数据)模型的产生、作用、描述方法、模型结构，理解各结构部分的作用
2.与关系模型有关的各项概念，理解其作用
3.关系代数的产生，运算分类，每种运算的规则和作用　　（包括外连接）
4.关系数据库的最基本构成对象
5.关系数据库中，数据查询需求的代数求解方法　　（第 **2**、**3** 章习题可联合练习）

Chapter 3 Basic SQL



1.SQL 的产生、特点、语法框架、各子框架的作用

2.SQL 的两种使用方式、使用的条件框架 （与第 5 章联合复习）

3.整理常用 SQL 语句的结构清单，列出 DDL、DCL、DML，列出学过的数据库对象及其定义、使用方法 （与第 5、7 章联合复习）

DB、table、primary key、foreign key、view、index、trigger、user、role、、、、、、

Create, Alter, Drop

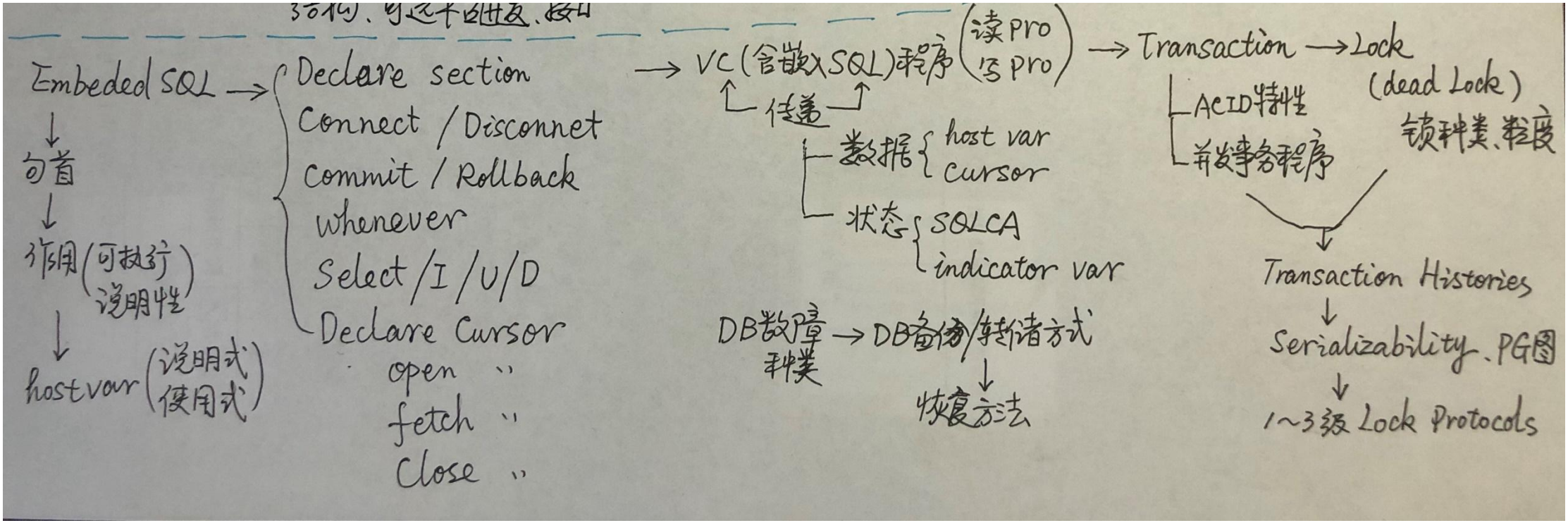
Select, group by (having), order by, select 嵌套, alias, qualifier. Insert, Update, Delete

Set function, Mathematical function, String function, Logical value

Predicates (Comparison, Between, Quantified, In, Is null, Exist, Like)

4.关系数据库中，数据查询需求的 SQL 求解方法 （第 2、3 章习题联合练习）

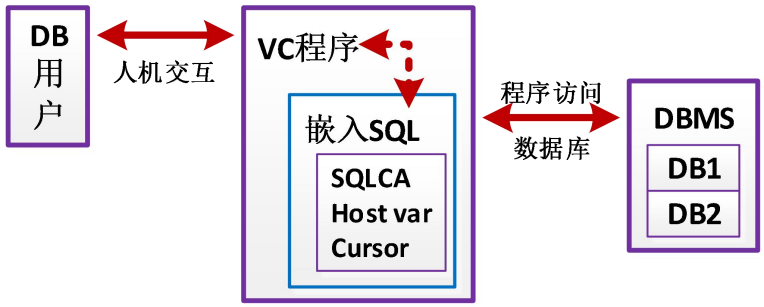
chapter 5 Programming to access DB & chapter 10 Update Transaction



1.在客户端，用高级程序设计语言编程访问数据库的工作步骤、软件开发框架

2.（客户端）数据库访问程序的开发语言、任务分工，嵌入式 SQL 语句标识

高级程序设计语言（负责与用户之间的信息交互）Embeded SQL（负责与数据库的信息交互，控制多用户程序的并发和死锁检测）



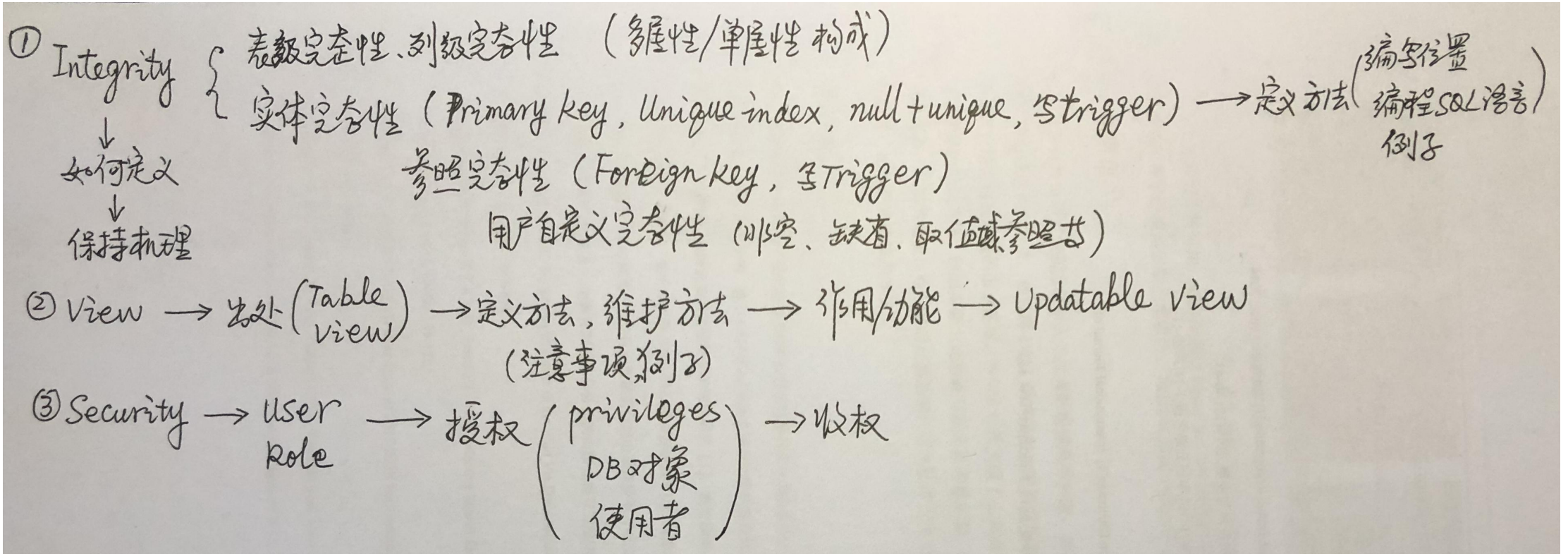
3.VC 与嵌入式 SQL 的状态、数据交换方法（SQLCA，Host variable，indicator variable，cursor）

4.常用嵌入式 SQL 语句的结构清单

5.数据库访问程序的编写案例、程序框架

6.多用户并发数据库访问程序的编写案例，Transaction 概念及并发控制原理，deadlock 概念及检测方法

chapter 7　Integrity, View, Security



1.关系 DB 中，数据完整性规则的分类角度、每种角度下的分类情况，每类 Integrity 的作用、实现的途径

(1)对象粒度 → on Table/ on Column

(2)对象多少 → 实体完整，参照完整

(3)特殊约束 → User define

2.学过的数据库安全性保障途径，每种途径下的 DB 安全性保障途径

(1)登录 → user　(a)DB account, password

(b)Create user/role,　Connect

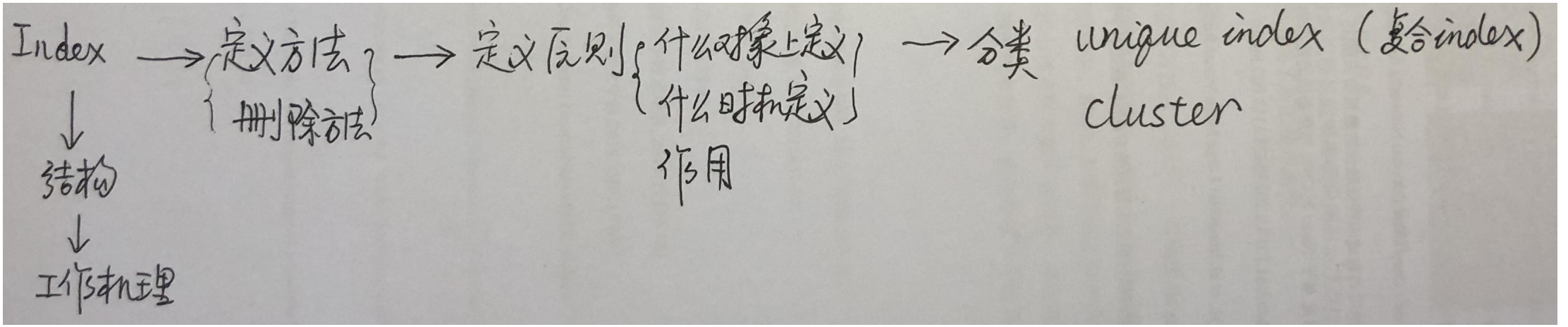
(2)使用 → Privileges　(a)权限种类、操作对象及粒度

(b) Grant, Revoke

→ View　(a)Create view，Grant, Revoke，Drop，select

(b)Update/Delete 限制

chapter 8　Index



1.关系 DB 中，Index 的分类角度、每种类别 index 的作用

(1)Index，数据库系统中的实现框架、工作机理

(2)Unique index，

(3)Cluster，

2.几种 index 定义方法

数据库系统——第九讲 嵌入式 SQL 语言之基本技巧

一．嵌入式 SQL 语言概述

重点与难点：数据库语言嵌入到高级语言中使用需要解决的问题、过程及其思维

（1）交互式 SQL 语言的局限

交互式 SQL 语言有很多优点：

- 记录集合操作
- 非过程性操作：指出要做什么，而不需指出怎样做

- 一条语句就可实现复杂查询的结果

然而，交互式 SQL 本身也有很多局限… …

- 从使用者角度：专业人员可熟练写出 SQL 语句，但大部分的普通用户…
- 从 SQL 本身角度：特别复杂的检索结果难以用一条交互式 SQL 语句完成，此时需要结合高级语言中经常出现的顺序、分支和循环结构来帮助处理

（2）嵌入式 SQL 语言

因此，高级语言+SQL 语言

- 既继承高级语言的过程控制性
- 又结合 SQL 语言的复杂结果集操作的非过程性
- 同时又为数据库操作者提供安全可靠的操作方式：通过应用程序进行操作

嵌入式 SQL 语言

- 将 SQL 语言嵌入到某一种高级语言中使用
- 这种高级语言，如 C/C++, Java, PowerBuilder 等，又称宿主语言(Host Language)
- 嵌入在宿主语言中的 SQL 与前面介绍的交互式 SQL 有一些不同的操作方式

（3） 高级语言中使用嵌入式 SQL 语言需要解决的问题

二．变量声明与数据库连接

（1）变量的声明与使用：

在嵌入式 SQL 语句中可以出现宿主语言语句所使用的变量：

```
exec sql select Sname, Sage into :vSname, :vSage from Student where Sname= :specName;
```

这些变量需要特殊的声明：

```
exec sql begin declare section;
char vSname[10], specName[10]=“张三”;
int vSage;
exec sql end declare section;
```

变量声明和赋值中，要注意：

- 宿主程序的字符串变量长度应比字符型字段的长度多 1 个。因宿主程序的字符串尾部多一个终止符为‘\0‘，而程序中用双引号来描述。
- 宿主程序变量类型与数据库字段类型之间有些是有差异的，有些 DBMS 可支持自动转换，有些不能。

（2）程序与数据库的连接与断开

三、程序与数据库的连接和断开

- 在嵌入式 SQL 程序执行之前，首先要与数据库进行连接

- 不同 DBMS，具体连接语句的语法略有差异
- SQL 标准中建议的连接语法为：

Exec sql connect to target-server as connect-name user user-name;

或　Exec sql connect to default;

- Oracle 中数据库连接:

Exec sql connect　:user_name identified by :user_pwd;

- DB2 UDB 中数据库连接:

Exec sql connect　to mydb　user :user_name using:user_pwd;

- 在嵌入式 SQL 程序执行之后，需要与数据库断开连接
- SQL 标准中建议的断开连接的语法为：

exec sql disconnect connect-name;

或　　exec sql disconnect current;

- Oracle 中断开连接:

exec sql commit release;

或　exec sql rollback release;

- DB2 UDB 中断开连接:

exec sql connect reset;

exec sql disconnect current;

（3）SQL 执行的提交与撤销

SQL 执行的提交与撤消

- SQL 语句在执行过程中，必须有提交和撤消语句才能确认其操作结果
- SQL 执行的提交：

Exec sql commit　work;

- SQL 执行的撤消**：

execsql rollback work;

- 为此，很多 DBMS 都设计了捆绑提交/撤消与断开连接在一起的语句,以保证在断开连接之前使用户确认提交或撤消先前的工作，例如 Oracle 中：

Exec sql commit release;

或　　Exec sql rollback release;

（4）事务的概念与特性

事务：(从应用程序员角度)

是一个存取或改变数据库内容的程序的一次执行，或者说一条或多条 SQL 语句的一次执行被看作一个事务。

事务一般是由应用程序员提出，因此有开始和结束, 结束前需要提交或撤消。

Begin Transaction

exec sql …

…

Exec sql …

Exec sql commit work | exec sql rollback work

End Transaction

注意：在嵌入式 SQL 程序中，任何一条数据库操纵语句(如 exec sql select 等)都会引发一个新事务的开始，只要该程序当前没有正在处理的事务。而事务的结束是需要应用程序员通过 commit 或 rollback 确认的。因此 Begin Transaction 和 End Transaction 两行语句是不需要的。

(从微观角度，或者从 DBMS 角度)

是数据库管理系统提供的控制数据操作的一种手段，通过这一手段，应用程序员将一系列的数据库操作组合在一起作为一个整体进行操作和控制，以便数据库管理系统能够提供一致性状态转换的保证。

（5）事务的概念与特性

事务的特性: ACID

- 原子性 Atomicity : DBMS 能够保证事务的一组更新操作是原子不可分的，即对 DB 而言，要么全做，要么全不做
- 一致性 Consistency: DBMS 保证事务的操作状态是正确的，符合一致性的操作规则，它是进一步由隔离性来保证的
- 隔离性 Isolation: DBMS 保证并发执行的多个事务之间互相不受影响。例如两个事务 T1 和 T2, 即使并发执行，也相当于或者先执行了 T1,再执行 T2;或者先执行了 T2, 再执行 T1。
- 持久性 Durability: DBMS 保证已提交事务的影响是持久的，被撤销事务的影响是可恢复的。

换句话说：具有 ACID 特性的若干数据库基本操作的组合体被称为事务。

数据集与游标

重点与难点：怎样在高级语言中处理数据集—游标的使用技巧

（1）如何读取单行数据和多行数据

单行结果处理与多行结果处理的差异(Into 子句与游标(Cursor))

检索单行结果，可将结果直接传送到宿主程序的变量中

检索多行结果，则需使用游标(Cursor)

（2）游标的使用概览

游标(Cursor)的使用需要先定义、再打开(执行)、接着一条接一条处理，最后再关闭

```
exec sql declare cur_student cursor for
select Sno, Sname, Sclass from Student where Sclass='035101' ;
exec sql open cur_student;
exec sql fetch cur_student into :vSno, :vSname, :vSclass;
… …
exec sql close cur_student;
```

游标可以定义一次，多次打开(多次执行)，多次关闭

三．可滚动游标与数据库的增删改

(1)可滚动游标的概念

- ODBC 支持的可滚动 Cursor

标准的游标始终是自开始向结束方向移动的，每 fetch 一次，向结束方向移动一次；一条记录只能被访问一次；再次访问该记录只能关闭游标后重新打开

- ODBC(Open DataBase Connectivity)是一种跨 DBMS 的 DB 操作平台，它在应用程序与实际的 DBMS 之间提供了一种通用接口
- 许多实际的 DBMS 并不支持可滚动游标，但通过 ODBC 可以使用该功能

(2)可滚动游标的定义和使用

- 可滚动游标是可使游标指针在记录集之间灵活移动、使每条记录可以反复被访问的一种游标
- NEXT 向结束方向移动一条；PRIOR 向开始方向移动一条；FIRST 回到第一条；LAST 移动到最后一条；ABSOLUT value_spec 定向检索指定位置的行，value_spec 由 1 至当前记录集最大值；RELATIVE value_spec 相对当前记录向前或向后移动，value_spec 为正数向结束方向移动，为负数向开始方向移动
- 可滚动游标移动时需判断是否到结束位置，或到起始位置

(3)数据的删除与更新

不需是查找删除(与交互式 DELETE 语句相同)，一种是定位删除

```
EXEC SQL DELETE FROM tablename [corr_name]
WHERE search_condition | WHERE CURRENT OF cursor_name;
```

- 一种是查找更新(与交互式 Update 语句相同)，一种是定位更新

```
EXEC SQL UPDATE tablename [corr_name]
```

SET columnname = expr [, columnname = expr …]

[WHERE search_condition] | WHERE CURRENT OF cursor_name

四．状态捕获及错误处理机制

重点与难点：错误捕获机制—设置错误陷阱与 SQLCA 的作用与使用

(1)基本机制

状态，是嵌入式 SQL 语句的执行状态，尤其指一些出错状态；有时程序需要知道这些状态并对这些状态进行处理

嵌入式 SQL 程序中，状态捕获及处理有三部分构成:

设置 SQL 通信区: 一般在嵌入式 SQL 程序的开始处便设置(exec sql include sqlca)

设置状态捕获语句: 在嵌入式 SQL 程序的任何位置都可设置；可多次设置；但有作用域(exec sql whenever sqlerror goto report_error;)

状态处理语句: 某一段程序以应对 SQL 操作的某种状态(report_error: exec sql rollback;)

状态捕获语句:exec sql whenever condition action;

Whenever 语句的作用是设置一个“条件陷阱”，该条语句会对其后面的所有由 Exec SQL 语句所引起的对数据库系统的调用自动检查它是否满足条件(由 condition 指出)

如果满足 condition，则要采取一些动作(由 action 指出)

(2)状态信息

典型 DBMS 系统记录状态信息的三种方法:

sqlcode:

典型 DBMS 都提供一个 sqlcode 变量来记录其执行 sql 语句的状态，但不同 DBMS 定义的 sqlcode 值所代表的状态意义可能是不同的，需要查阅相关的 DBMS 资料来获取其含义。

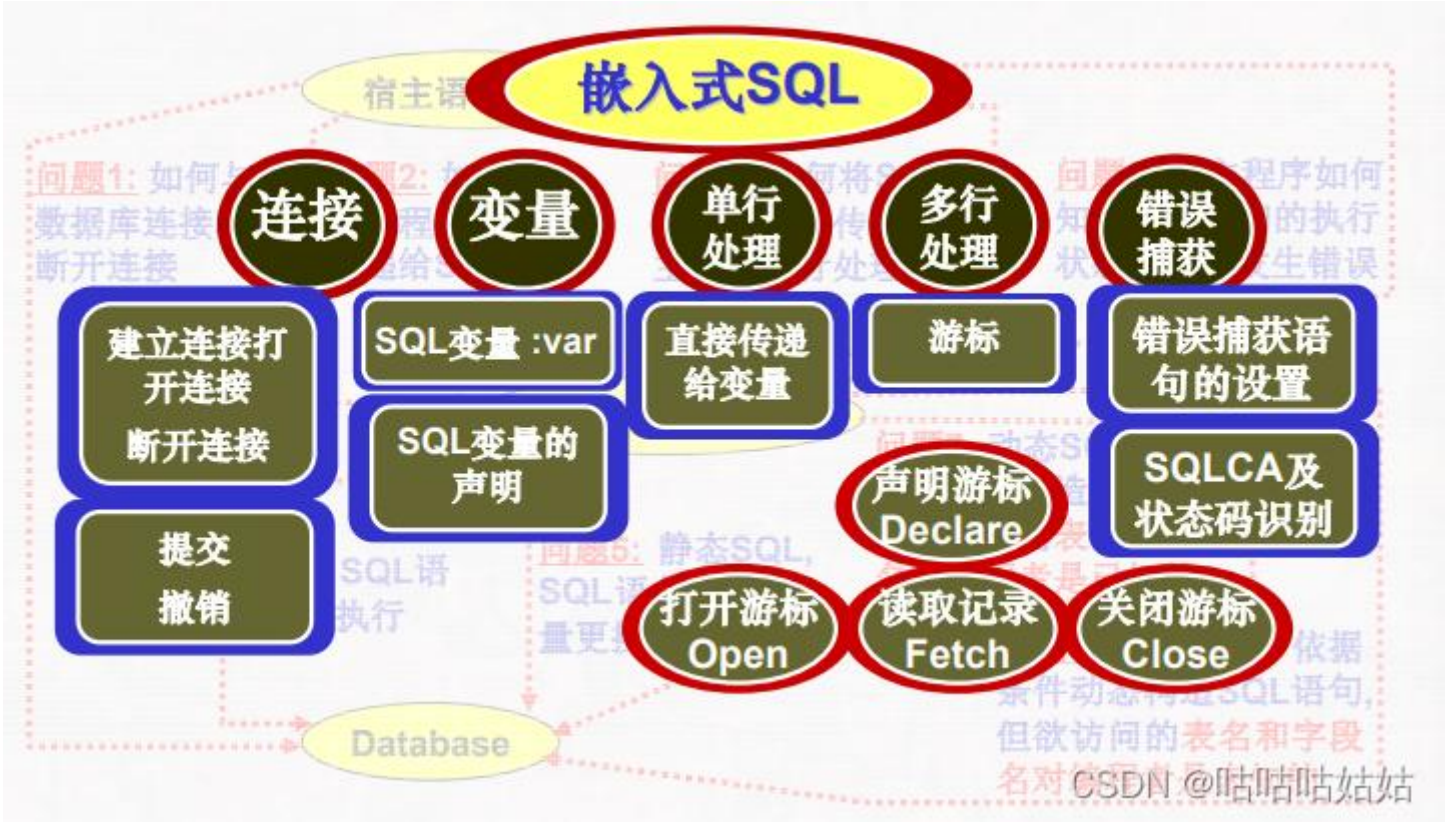
sqlca.code；：

典型 DBMS 都提供一个 sqlcode 变量来记录其执行 sql 语句的状态，但不同 DBMS 定义的 sqlcode 值所代表的状态意义可能是不同的，需要查阅相关的 DBMS 资料来获取其含义。

sqlstate:

有些 DBMS 提供的记录状态信息的变量是 sqlstate 或 sqlca.sqlstate

小结


嵌入式SQL
连接
变量
单行处理
多行处理
错误捕获
建立连接打开连接
断开连接
提交
撤销
SQL变量 :var
SQL变量的声明
直接传递给变量
游标
错误捕获语句的设置
SQLCA及状态码识别
声明游标 Declare
打开游标 Open
读取记录 Fetch
关闭游标 Close
Database
CSDN @咕咕咕姑姑