# Java知识点 Part2

## MySQL引擎

常用的两个InnoDB和MyIASM引擎。

InnoDB支持事务，行级锁，外键约束。不支持全文检索。因为是行级锁，所以写操作效率高，不锁全表，高并发情况下效率高。不保留行数，select count(\*)的时候要扫全表。

MyIASM不支持事务行级锁外键约束。支持全文检索。因为不支持行级锁，写操作就锁全表。保留行数，select count(\*)不扫全表。读操作多的时候用这个比较好。

二者用的索引都是B+树，但MyIASM的树存的是地址，也就是索引与数据分开，这是非聚集索引。

InnoDB的B+树存的是实际值，聚集索引。

## SQL查询

有一张学生课程分数表tabA，id、学号、课程号、分数。求2门及以上课程分数<60的学号。

select 学号 from tabA where 分数<60 group by 学号 having count(sno) >=2;

## ArrayList删除元素

List有a,b,b,c,d,b等元素，删除b

Iterator it=list.iterator();

while(it.hasNext()){

String str=it.next();

if(“b”.equals(str)){

it.remove();

}

}

## Java的异常体系结构



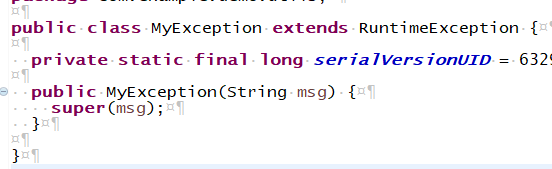
程序处理不了的错误，比如虚拟机报OutOfMemoryError、ThreadDeath这些，一旦发生，JVM直接就终止线程了。

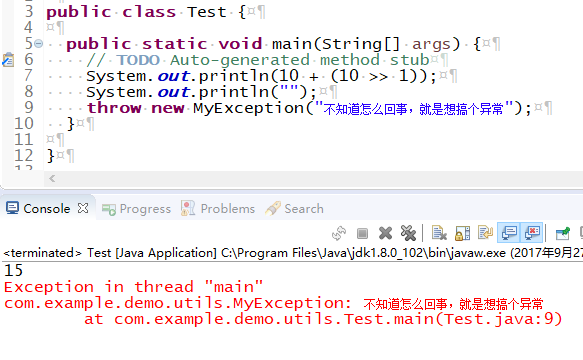
程序能处理的都叫异常。

运行异常，子类有比如空指针异常、越界异常。

非运行异常，比如IO异常，SQL异常，自定义的异常。

## 自定义一个异常





## 如何防止SQL注入

1. 预编译。MyBatis里面就用#，别用$。
2. 过滤。可以通过正则表达式或者关键字，对输入的内容进行校验。可以放在Java程序里，也可以通过JS放在页面中。

## 事务

事务四大特性ACID：

**原子性**，N个步骤，要么全成功，要么全失败。

**一致性**，事务执行前后必须保持一致性。比如我有1000，你有1000，不管咱俩怎么转账，最终合计都必须是2000。

**隔离性**，事务独立，不干扰。A事务新增1千条记录，不提交，别的事务都查不到这些数据。

**持久性**，提交之后结果持久保存。

**脏读：**A事务读了未提交的B事务的数据。B事务插入100条数据，但是只插入了50条，A事务就能读到这50条。

**不可重复读：**A事务中查询两次数据，数据不同。A事务查询某值为1，B事务更新某值为2，A事务继续查询某值为2。

**幻读：**A事务将表中所有记录的X字段从1改为2，这时候B事务插入一条数据，X字段还是1。A事务再查一下，字段X本应该全部都是2，现在却多了一个为1的。

隔离级别，从高到低：

**串行化：**隔离100%，脏读、不可重复读、幻读都能避免。

**可重复读（MySQL默认）：**新增数据，对其他事务可见，修改不可见。避免不了幻读。

**读取已提交：**A事务读取数据X，B事务对X做了修改或删除，并提交事务，A再读时是新数据。避免了脏读。

**读取未提交：**同上，B事务修改之后没提交，A事务也能读取新的数据。

Spring的隔离同上面4个，但是多了一个：使用数据库默认的隔离级别。

Spring的传播行为：

propagation\_requird：支持当前事务，如果没有就新建一个。

propagation\_supports：支持当前事务，如果没有就按照非事务运行。

propagation\_mandatory：必须有个事务，否则报异常。

propagation\_requires\_new：必须在自己事务中运行，如果有一个事务运行的话，挂起。

propagation\_not\_supported：不在事务中运行，如果有一个事务运行，挂起。

propagation\_never：不在事务中运行，如果有事务运行，抛异常。

propagation\_nested：嵌套事务，父事务提交它提交，父事务回滚它回滚。