**ONEDATA平台数据存储与优化**

1. **节约计算资源的使用**

为了节约计算资源的使用，现做出如下更改：

所有层job的计算资源Core缩小至10，Memory缩小至5G。数据量较大的JOB特殊对待，在WorkFlow调度时单独修改其使用资源（修改executor的同时，不要忘记修改max executor）

即:

默认情况：

spark.num.executors=5

spark.executor.cores=2

spark.executor.memory=5g

spark.driver.memory=5g

spark.max.executors=5

特殊情况：

spark.num.executors=10

spark.executor.cores=2

spark.executor.memory=10g

spark.driver.memory=5g

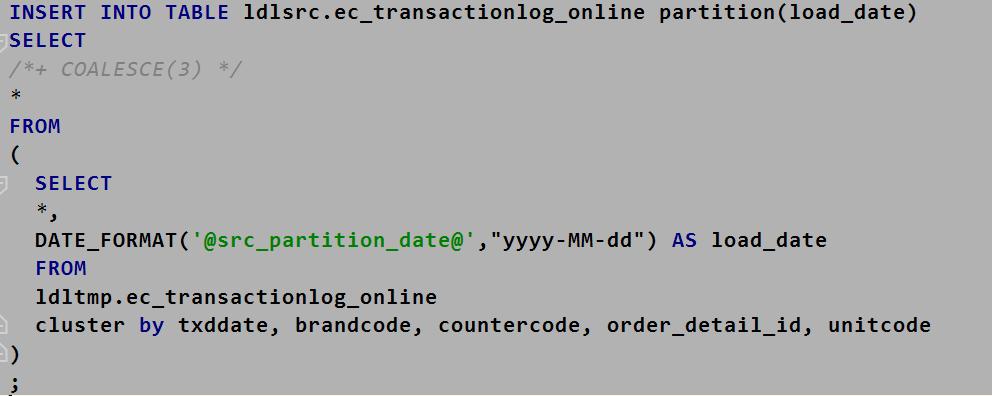
spark.max.executors=10

1. **保证源头数据的分布均匀**

ONEDATA的数据在进入了DWS层后才会进行复杂的业务逻辑计算，为了保证在计算的时候数据是均匀分布的就必须在计算前保证从底层表取出的数据是均匀分布的。在控制了底层数据均匀分布的情况下，在当前层计算时，才能保证不会因为数据分布的不均匀导致计算时间延长、数据倾斜 等问题

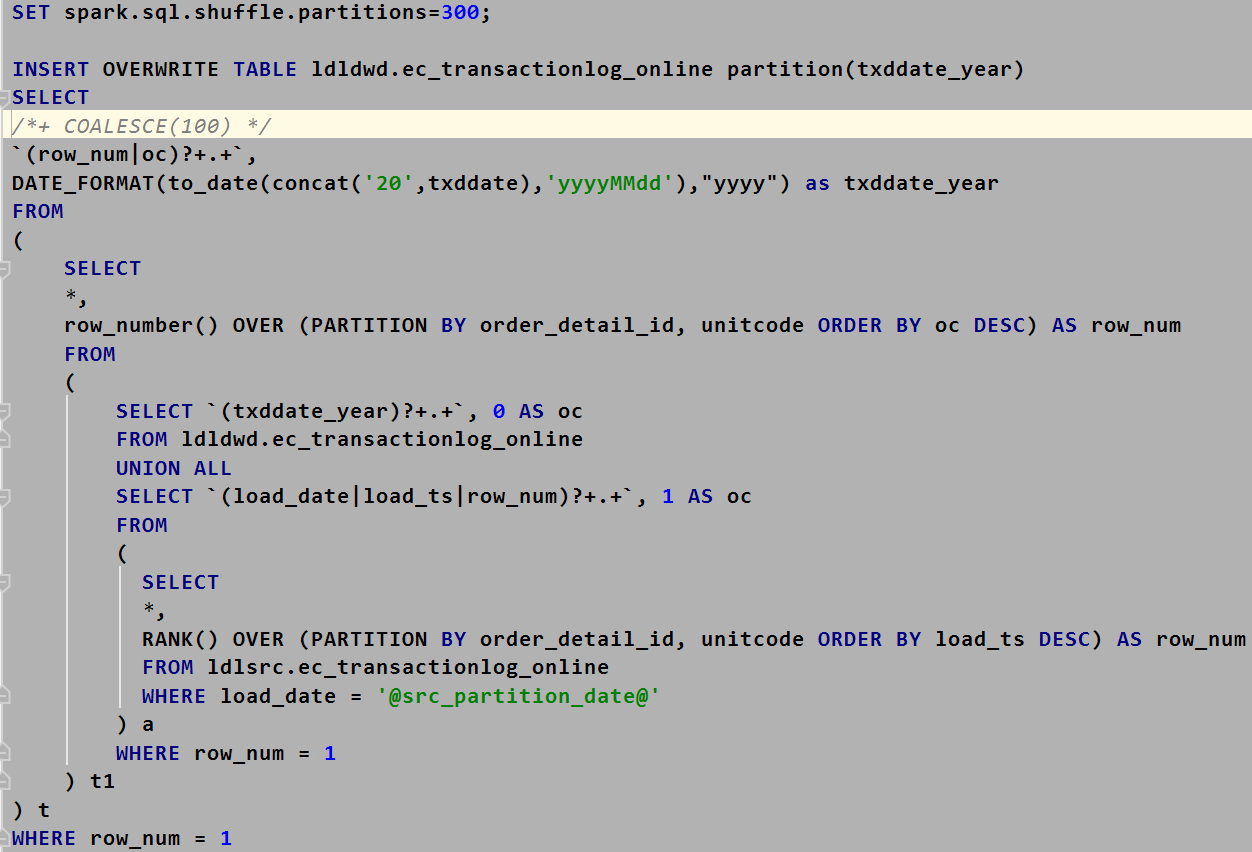
TMP -> SRC 的逻辑必须严格控制如下：

通过CLUSTER BY 将数据排列重新混洗，再通过COALESCE（3）告诉SPARK，存盘前先收缩分区至3个



1. **DWD、DWS、ADS的处理逻辑**

在确保了底层表数据的均匀分布后，当前表只需要添加COALESCE，将存盘时的分区数设置为你需要的大小，避免小文件问题。当然不要忘记了设置当前任务的并发度，并发度的设置可以参考实际分区数，最好是实际分区数的倍数



1. **不同分区的数据分片控制**

当某些分区的数据量偏小，可能低于最大分区数据量的5—10倍 或者更小的时候。可以针对不同分区，分别设置对应的数据分片数量 --- 利用Hive的多表写入

例如：在2014年的时候公司订单量非常小，在数据量偏小的情况下，设置为10个分片

但是2018年开始公司的订单量出现爆炸式增长，随着每年的增长，逐年调增分片数

