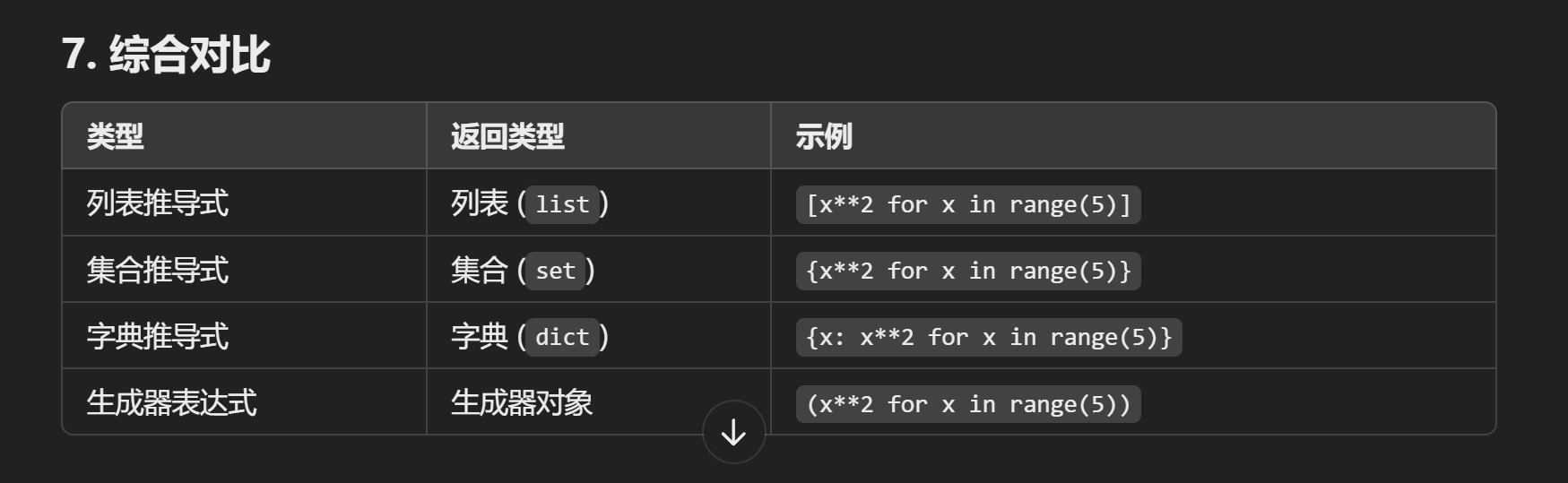
python中生成器是一种特殊的迭代器 无需手动实现\_\_iter\_\_方法

和\_\_next\_\_方法

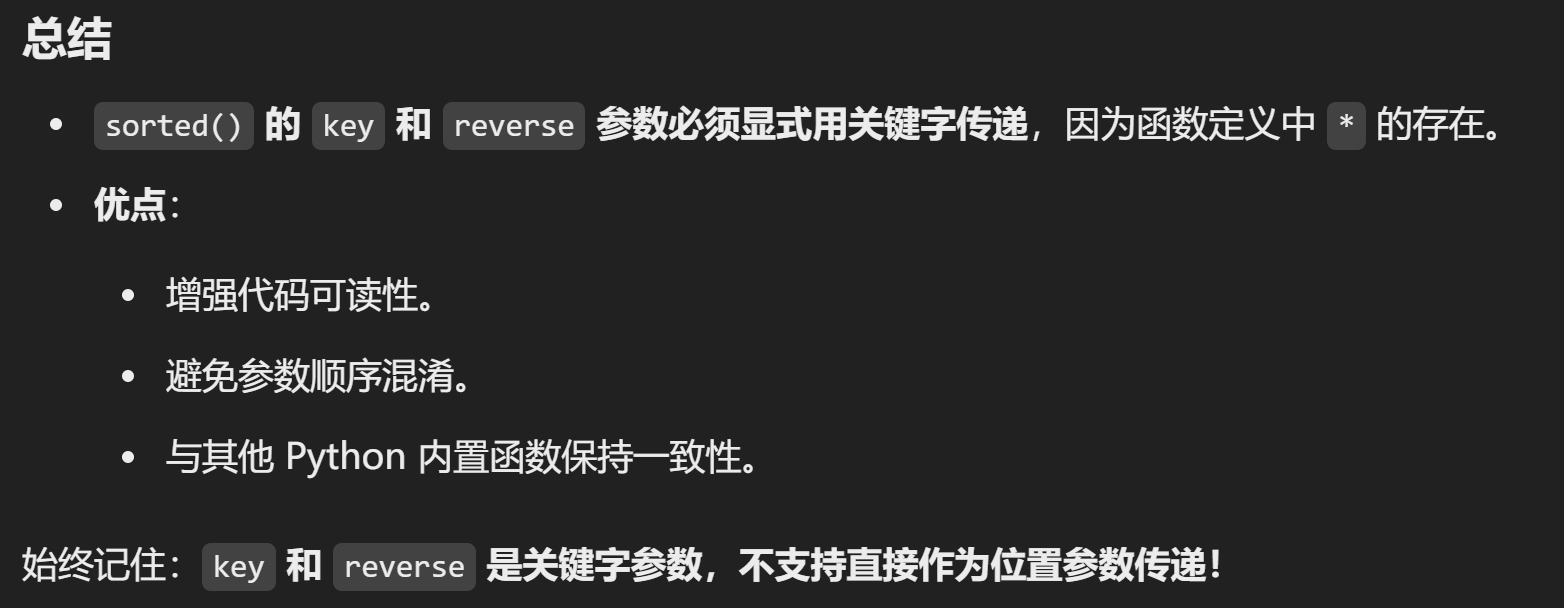
迭代器和可迭代对象有区别 可迭代对象没有\_\_next\_\_方法 必须调用\_\_iter\_\_生成一个迭代器

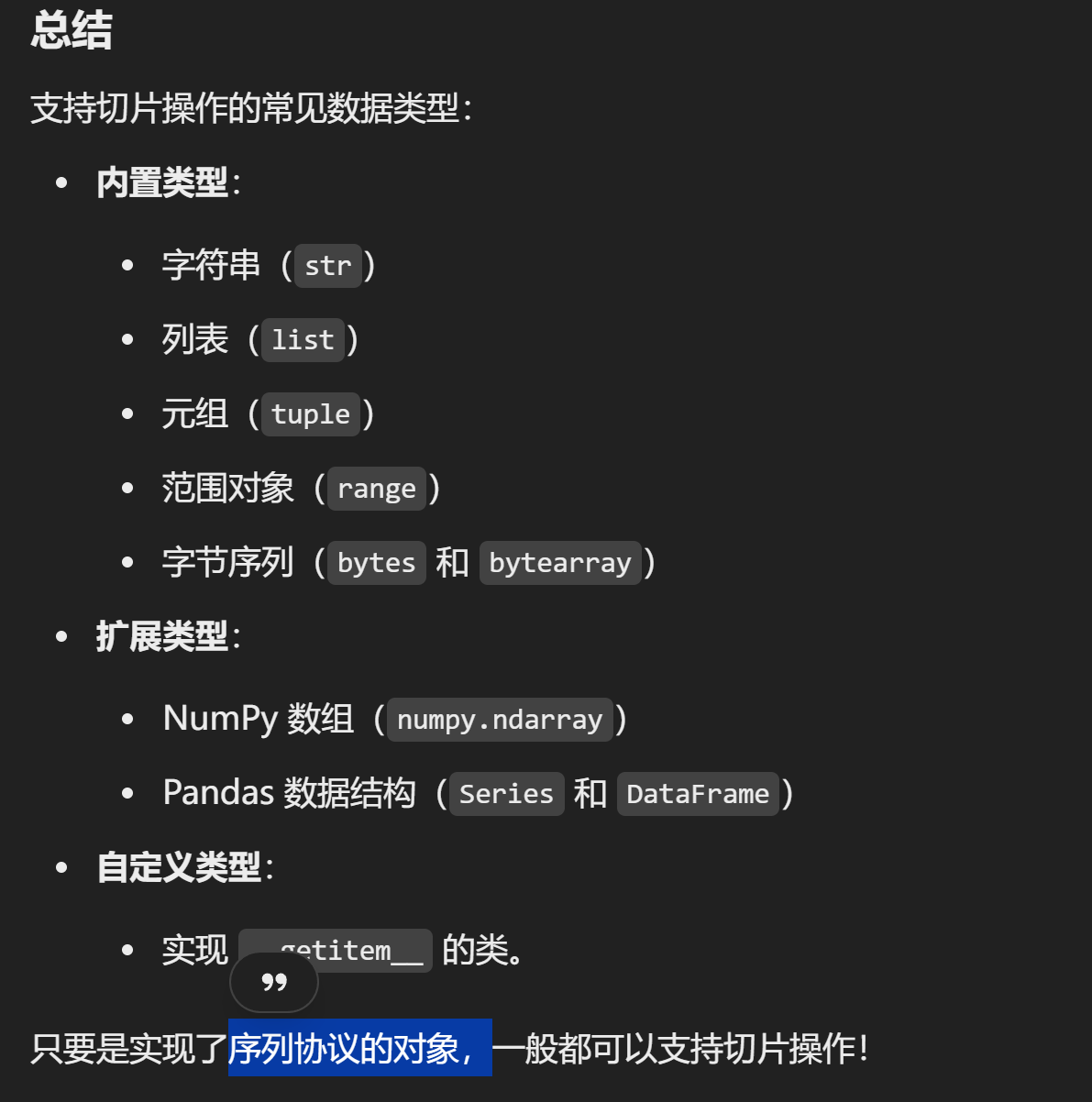






算法 很多应该是由理论和数据驱动和优化的







Crud 增删查改操作



Trigger是定义在表的操作上的

Sql语句以；为分隔符 存储过程 触发器和函数可以包含多个sql语句 所以需要先修改默认分隔符 delimiter $$

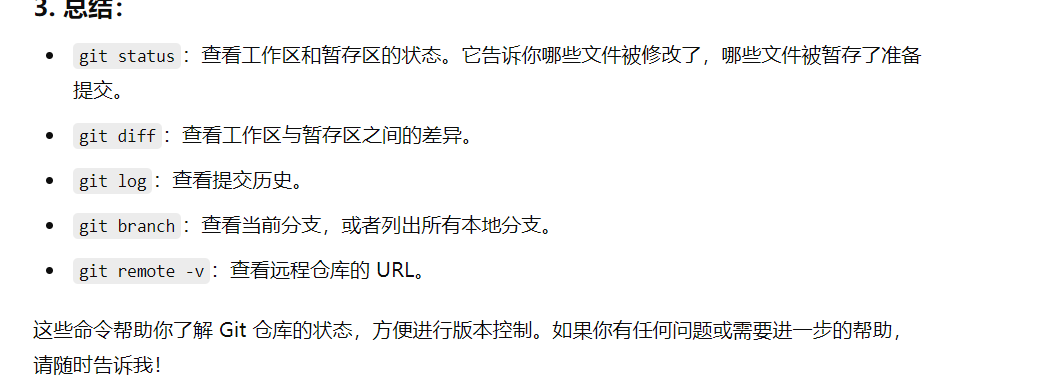
六大约束

索引约束 主键约束 非空约束 check约束 默认约束 unique约束

表 视图

函数

git与github 版本控制 暂存区 本次仓库 远程仓库 合并冲突



 存储过程



把生成的列表into到输出参数里 该参数不是一个标量

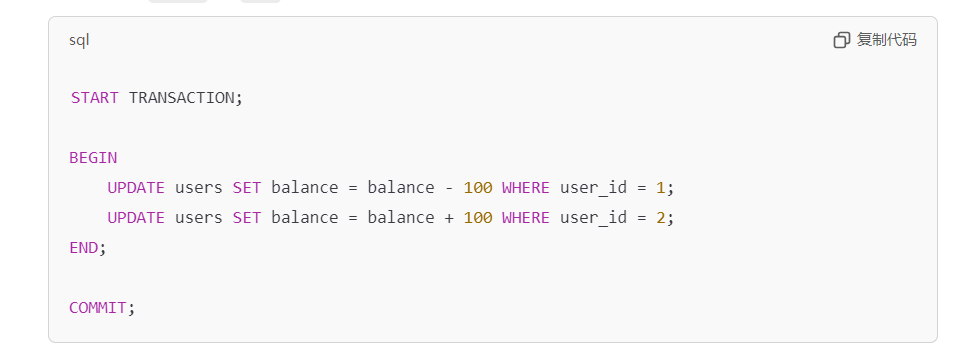




触发器

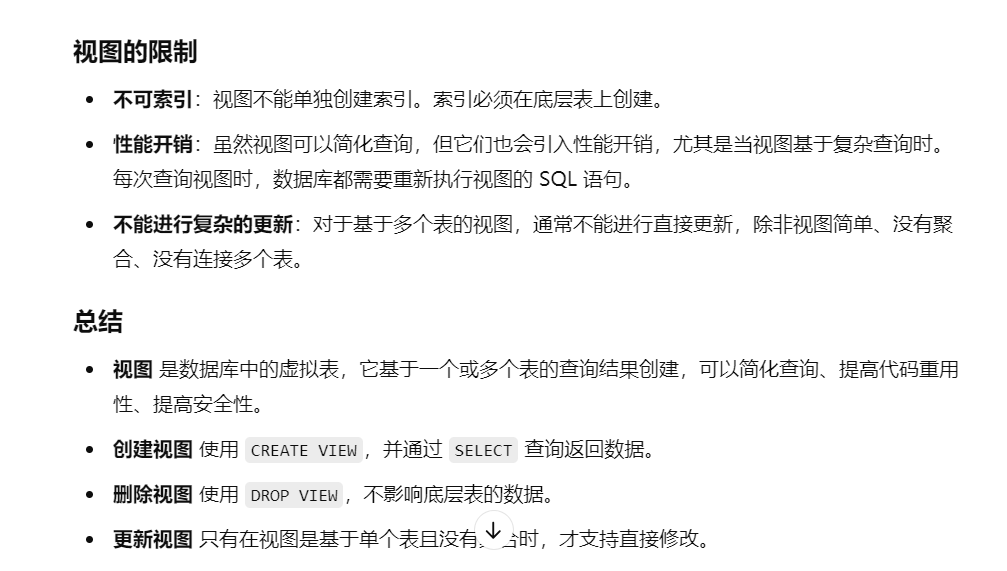
数据库的表设计 数据库的实现

事务



视图

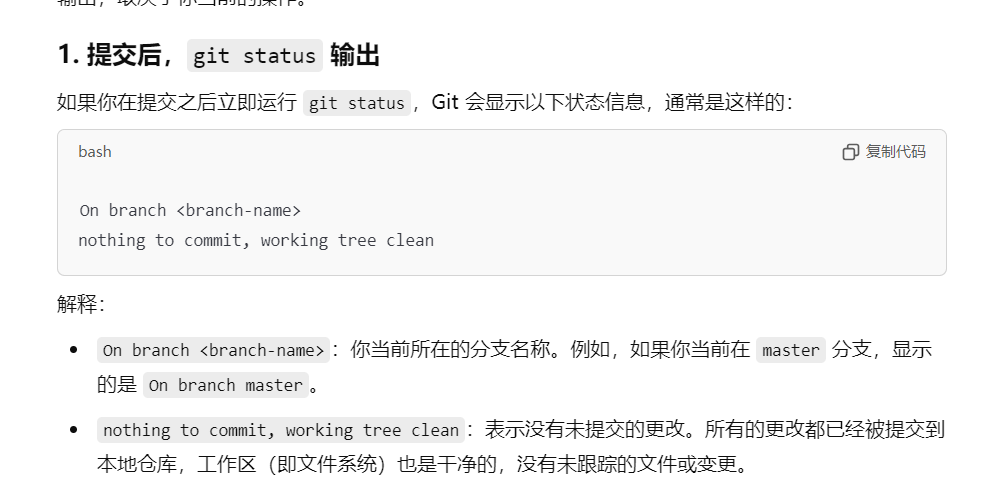
其实是虚拟表 一组查询语句 每次把视图当表使用时 都会执行一遍语句

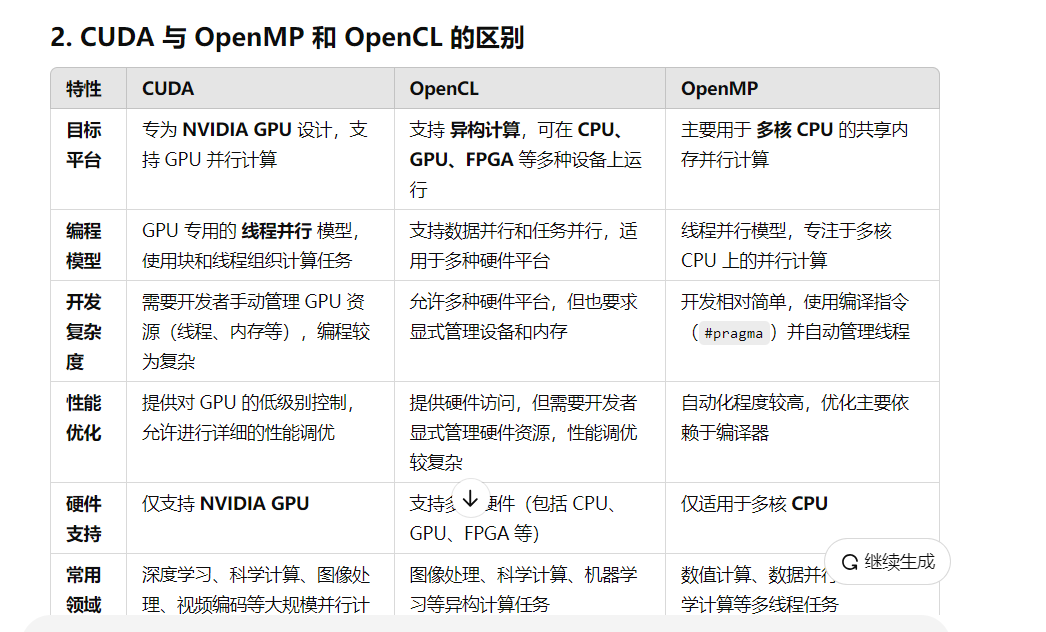


索引 单列索引 复合索引 主键索引 外键索引 唯一性索引、

利用查询 排序 但不利于 增删改 而且有空间消耗

请将select语句理解为展示





向量范数：对一个n维向量 xn ||x|| R^n->R 函数的结果是一个标量

范数函数的几个充分必要条件：

1 正定性 ||x||>=0 当x==0 时 ||x||=0

2 绝对齐次性 ||ax||=|a|||x||

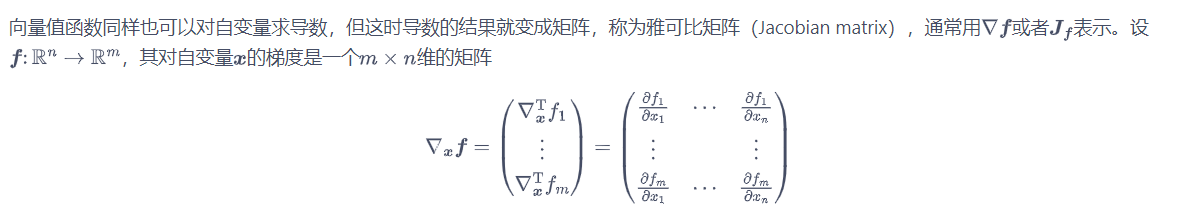
3 三角不等式 ||x+y||<=||x||+||y||

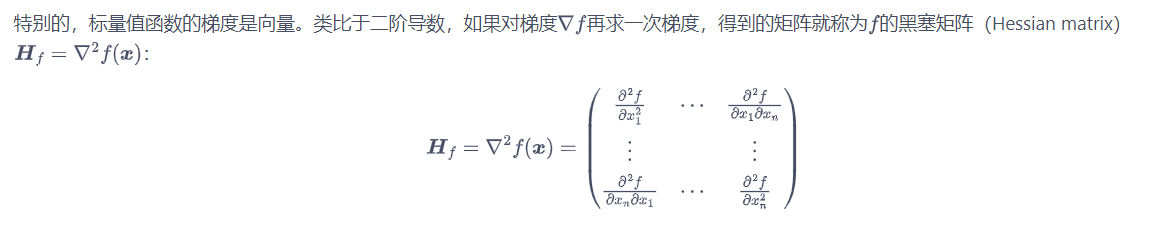
范数各式各样 常用的p-范数 对矩阵的范数 常见的为F范数

对于多元函数求梯度

多元泰勒函数 以可计算的多项式逼近无法计算的函数本身

向量值函数 R^n->R^m





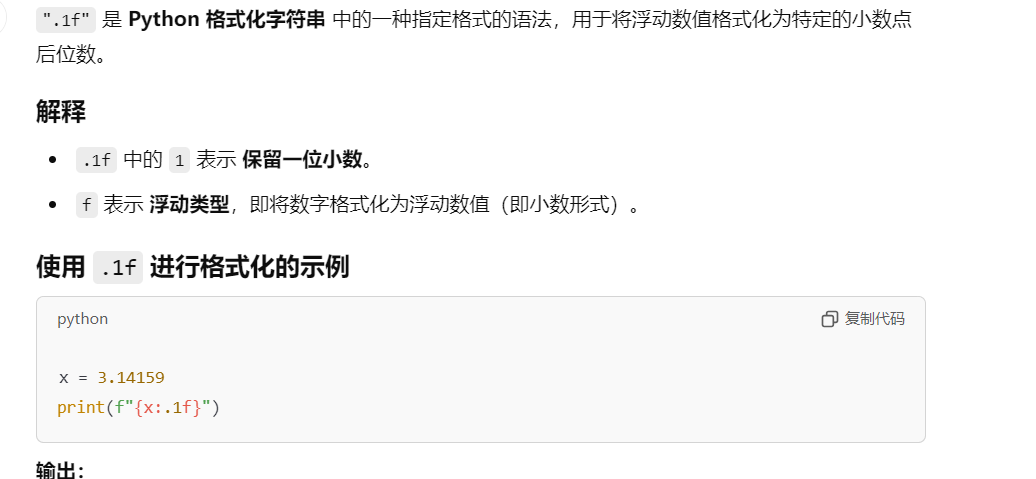
Numpy科学计算库

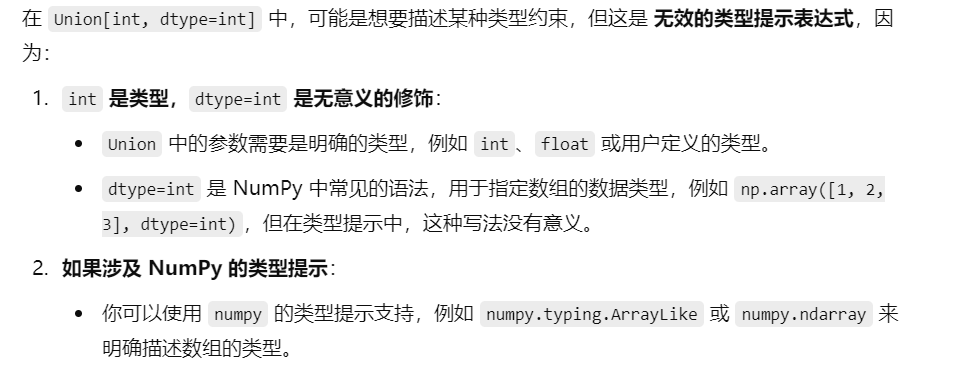
创建array对象

Array对象的几个属性 shape ndim

Array对象的几个操作 高级索引 高级切片

Array对象的几个计算









实验室有gpu的机器 nvida驱动（os层面上） cuda支持 你的pytorch代码

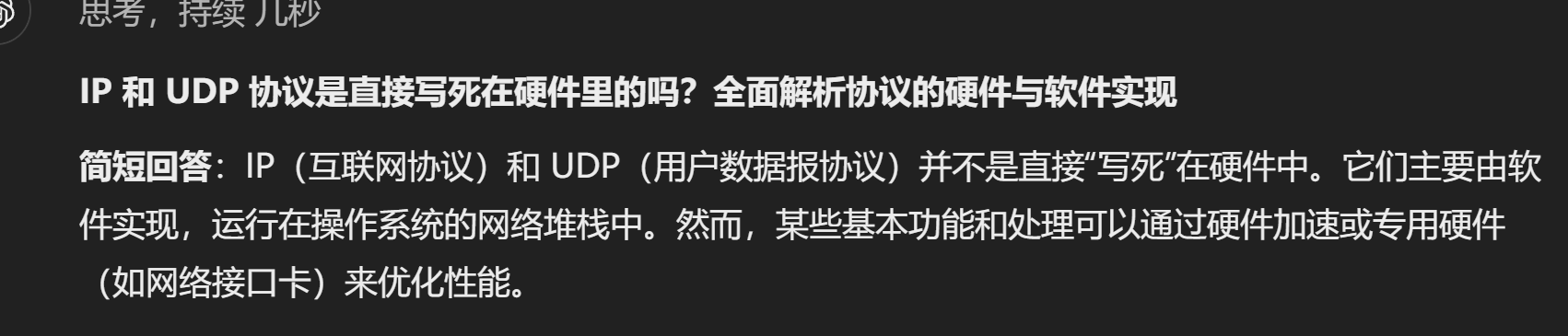
docker 就是一个软件 有引擎 有cli命令行

软件 协议 包

不进行网络通信的程序不占用网络端口

协议是规章制度 应用层协议通常由两端的软件实现 比如 ssh

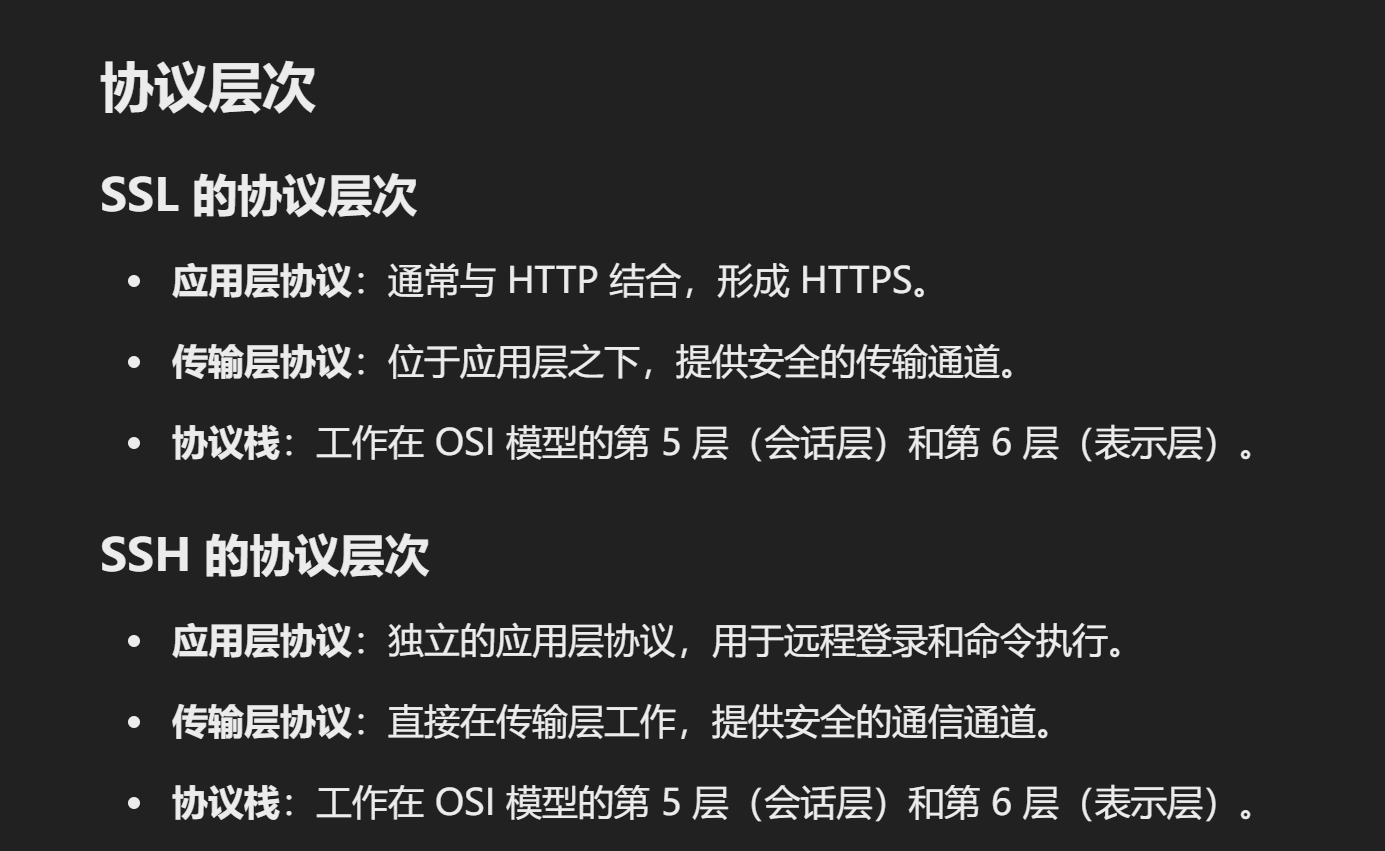




ssh

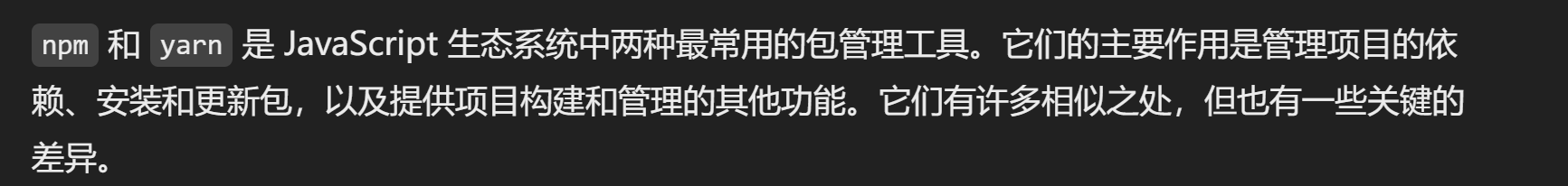


ssl/tls





关于ssl/tls的证书 公钥私钥以及 ssh的公钥和私钥









网络工程师

前后端开发工程师

算法工程师（各种研究领域）

数据分析工程师

软件工程师

测试工程师

硬件工程师

运维工程师 日志 监控

Ml

划分数据集 为 训练集和测试集 （可以一次划分也可以多次划分 称之为交叉验证，每次计算的loss进行累加取平均值 这样防止随即划分划分到有偏差的一组数据）

数据清洗打标签 最好能去除离群点也就是噪声

建立模型 损失函数 为了防止过拟合 可以加入一个正则化约束拉姆达 岭回归 lasso回归 分别采用L2 L1范数

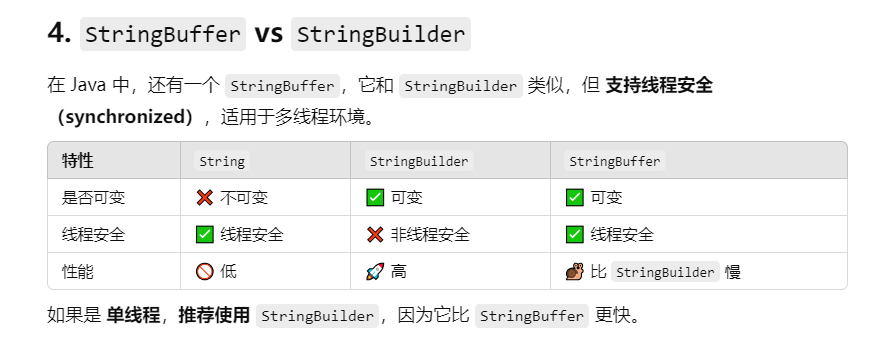
采用MBGD

对数据进行特征提取 相似度计算 特征归一化

对多个epoch的训练集数据和测试集数据进行作图分析 防止欠拟合和过拟合

训练的epoch过多 模型复杂度过高会导致过拟合 正则化约束可以有效改善

Logistic函数 和softmax函数 是将二分类和多分类问题转换为概率回归问题

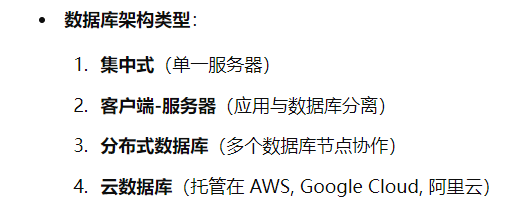






数据库管理系统

数据库管理系统体系结构





设计表开始 一对一 多对一 多对多 DDL

Crud DML DQL

权限控制 DCL

Sql数据类型 不同数据库有不同的实现

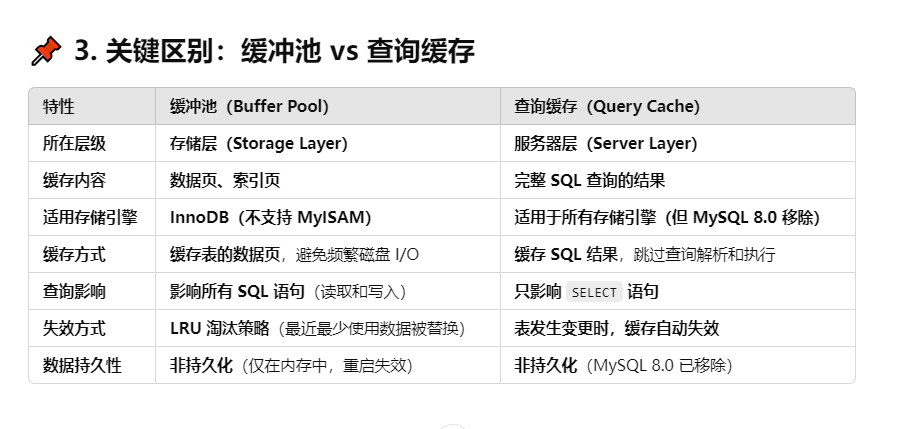
Mysql主要有数值类型 int bigint tinyint 字符类型char varchar text blob 日期类型 date time datetime json类型JSON set()多选类型 enum（）枚举类型

Sql语句分隔符

一般函数 触发器 存储过程会含有多个sql语句 因此需要先修改默认分隔符再改回去

表 视图 事务

六大约束



Mysql四层

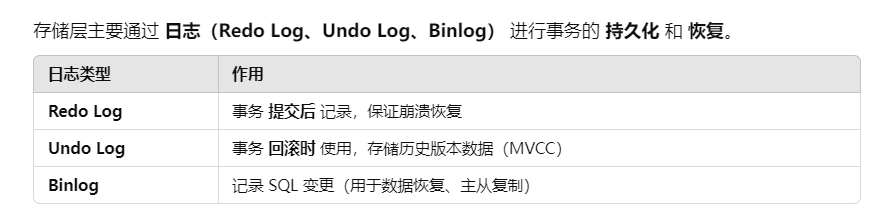
连接层 控制链接个数 核对密码

服务层 查询缓存(mysql8.0移除) sql解析器 查询优化 权限控制grant revokes 等等



可插拔存储引擎层 执行操作 执行事务 锁 多版本并发控制mvcc 事务隔离级别

存储层（数据文件 索引文件 日志文件 缓冲池）





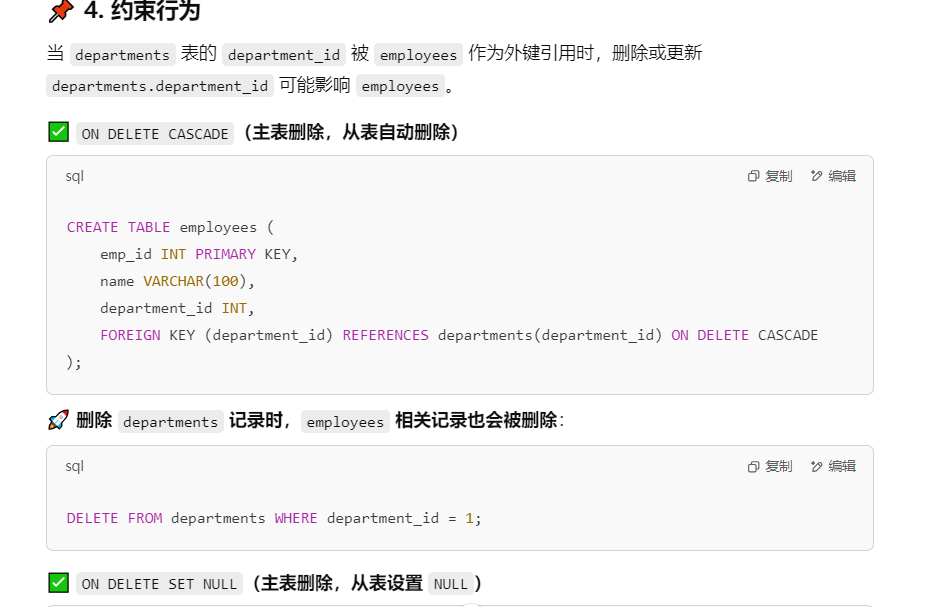
自动优化和手工优化 比如对一个join操作 查询优化器会进行优化 也可以手工优化,手工优化加索引 sql优化器识别索引 进行优化生成好的执行计划

日志，主从复制和分库分表和读写分离：

创建procedure function trigger都要用delimiter $$ 并且有begin end end末尾跟$$

创建transaction则 因为有start transaction 和 commit 或者rollback 所以不需要begin end 以及delimiter





并发事务问题



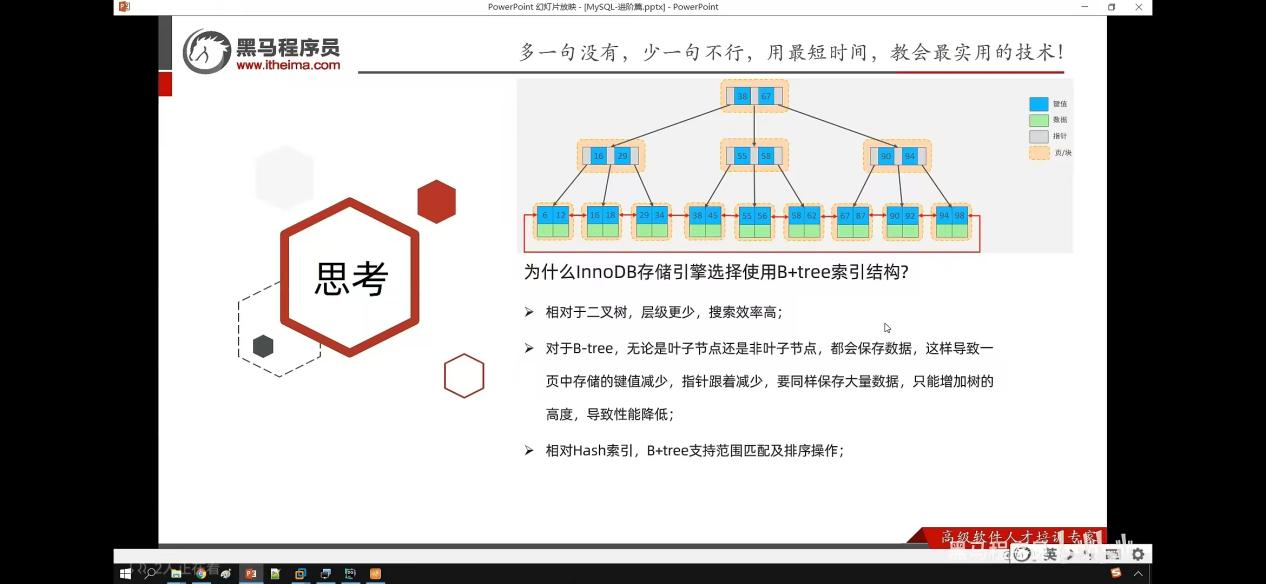
存储引擎是针对表的而不是针对数据库的 因此 不同表可以指定不同存储引擎

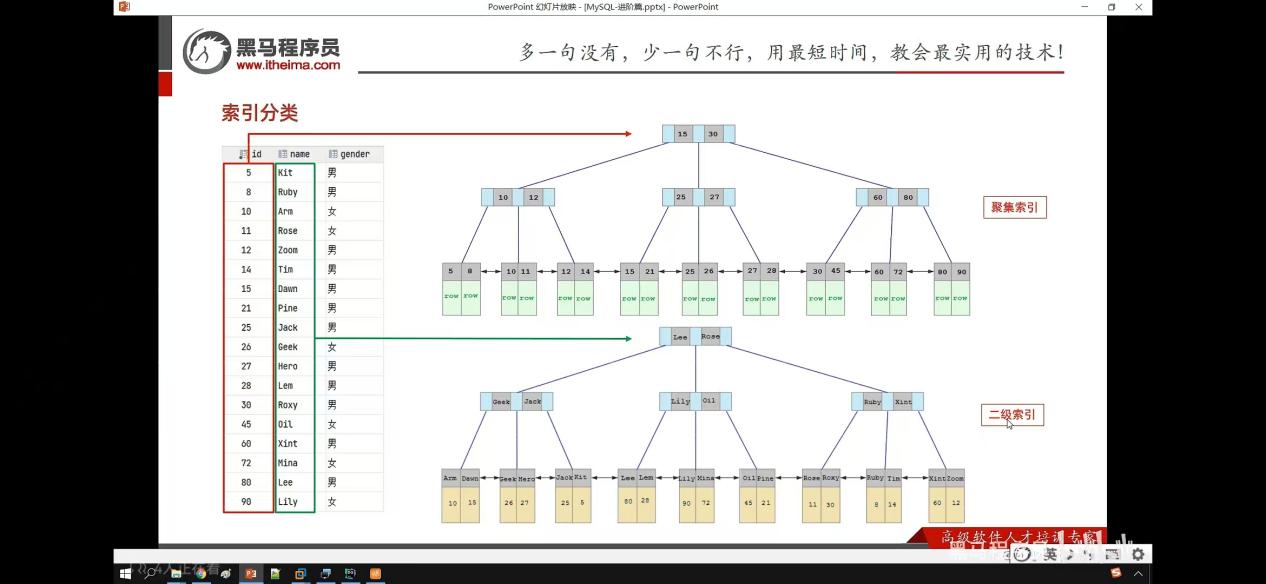
存储引擎又被称为表类型

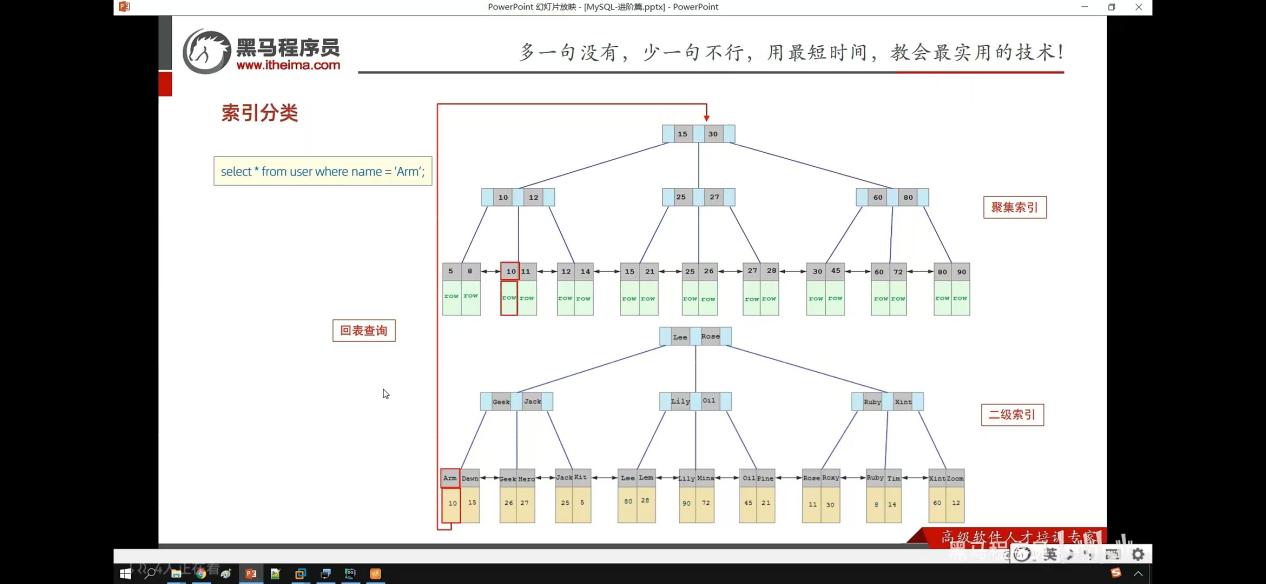
Innodb 支持 事务 外键 行级锁 b+索引

myisam 支持 表锁 访问快 b+索引

Memory 支持哈希索引 哈希索引不能范围查询和排序









对于特定的innodb 分为聚集索引和二级索引 聚集索引一般默认为主键索引或者第一个唯一索引

多用联合索引避免回表查询



视图 view on cascades check point(默认)（不检查本地的条件 但会检查级联的视图的条件）

On local check point 则会检查当前的以及用到的之前的

一个数据库 有系统变量（global 或者当前会话的sesssion） 用户变量 局部变量

Select @@系统变量名

set session/global 系统变量名 =变量值

Select @用户变量名 ：= 变量值

Set @用户变量名=变量值

Declare 变量名 变量类型（default）;

Set 变量名 =变量值；

Declare cursor

Declare handler

触发器目前是针对 update insert delete三种操作的三种触发器

Old new

目前只支持行级触发器 比如update语句影响了四行 那就触发四次 而不支持语句级，语句级是指触发一次

Begin

End;

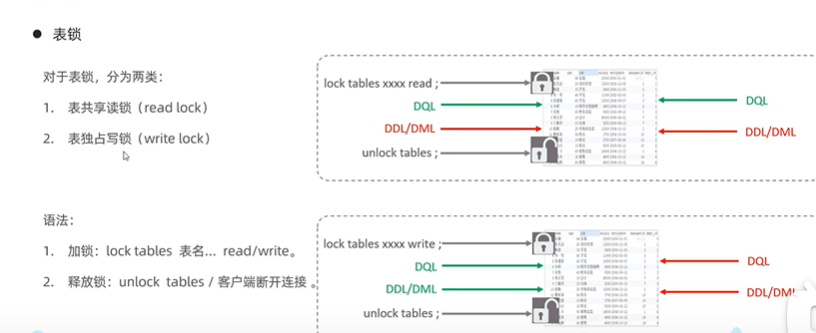
锁 分为 全局锁（锁整个数据库） 表级锁 行级锁

Lock tables people read

Unlock tables;

Lock tables people write

Unlock tables;

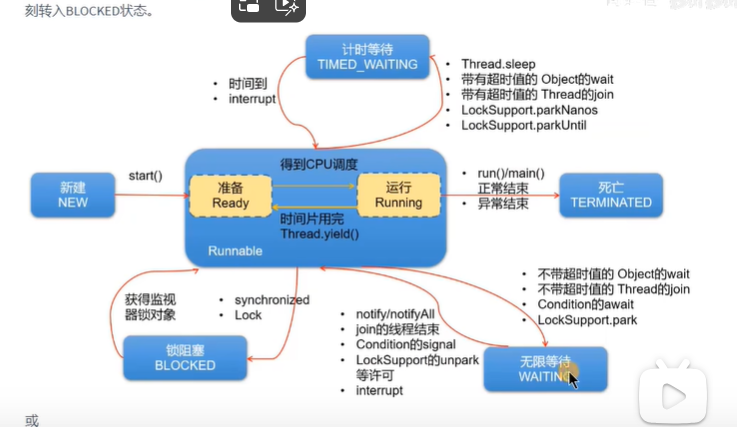


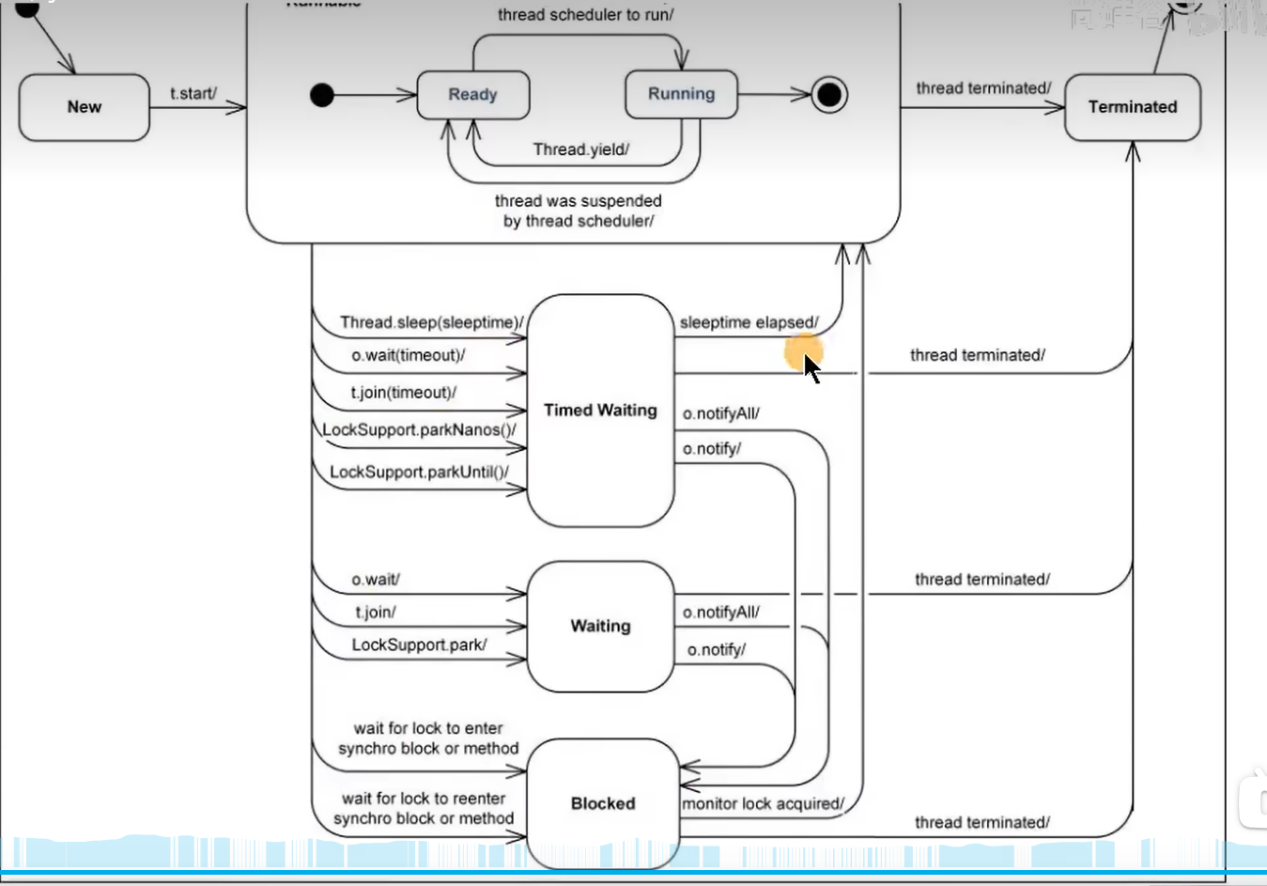
事务的提交伴随着锁的释放

锁的互斥与共享

Select语句会默认事务提交吗？

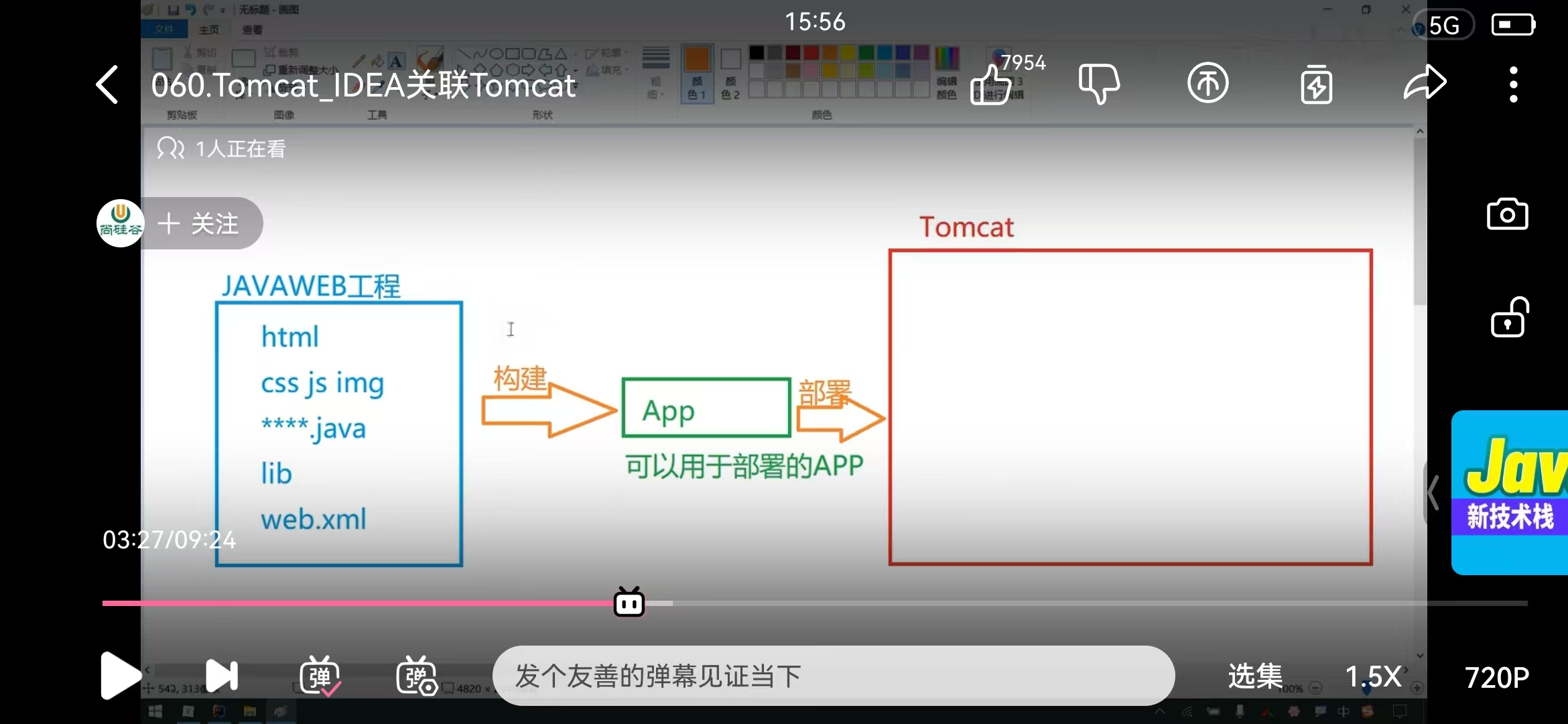


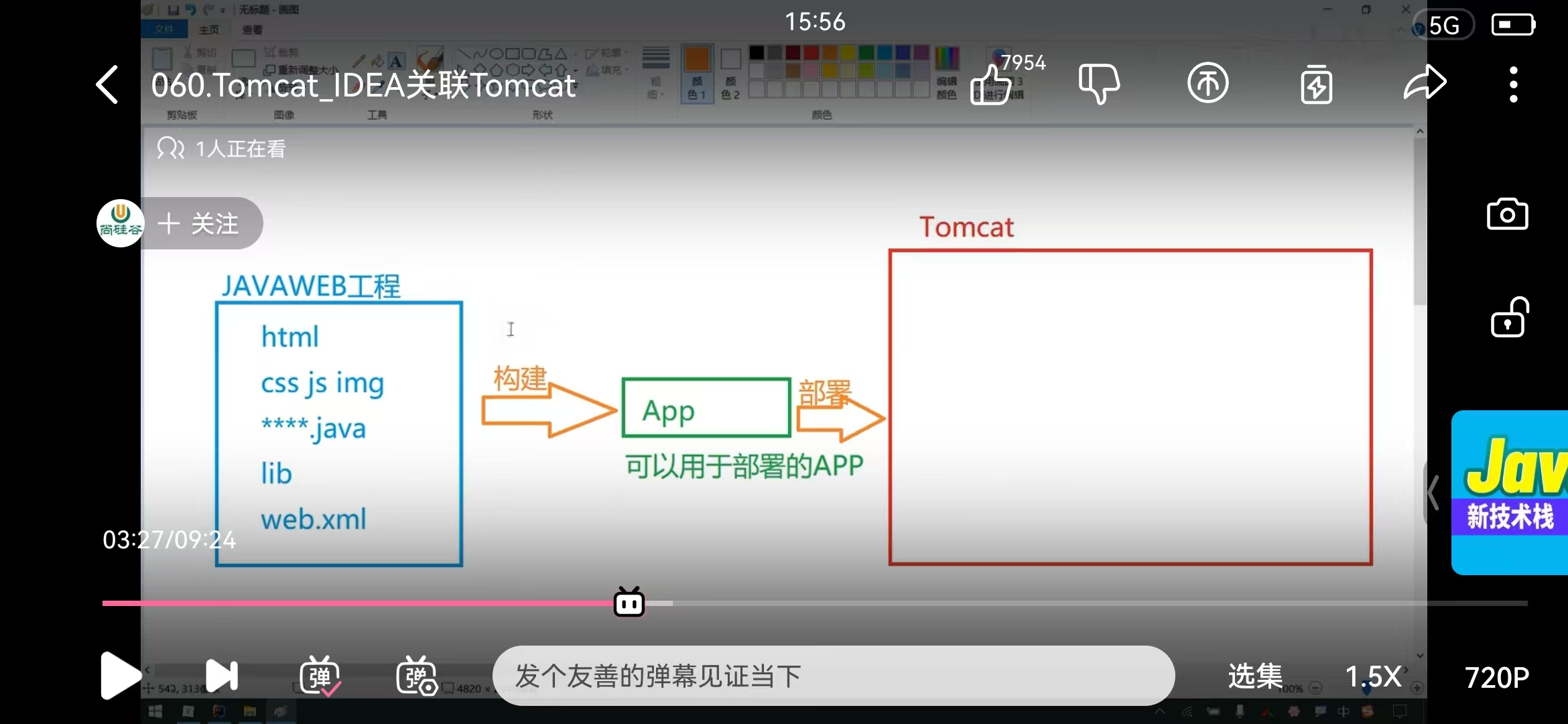




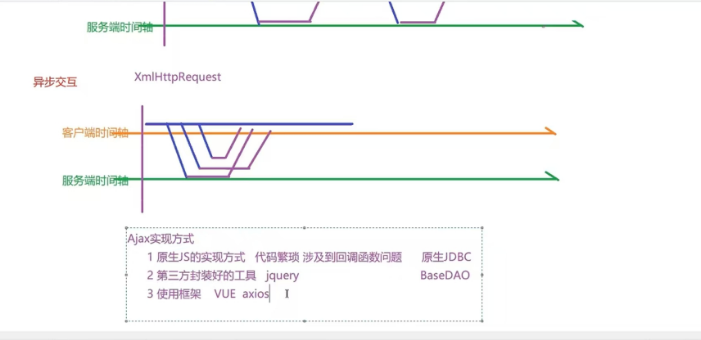














传统io(BIO) 新的io newio (nio)

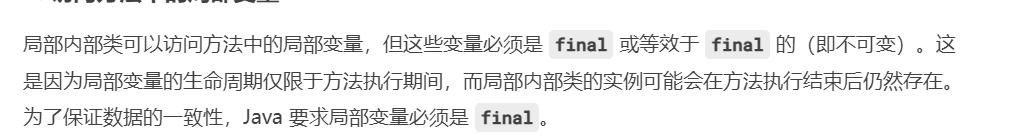


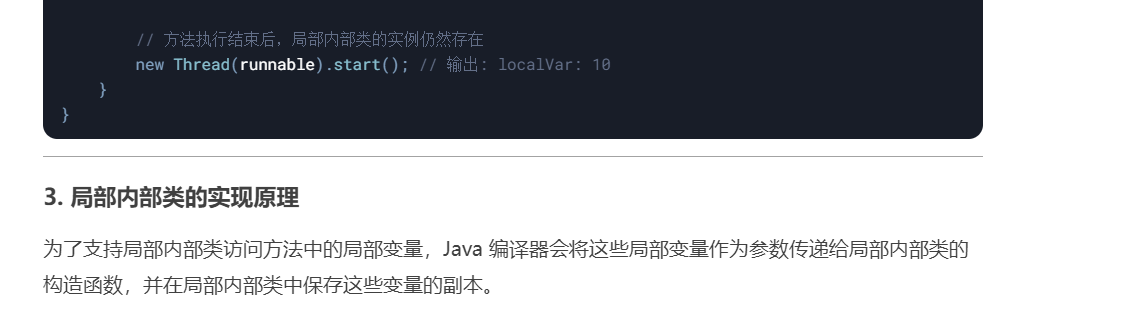




Jmm是比jvm更底层的内存划分







类的创建销毁不是在方法栈上 而是在堆上 由GC控制

所以在方法里声明的局部内部类的生命周期可能大于方法的生命周期 因此其访问的方法的局部变量会作为参数传入局部内部类的构造函数中 所以该参数必须是final类型的，局部内部类还可以访问外部类的属性包括静态的。











sql及索引优化 库表优化 系统配置优化 硬件优化

隐式的创建索引 一个字段如果是主键 外键约束 唯一性约束 自动创建索引

显示创建索引

索引的可见性 隐藏索引 一种软删除 避免直接删除索引造成问题

在join操作 索引嵌套循环 块嵌套循环（被废弃为hash join） 简单嵌套循环

索引下推