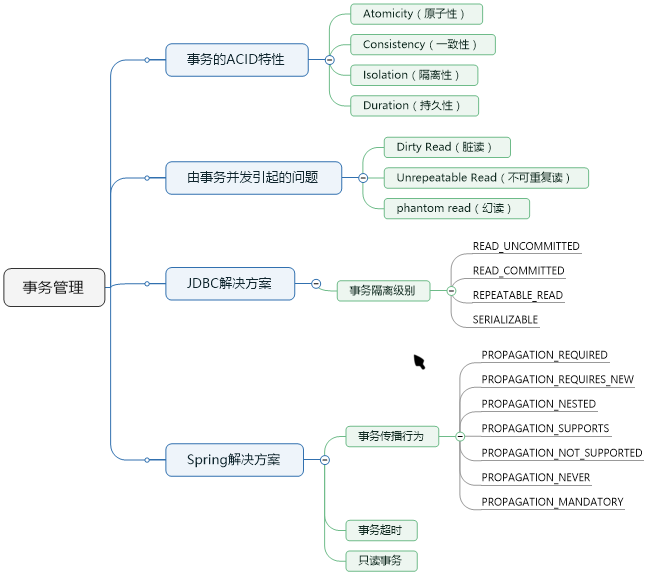
1. **超键、候选键、主键、外键**
   1. 超键：在关系中能唯一标示元组的属性集称为关系模式的超键。一个属性或者多个属性组合在一起都可以作为超键，包括候选键和主键
   2. 候选键：是最小超键，即没有冗杂原色的超键
   3. 主键：数据库中对存储数据对象予以唯一和完整标示的数据列或属性的组合。一个数据列只能有一个主键，且主键不能为空
   4. 外键：在一个表中存在的另一个表的主键，就是这个表的外键
2. **什么是事务？**
   1. 就是被绑在一起作为一个逻辑工作单元的sql语句分组，如果任何一个语句操作失败，则整个操作都会失败，，然后就会回滚到操作前的状态或者是之前的节点，要将有组语句作为事务考虑。就需要通过ACID测试，即原子性，一致性，隔离性，和持久性
   2. 原子性是基础，隔离性是手段，持久性是目的，一致性才是老大
   3. 隔离性的问题
      1. Dirty read脏读
         1. 事务A读取了事务B未提交的数据，并在这个基础上又做了其他操作
      2. Unrepeatable read 不可重复读
         1. 事务A读取了事务B已提交的数据
      3. Phantom read 幻读
         1. 事务A读取了事务B已提交的新增数据
   4. Transaction的隔离级别(Transaction Isolation Level)级别从小到大：

dirty read unrepeatable read phantom read

* + 1. READ\_UNCOMMITTED 允许 允许 允许
    2. READ\_COMMITTED 禁止 允许 允许
    3. REFEATABLE\_READ 禁止 禁止 允许
    4. SERILIZABLE 禁止 禁止 禁止
  1. Spring对JDBC进行了补充和扩展，提供了一个重要功能那就是事务传播行为（Transaction Propagetion Behavior）,共有7种,Spring解决的是方法之间的事务传播，从方法A到方法B，需要面对B，看A有事务吗？
     1. PROPAGATION\_REQUIRED（Spring默认传播行为）
        1. A没有就新建一个事务，如果有就加入当前事务
     2. PROPAGATION\_REQUIRES\_NEW
        1. A没有就新建一个事务，有就将当前事务挂起，意思创建了一个新的事务，和原来的事务没有关系了
     3. PROPAGATION\_NESTED
        1. A没有就新建一个事务，有就在当前事务中嵌套其他事务，所嵌套的子事务与主事务之间是有关联的（当前事务提交或回滚，子事务也会提交或回滚）
     4. PROTAGATION\_SUPPORTS
        1. A没有就以非事务方式执行，有就使用当前事务
     5. PROTAGATION\_NOT\_SUPPORTED
        1. A没有就以非事务方式执行，有就将当前事务挂起
     6. PROTAGATION\_NEVER
        1. 如果没有就当非事务执行，有就抛异常
     7. PROGATATION\_MANDATORY
        1. 如果没有就抛异常，有就使用当前事务
  2. Spring还提供了一些附加的事务功能
     1. 事务超时：(Transaction Timeout)：为了解决事务时间太长，消耗太多资源，给事务设值最大时长，如果超出就回滚事务
     2. 只读事务(Readonly Transaction)：为了忽略那些不需要事务的方法，读读取数据
     3. 

1. **什么是锁？**
   1. 在所有的DBMS中，锁是实现事务的关键，锁可以保证事务的完整性和并发性。与现实生活中锁一样，可以使某些数据的拥有者，在某段时间内不能使用某些数据或者数据结构，锁是分级别的
2. **数据库的四个特性和含义**
   1. 原子性Atomicity：整个事务中的所有操作，要么全部成功，要么全部失败。事务在执行过程中发生错误，会被回滚（Rollback）到事务开始之前的状态或者某个节点，就像这个事务从来没有执行过一样。
   2. 一致性Consistency：在事务开始之前和事务结束之后，数据库的完整性约束没有被破坏
   3. 隔离性Isolation：如果有两个事务，运行在相同的时间内，执行相同的功能，事物的隔离性将确保每一事务在系统中认为只有该事务在使用系统。这种属性称之为串行化，为了防止事务操作间的混淆，必须串行化或序列化请求，使得在同一时间仅有一个请求用于同一数据
   4. 持久性Durability：在事务完成以后，该事务对数据库的所作的更改便持久化的保存在数据库中，并不会被回滚
3. **什么是视图**
   1. 视图是一种虚拟的表，具有和物理表相同的功能。可以对视图进行CRUD操作，视同通常是有一个表或者多个表的行或列的子集。对视图的修改不影响基本表。它使得我们获得数据更容易，相比多表查询
      1. 不希望访问者获取整个表的信息，只暴露部分字段给访问者，所以就建立了一个虚表，就是视图。
      2. 查询的数据来源于不同的表，而查询者希望以统一的方式查询，这样也可以建立一个视图。直接从视图中获取数据
      3. 注意：这个视图是在数据库中创建的不是代码创建的
4. **触发器的作用？**

触发器是一种特殊的存储过程，主要通过事件触发而被执行的，可以强化约束，来维护数据的完整性和一致性，可以跟踪数据库的操作从而不允许未经许可的更新或者变化。可以联级运算。如：某表上的触发器包含对另一个表的数据操作，而该操作又会导致该表触发器被触发

1. 维护数据库的完整性和一致性优先使用约束，其次使用触发器，最后万不得已使用自定义的业务逻辑
2. **索引**
   1. 数据库索引是数据管理系统中一个排序的数据结构，以协助快速查询。更新数据库表中数据，数据库的索引通常使用B树或者是B+树
   2. 在数据库之外，数据库系统还维护着满足特定查找算法的数据结构，这些数据结构以某种方式引用（指向）数据，这样就可以在数据结构上实现高级查找算法。这种数据结构就是索引
   3. 为表设值索引一是增加了数据库的存储空间，二是在插入或者修改数据的时候要花费较多的时间（因为索引在此时也会发生变化）
   4. 优点：创建索引达到提高系统的性能
      1. 通过创建唯一性索引，可以保证数据库表中每一行数据的唯一性
      2. 加快数据的检索速度，（创建索引的最主要原因）
      3. 加速表与表之间的连接（实现数据的参考完整性方面特别有意义
      4. 在使用分组和排序子句进行数据检索时，也可以减少查询中分组和排序的时间
      5. 可以在查询过程中，使用优化隐藏器，提高系统的性能
   5. 缺点：
      1. 创建索引和维护索引耗费时间，时间和数据量成正比
      2. 索引需要占用物理空间，建立聚簇索引空间会更大
      3. 当表中的数据发生改变，索引也要动态维护，这就降低了数据的维护速度，索引是建立在列上面的。考虑该建在哪些列上面：
         1. 在经常需要搜索的列，加快搜索速度
         2. 在作为主键的列上，强制该列的唯一性和组织表中数据的排列结构
         3. 在经常使用在连接的列上，这些列主要是一些外键，加快连接速度
         4. 在经常需要根据范围进行搜素的列上创建索引，因为索引已经排序，其指定的范围是连续的
         5. 在经常需要排序的列上，因为索引已经排序可以利用索引的排序加快排序查询时间
         6. 在经常使用在where子句中的列上面创建索引，加快条件的判断速度
      4. 某些列不要创建索引：
         1. 取值少，如性别
         2. 很少用到的列
         3. 某些大数据，text image bit数据等
         4. 当修改性大于检索性
3. **Drop delete truncate区别**
   1. Drop直接删除表
   2. Truncate删除表中数据，再插入时自增长id又从1开始
   3. Delete删除表中数据，可以加where子句
   4. Delete语句每次删除一行，同时将此次操作作为事务记录日志保存，以便回滚
   5. Truncate table则一次性的从表中删除所有数据，不能恢复，也不会激活与表之间的删除触发器，执行速度快，对于有foreign key约束的表不能使用truncate
   6. 表和索引占空间，当表被truncate后，这个表和索引的所占空间恢复到初始大小，而delete不会减少空间大小，drop则会把所有空间释放
   7. Drop>truncate>delete truncate只能对表，delete可以是table也可以是view，truncate和delete只删除数据，drop删除整个表数据和结构
   8. Truncate和不带where的delete只删除数据。Drop则删除表的结构被依赖的约束(constrain)，触发器trigger索引index，依赖于该表的存储过程/函数将被保留，但其状态会变为invalid
   9. Delete为DML(data maintain Language)，这个操作会放到rollback segment，事务提交后才会生效，如果有相应的trigger执行的时候将被触发
   10. Truncate drop 是DDL(data define language)，操作立即生效，不能回滚
4. **数据库的三大范式**
   1. 第一范式1NF，在任何一个数据库中，1FN是对关系模式数据库的基本要求，不满足1FN的数据库就不是关系数据库
      1. 1NF数据库的每一列都是不可分割的基本数据项，同一列中不能有多个值 1NF中表的每一行只包含一个实例的信息
      2. 简而言之就是1NF要求数据库表中的每一列(每个字段)必须是不可分割的最小单元
   2. 第二范式2NF，简而言之就是要求表中的所有列，到必须依赖于主键，而不能有任何一列与主键没有关系
   3. 第三范式3NF前期要满足第二范式2NF，3NF要求表中的每一列只与主键直接相关而不是间接相关，表中的每一列只能依赖于主键