1. 高效TOPN（自定义分区、分组、排序）
2. 共同好友案例
3. 本地提交，集群运行
   1. conf.set("mapreduce.job.jar", "D:\\javaProject\\mr\_demo\\target\\mr\_demo-1.0.jar");
   2. conf.set("fs.defaultFS","hdfs://mini1:9000");//设置文件系统
   3. conf.set("mapreduce.app-submission.cross-platform", "true");//设置跨平台提交
   4. conf.set("mapreduce.framework.name", "yarn");//设置进行计算的框架是yarn
   5. conf.set("yarn.resourcemanager.hostname", "mini1");//设置resourcemanager
   6. System.setProperty("HADOOP\_USER\_NAME","root"); //解决权限问题
4. Mr的join
   1. Reduce端的join：处理的思想就是在map端将来源不同或者是不同用处的数据打上不同的标记，以便在reduce端能够识别并进行关联关系。适合全是大表的场景，优点：解决的业务范围比较广。缺点：从map端到reduce端的数据量很大，有些时候无效数据比较多，增加了传输时间
   2. Map端join:处理的思想就是将小表进行分布式缓存，然后在map端取出数据进行关联查询。使用场景：大表和小表同时存在（至少需要一张小表）优点：从缓存中取数据，在map端关联，减少了map端到reduce的数据传输。缺点：只是用有小表的需求
5. Job提交流程
6. 