Hive

1. hive 内部表和外部表的区别

未被external修饰的是内部表（managed table），被external修饰的为外部表（external table）

区别：

内部表数据由Hive自身管理，外部表数据由HDFS管理；

内部表数据存储的位置是hive.metastore.warehouse.dir（默认：/user/hive/warehouse），外部表数据的存储位置由自己制定（如果没有LOCATION，Hive将在HDFS上的/user/hive/warehouse文件夹下以外部表的表名创建一个文件夹，并将属于这个表的数据存放在这里）；

删除内部表会直接删除元数据（metadata）及存储数据；删除外部表仅仅会删除元数据，HDFS上的文件并不会被删除；

2. hive 有索引吗

Hive支持索引，但是Hive的索引与关系型数据库中的索引并不相同，比如，Hive不支持主键或者外键。

Hive索引可以建立在表中的某些列上，以提升一些操作的效率，例如减少MapReduce任务中需要读取的数据块的数量。

在可以预见到分区数据非常庞大的情况下，索引常常是优于分区的。

虽然Hive并不像事物数据库那样针对个别的行来执行查询、更新、删除等操作。它更多的用在多任务节点的场景下，快速地全表扫描大规模数据。但是在某些场景下，建立索引还是可以提高Hive表指定列的查询速度。（虽然效果差强人意）

索引适用的场景

适用于不更新的静态字段。以免总是重建索引数据。每次建立、更新数据后，都要重建索引以构建索引表。

Hive索引的机制如下：

hive在指定列上建立索引，会产生一张索引表（Hive的一张物理表），里面的字段包括，索引列的值、该值对应的HDFS文件路径、该值在文件中的偏移量;

v0.8后引入bitmap索引处理器，这个处理器适用于排重后，值较少的列（例如，某字段的取值只可能是几个枚举值）

因为索引是用空间换时间，索引列的取值过多会导致建立bitmap索引表过大。

但是，很少遇到hive用索引的。说明还是有缺陷or不合适的地方的。

3. 运维如何对hive进行调度

将hive的sql定义在脚本当中

使用azkaban或者oozie进行任务的调度

监控任务调度页面

4. ORC、Parquet等列式存储的优点

ORC和Parquet都是高性能的存储方式，这两种存储格式总会带来存储和性能上的提升

Parquet:

Parquet支持嵌套的数据模型，类似于Protocol Buffers，每一个数据模型的schema包含多个字段，每一个字段有三个属性：重复次数、数据类型和字段名。

重复次数可以是以下三种：required(只出现1次)，repeated(出现0次或多次)，optional(出现0次或1次)。每一个字段的数据类型可以分成两种：

group(复杂类型)和primitive(基本类型)。

Parquet中没有Map、Array这样的复杂数据结构，但是可以通过repeated和group组合来实现的。

由于Parquet支持的数据模型比较松散，可能一条记录中存在比较深的嵌套关系，如果为每一条记录都维护一个类似的树状结可能会占用较大的存储空间，因此Dremel论文中提出了一种高效的对于嵌套数据格式的压缩算法：Striping/Assembly算法。通过Striping/Assembly算法，parquet可以使用较少的存储空间表示复杂的嵌套格式，并且通常Repetition level和Definition level都是较小的整数值，可以通过RLE算法对其进行压缩，进一步降低存储空间。

Parquet文件是以二进制方式存储的，是不可以直接读取和修改的，Parquet文件是自解析的，文件

中包括该文件的数据和元数据。

ORC:

ORC文件是自描述的，它的元数据使用Protocol Buffers序列化，并且文件中的数据尽可能的压缩以降低存储空间的消耗。

和Parquet类似，ORC文件也是以二进制方式存储的，所以是不可以直接读取，ORC文件也是自解析的，它包含许多的元数据，这些元数据都是同构ProtoBuffer进行序列化的。

ORC会尽可能合并多个离散的区间尽可能的减少I/O次数。

ORC中使用了更加精确的索引信息，使得在读取数据时可以指定从任意一行开始读取，更细粒度的统计信息使得读取ORC文件跳过整个row group，ORC默认会对任何一块数据和索引信息使用ZLIB压缩，因此ORC文件占用的存储空间也更小。

在新版本的ORC中也加入了对Bloom Filter的支持，它可以进一

步提升谓词下推的效率，在Hive 1.2.0版本以后也加入了对此的支

持。

5. 数据建模用的哪些模型？

星型模型

星形模式(Star Schema)是最常用的维度建模方式。星型模式是以事实表为中心，所有的维度表直接连接在事实表上，像星星一样。

星形模式的维度建模由一个事实表和一组维表成，且具有以下特点：

a. 维表只和事实表关联，维表之间没有关联；

b. 每个维表主键为单列，且该主键放置在事实表中，作为两边连接的外键；

c. 以事实表为核心，维表围绕核心呈星形分布；

雪花模型

雪花模式(Snowflake Schema)是对星形模式的扩展。雪花模式的维度表可以拥有其他维度表的，虽然这种模型相比星型更规范一些，但是由于这种模型不太容易理解，维护成本比较高，而且性能方面需要关联多层维表，性能也比星型模型要低。所以一般不是很常用。

星座模型

星座模式是星型模式延伸而来，星型模式是基于一张事实表的，而星座模式是基于多张事实表的，而且共享维度信息。前面介绍的两种维度建模方法都是多维表对应单事实表，但在很多时候维度空间内的事实表不止一个，而一个维表也可能被多个事实表用到。在业务发展后期，绝大部分维度建模都采用的是星座模式。

6. 为什么要对数据仓库分层？

用空间换时间，通过大量的预处理来提升应用系统的用户体验（效率），因此数据仓库会存在大量冗余的数据。

如果不分层的话，如果源业务系统的业务规则发生变化将会影响整个数据清洗过程，工作量巨大。

通过数据分层管理可以简化数据清洗的过程，因为把原来一步的工作分到了多个步骤去完成，相当于把一个复杂的工作拆成了多个简单的工作，把一个大的黑盒变成了一个白盒，每一层的处理逻辑都相对简单和容易理解，这样我们比较容易保证每一个步骤的正确性，当数据发生错误的时候，往往我们只需要局部调整某个步骤即可。

7. 使用过Hive解析JSON串吗

hive 处理json数据总体来说有两个方向的路走

将json以字符串的方式整个入Hive表，然后通过使用UDF函数解析已经导入到hive中的数据，比如使用LATERAL VIEW json\_tuple的方法，获取所需要的列名。

在导入之前将json拆成各个字段，导入Hive表的数据是已经解析过得。这将需要使用第三方的

SerDe。

8. sort by 和 order by 的区别

order by 会对输入做全局排序，因此只有一个reducer（多个reducer无法保证全局有序）只有一个reducer，会导致当输入规模较大时，需要较长的计算时间。

sort by不是全局排序，其在数据进入reducer前完成排序.

因此，如果用sort by进行排序，并且设置mapred.reduce.tasks>1， 则sort by只保证每个reducer的输出有序，不保证全局有序。

9. 怎么排查是哪里出现了数据倾斜

10. 数据倾斜怎么解决

11. hive 小文件过多怎么解决

解决hive小文件过多问题

12. hive优化有哪些？

数据存储及压缩。

针对hive中表的存储格式通常有orc和parquet，压缩格式一般使用snappy。相比与textfile格式表，orc占有更少的存储。因为hive底层使用MR计算架构，数据流是hdfs到磁盘再到hdfs，而且会有很多次，所以使用orc数据格式和snappy压缩策略可以降低IO读写，还能降低网络传输量，这样在一定程度上可以节省存储，还能提升hql任务执行效率；

通过调参优化。

并行执行，调节parallel参数；

调节jvm参数，重用jvm；

设置map、reduce的参数；开启strict mode模式；

关闭推测执行设置。

有效地减小数据集将大表拆分成子表；结合使用外部表和分区表。

SQL优化

大表对大表：尽量减少数据集，可以通过分区表，避免扫描全表或者全字段；

大表对小表：设置自动识别小表，将小表放入内存中去执行。