1. MySQL索引的作用，能不能每个字段都加索引？

答：索引就和目录的作用类似，在数据库表记录中，利用索引，可以快速过滤查找到数据记录。

可以每个字段都加索引，但是建议最好不要。首先创建索引和维护索引要耗费时间，这种时间会随着数据量的增加而增加。其次索引需要占物理空间，所需空间就会更大。最后这样会降低读写速度，而且为了保证它的正确性，就会浪费很多资源。

1. ==和===和equals的区别

答：在JavaScript中，严格相等运算符===比较类型的值，而相等运算符==只比较值是否相等。比较不同类型的数据时，相等运算符会先进行数据类型转换再进行严格相等运算符的比较。

在Java中，==是Java提供的等于比较运算符，用来比较两个变量指向的内存地址是否相同。而equals()是object提供的一个方法，默认实现就是返回两个对象==的比较结果。但equals()可以被重写。

1. 线程池的主要的参数？

答：

①corePoolSize(核心线程数量)。

向线程池提交一个任务，此时，若线程池已创建的线程数小于corePoolSize，即便此时存在空闲线程，也会通过创建一个新线程来执行该任务，直到已创建的线程数大于或等于 corePoolSize时，(除了利用提交新任务来创建和启动线程（按需构造），也可以通过 prestartCoreThread() 或 prestartAllCoreThreads() 方法来提前启动线程池中的基本线程。）

②maximumPoolSize(最大线程数量)

线程池所允许的最大同时可执行线程数量。当队列满了，且已创建的线程数小于maximumPoolSize，则线程池会创建新的线程来执行任务。另外，对于无界队列，可忽略该参数

③keepAliveTime(线程存活保持时间)

当线程池中线程数大于核心线程数时，线程的空闲时间如果超过线程存活时间，那么这个线程就会被销毁，直到线程池中的线程数小于等于核心线程数。

④workQueue(任务队列)

用于传输和保存等待执行任务的阻塞队列。

⑤threadFactory(线程工厂)

用于创建新线程。threadFactory创建的线程也是采用new Thread()方式，threadFactory创建的线程名都具有统一的风格：pool-m-thread-n（m为线程池的编号，n为线程池内的线程号）

⑥handler（线程饱和策略）

当线程池和队列都满了，再加入线程会执行此策略。

1. 抽象类中的方法能不能用final修饰

答：不可以，通过理解抽象类的作用我们就发现，抽象类必须要被继承，如果用final修饰抽象类，这个抽象类就无法被继承，自然就无法使用了。

1. Integer 和 int 的区别

答：①Integer是int的包装类，int则是java的一种基本数据类型

②Integer变量必须实例化后才能使用，而int变量不需要

③Integer实际是对象的引用，当new一个Integer时，实际上是生成一个指针指向此对象；而int则是直接存储数据值

④Integer的默认值是null，int的默认值是0

1. 线程的生命周期

当线程被创建并启动以后，它既不是一启动就进入了执行状态，也不是一直处于执行状态。在线程的生命周期中，它要经过新建（New）、就绪（Runnable）、运行（Running）、阻塞（Blocked）和死亡（Dead）五种状态。尤其是当线程启动以后，它不能一直“霸占”着CPU独自运行，所以CPU需要在多条线程之间切换，于是线程状态也会多次在运行、阻塞之间切换。

1. 什么是事务

事务就是用户定义的一系列数据库操作，这些操作可以视为一个完成的逻辑处理工作单元，要么全部执行，要么全部不执行，是不可分割的工作单元。

1. 对多态的理解

从定义上讲，[多态](https://so.csdn.net/so/search?q=%E5%A4%9A%E6%80%81&spm=1001.2101.3001.7020)是事物运行过程中存在的不同状态。多态存在需要满足三个前提：（1）要有继承关系（2）子类要[重写](https://so.csdn.net/so/search?q=%E9%87%8D%E5%86%99&spm=1001.2101.3001.7020)父类方法（3）父类数据类型的引用要指向子对象

1. 在vue中如何初始化echarts

第一步、安装

cnpm install echarts --save

第二步、引入

2.引入到项目中

2.1 全局引入 （在main.js中引入）

import echarts from 'echarts'

Vue.prototype.$echarts = echarts

直接绑定在vue实例上，在任何组件中，直接 this.$echarts 即可使用

2.2局部引入（在所需页面引入）

import echarts from 'echarts'

第三步、初始化echarts

3.1 你需要echarts的页面中得创建一个dom元素

<div id="myCharts" ref="myCharts"></div>

3.2 在mounted中初始化echarts （ 以下例子为折线图）

mounted(){

//使用init初始化

1. const myCharts = this.$echarts.init(this.$refs.myCharts);

2. let options = {

title: {

text: '过去某周气温变化', //图表顶部的标题

},

tooltip: { //鼠标悬浮框的提示文字

trigger: 'axis'

},

legend: {

data:['最高气温','最低气温']

},

xAxis : [{ //x轴坐标数据

type : 'category',

boundaryGap : false,

data : ['周一','周二','周三','周四','周五','周六','周日']

}],

yAxis : [{ //y轴坐标数据

type : 'value',

axisLabel : {

formatter: '{value} °C'

}

}],

series: [ //驱动图表生成的数据内容数组，几条折线，数组中就会有几个对应对象，来表示对应的折线

{

name:"最高气温",

type: "line", //pie->饼状图 line->折线图 bar->柱状图

data:[11, 11, 15, 13, 12, 13, 10],

},

{

name:"最低气温",

type: "line", //pie->饼状图 line->折线图 bar->柱状图

data:[1, -2, 2, 5, 3, 2, 0],

}

]}

3. myCharts.setOption(options);

}

ok ，这样就成功的实现了 echarts

1. char和varchar的区别

①CHAR是一种固定长度的类型，而VARCHAR则是一种可变长度的类型。

②在CHAR(M)类型的数据列内，每个值要占M个字节，若小于M，则MySQL会在它的右边用空格来补足，但在查询时，这些空格会被去掉。

③在VARCHAR(M)类型的数据列里，除了存放数据所占用字节，还会再占用一个字节来记录其内容长度（L+1个字节），多余空格将会在以后插入操作的过程中被去掉。

④采用CHAR会多占用空间，造成空间的浪费。（如果定义了CHAR(100)，最多可存放100个“a”或100个汉字，如实际存放一个“a”或一个汉字“我”，那么此列实际占用空间还是100，多出去的99个长补空格，浪费空间。如果采用VARCHAR，则节省空间。我们还用上边的例子，把CHAR(100)改为varCHAR(100)，情况完全不一样了，如实际存放一个“a”或一个汉字“我”，那么此列实际占用空间就是1个。）