数据同步的题目,怎么去重,怎么更新数据

在Kettle里数据同步时,有个去除重复数据的组件可以去重

Kettle 如何做增量同步, 海量数据用增量更新很慢吧,怎么不在etl中直接实现

方式有挺多,比如两个表输入—插入更新,或者是建时间戳中间表.其实原理都是取源表增量数据更新到目标表,ETL工具只是处理跨库用的,只要是工具就会有出错的可能,至少我们项目组是能用存储过程的尽量用存储过程做数据同步.

有一张表A,表结构有100个字段.表B,表结构有50个字段,表结构不同的话怎么去同步？

字段映射

什么是 映射 ？mapping=映射.

在ETL过程中源表和 目标表的对应关系就是一种映射关系 .

在逻辑模型中表和表之间的关系 也叫做 映射 .

数仓架构是怎么样的,层次怎么划分

我们项目是传统数仓,主要分为三层,贴源层、数仓层、集市层

分别的工作内容：

ODS层（贴源层/可操作数据层）主要是做数据的抽取、转换和同步,通过调度工具（CTM）ETL按照每天/每月,定期从源系统中将数据抽取到ODS层；(数据的收集汇总)

DW层（数据仓库层）主要是从ODS层同步数据到数据仓库,根据业务需求对事实表和维度表进行建模（维度建模, 星型模型和雪花模型）；

DM层（数据集市层）主要是为了报表提供数据支撑,会从DW层同步数据到DM层,同时对各项指标提供数据或者通过函数计算得到报表需要的数据指标. (数据的分析和报表数据展示)

三层的解释：

 ODS（贴源层）主要是数据的收集汇总,通过ETL工具来实现,数据抽取转换和同步.作用：ODS层是数据仓库准备区,为DW层提供基础原始数据,可减少对业务系统的影响；

 建模原则：从业务系统增量抽取、保留时间由业务需求决定,可分表进行周期存储,数据不做清洗转换与业务系统数据模型保持一致,按主题逻辑划分.

 DW层（数据仓库层）主要是从ODS数据库中,抽取出我们需要的数据按照事实和维度进行建模,比如星型模型和雪花模型；

 建模方式及原则:为支持数据重跑可额外增加数据业务日期字段,可按年月日进行分表,用增量ODS层数据和前一天DW相关表进行merge处理.

 DM层（数据集市）主要是做数据分析和报表可视化展示,根据DW层数据按照各种维度或多种维度组合把需要查询的一些事实字段进行汇总统计并作为单独的列进行存储,（宽表）满足一些特定查询,数据挖掘应用；

 建模方式及原则：尽量减少数据访问时计算,优化检索；维度建模,星型模型；事实拉宽,度量预先计算；分表存储.

粒度是什么

是数据代表的粗细程度,比如明细就是原子粒度,一天多少,粒度到天,一月多少,粒度到月,越聚合粒度越粗

订单明细 金额,以订单ID区别数据,粒度就是到每个订单,每日订单汇总金额,以每天的订单汇总,粒度到天,金额是维度.

数据口径是什么

这个是描述 维度 的粗细的一种表达方式 .

跟粒度差不多,比如 统计口径为：月

时间口径/时间粒度 从大到小年季度月周天 ,时分秒

性能优化

性能优化有三方面：索引优化,分区优化,执行计划hints优化

索引优化或者怎么建索引,哪些情况可以使用索引,哪些不适合索引,索引失效

一、索引优化—索引能够带来数据查询的效率提升,但是它会损失数据库性能

,单表查询就是创建索引,也就是给经常做为查询条件的列创建索引

,关联查询的时候（left join,或者inner join）,在从表中建立索引,索引的字段是查询的条件,主表中不用建立索引

子查询优化：子查询不要放在被驱动表；子查询不能放在select后面；先过滤大量数据之后在进行子查询

使用where条件能过滤完的条件,就不要使用having,因为where的执行有与having

使用适合索引的场景

①经常作为查询条件的

②多表查询时,与其他表关联的的字段,在外键建立索引

③经常作为排序,统计,或者分组的字段

数据可的数据量一定要大,数据量小的时候不如不建立

不适合索引的条件

①频繁更新的字段

②经常增删改,或者数据量较小的表

③where条件中使用不到的情况

④列中null值太多

索引失效

①没有使用到索引最左原则

②like查询%前置,如like‘%张’

③使用不等于符号,或者notin或者or

④出现了隐式转换,如字符串作为查询条件时候,没有使用单引号

存储过程的优化

之前我们用加索引的方式,后来加索引导致数据同步慢了,就没用,我们还用到加临时表的方式,让存储过程不那么复杂来优化,就多表关联的时候,我先把几张表先关联下做个结果放到临时表,然后再关联其他表,我还用到union all的方式来分开计算好指标,然后再合并的方式

一个存储过程可以更新多张表吗.

彼此没有依赖关系的可以同时写多个,相当于并行跑