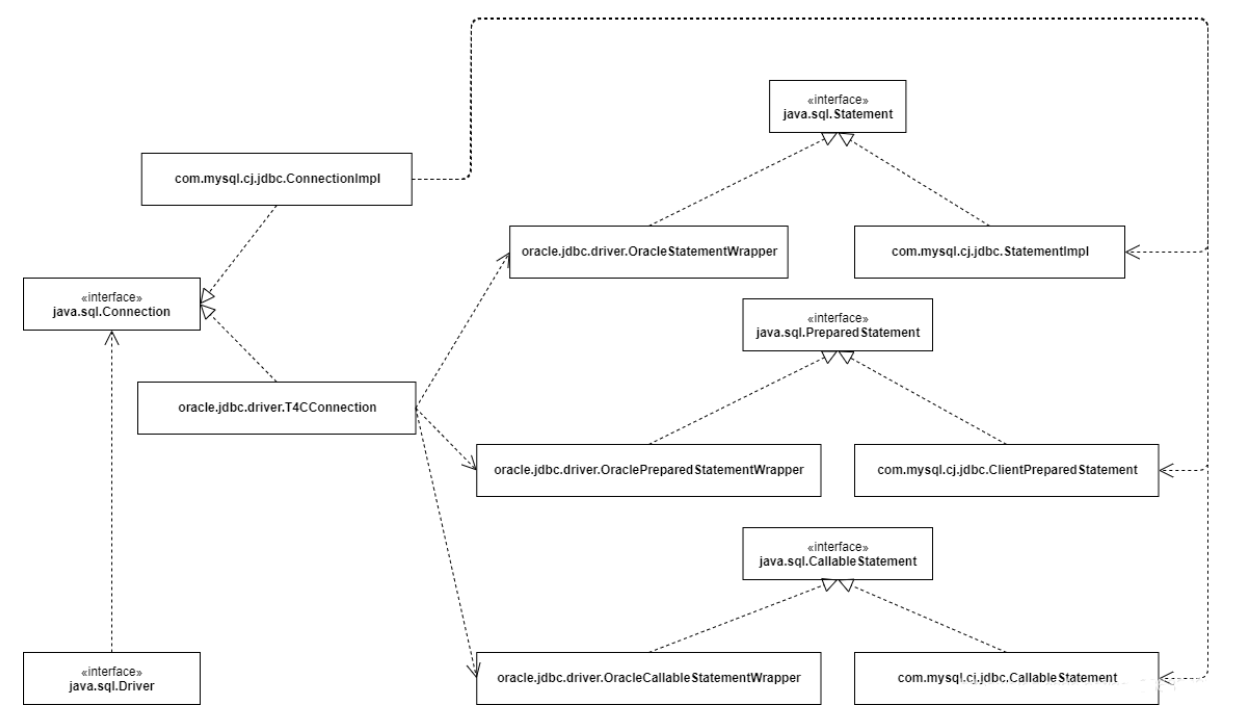
抽象工厂模式  
  
抽象工厂模式（Abstract Factory Pattern）是围绕一个超级工厂创建其他工厂。该超级工厂又称为其他工厂的工厂。  
  
这种类型的设计模式属于创建型模式，它提供了一种创建对象的最佳方式。  
  
在抽象工厂模式中，接口是负责创建一个相关对象的工厂，不需要显式指定它们的类。每个生成的工厂都能按照工厂模式提供对象。  
  
  
介绍  
  
意图：提供一个创建一系列相关或相互依赖对象的接口，而无需指定它们具体的类。  
主要解决：主要解决接口选择的问题。  
  
何时使用：系统的产品有多于一个的产品族，而系统只消费其中某一族的产品。  
  
如何解决：在一个产品族里面，定义多个产品。  
  
关键代码：在一个工厂里聚合多个同类产品。  
  
应用实例：  
 1、JDk中的 java.sql.Connection  
 2、UIManager (swing外观)  
  
优点：当一个产品族中的多个对象被设计成一起工作时，它能保证客户端始终只使用同一个产品族中的对象。  
  
缺点：产品族扩展非常困难，要增加一个系列的某一产品，既要在抽象的 Creator 里加代码，又要在具体的里面加代码。  
  
使用场景：  
 1、QQ 换皮肤，一整套一起换。  
 2、生成不同操作系统的程序。  
  
注意事项：产品族难扩展，产品等级易扩展。  
  
  
JDk中的 java.sql.Connection:  
 jdk中连接数据库的代码是典型的抽象工厂模式，每一种数据库只需提供一个统一的接口：  
 Driver-> Connection 是一个经典的抽象工厂，而 Statement、PreparedStatement、CallableStatement 是 Connection 这个抽象工厂中提供的三个抽象产品  
数据库Driver 起到 Client 的作用，我们只需要把 Driver 注册进 DriverManager，就可以生成需要的 Connection。每次操作数据库只需要使用 Java 提供的这套接口就可以，不需要考虑使用的是什么 SQL 数据库  
这些抽象工厂与抽象产品均由对应的数据库驱动实现，下面以 MySQL 与 Oracle 的驱动进行举例。



前提: SPI

DriverManager.getConnection() ->　时根据类加载时机．　执行

DriverManager中的 static块. 加载mysql驱动 . 即 mysql中的Driver类，它实现了 Driver接口.

再getConnection()得到 mysql中的 ConnectionImpl类的对象.