数据湖计算 DLC

常见问题

腾讯云

【联系我们】

我们致力于为您提供个性化的售前购买咨询服务，及相应的技术售后服务，任何问题请联系 4009100100或95716。

文档目录

常见问题
权限类常见问题

最近更新时间：2023-10-12 16:38:11

为什么要同时开通 CAM 权限和 DLC 权限?

类型	说明
CAM 权限	开通子账号访问 DLC 权限。
DLC 权限	添加 CAM 子账号为 DLC 用户账号后，可对数据库表与引擎的权限进行配置管理。

为什么我开通了子账号，还是使用不了数据探索（数据作业）？

账号开通后，DLC 管理员需要在数据湖计算 DLC 的 权限管理 功能里，开通子账号的角色属性、引擎使用权限和数据的库表读写权限。

引擎类常见问题

最近更新时间：2025-06-12 12:00:42

共享引擎和独享引擎的区别是什么？

引擎类型	说明	计费模式	特点
共享引擎	为当前地域下所有用户公共使用的引擎	**按量计费**：按扫描量计费，不使用不产生任何费用	1. 无需配置即可使用 2. 适合数据量较小、临时数据计算场景。
独享引擎	为用户独享引擎资源	**按量计费**：按 CU 量计费，没有任务时可挂起集群，挂起时不产生任何费用	1. 资源独享，支持配置资源规模，弹性伸缩 2. 适合有一定任务量但任务周期不规律的数据计算场景
		包年包月：按 CU 量计费，集群无需等待随时可用。弹性部分按量计费。	1. 资源独享，支持配置资源规模，弹性伸缩 2. 适合任务量大且稳定的数据计算场景

一个集群，支持多少任务并行？可以调整吗？

1个集群，任务并行数是5，如果需要调整，可以在 数据引擎 功能中，选择需要修改的引擎，单击**规格配置**进行调整。

为什么任务实际使用 CU 核数会小于引擎的集群规模中指定的 CU 核数？

以下几种情况都可能导致实际使用 CU 核数小于引擎的集群规模中指定的 CU 数：

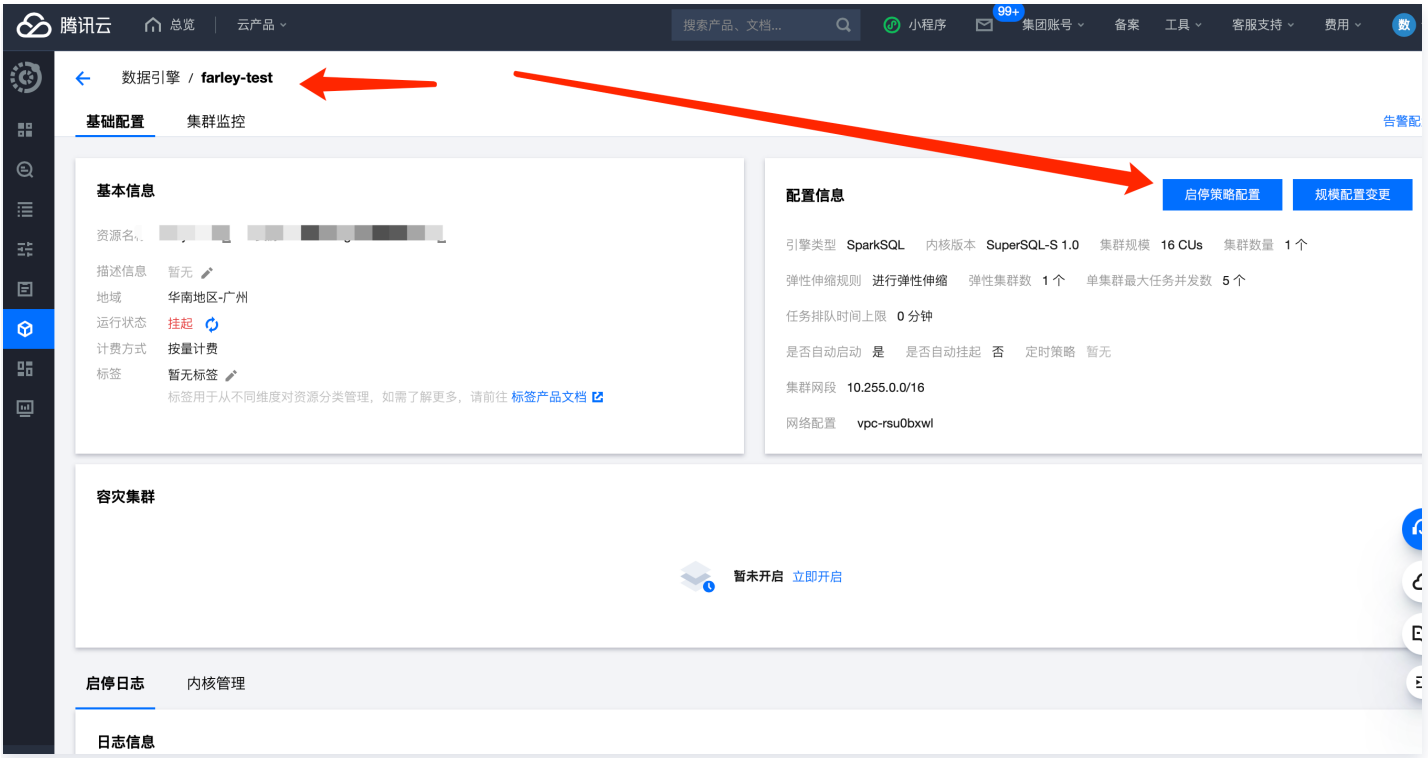
1. 集群中有其他任务正在执行。
2. 批作业集群指定的 driver 资源 + executor 资源总数小于集群规模。
3. 按量计费集群，使用时才会发起资源申请，当 CU 核数较多时，不能确保资源完全满足申请的 CU 数。
4. 批处理作业指定了增强型网络配置用于打通其他 VPC 网络，但是其他 VPC 网络的 IP 数不足以启动所有的 executor。

DLC 集群是否可以访问同地域的其他 VPC 下的 IP/服务？是否可以访问外网？

DLC 引擎可以访问同地域下的其他 VPC，需要在 **数据引擎** > **网络配置** 中创建一个网络配置和目标 VPC 打通，并在批处理作业中指定使用该网络配置。DLC 引擎默认是无法访问外网的。但是通过增强型网络配置，并且目标 VPC 配置了可以访问外网的路由规则，那么 DLC 可以通过增强型网络配置访问外网。

如何修改集群的自动启停时间?

在 数据引擎 功能中，单击引擎列表中需要修改的**引擎名称**，进入修改页，单击**启停策略配置**进行修改。



若在任务运行中进行集群变配，是否会导致任务失败?

我们为您罗列了各类引擎在常见任务进行中，若发起变配带来的任务影响，请您参考：

引擎类型	购买方式	是否会影响任务失败
SuperSQL-Spark作业	按量计费	任务不会受影响
	包年包月	**当发起集群规格变配时**： • 扩容集群规格不会影响任务 • 缩容集群规格，会在流程中等待pod运行结束后再去隔离/销毁缩容的机器，如果任务运行时间长可能导致流程卡住，导致任务重启
SuperSQL-SparkSQL	按量计费	**当发起集群规格变配时**，会导致任务重启
	包年包月	**当发起集群规格变配 或 减少集群数量时**，会导致任务重启
SuperSQL-Presto	按量计费	**当发起集群规格变配时**，会导致任务重启
	包年包月	**当发起集群规格变配 或 减少集群数量时**，会导致任务重启

如何排查标准引擎运行任务长时间未正常执行的原因?

当您的任务长时间处于“初始化”、“启动中”或“排队中”状态时，可能导致任务无法正常执行。请根据以下步骤和任务类型，逐步排查问题原因，帮助您快速定位并解决问题。

1. 区分细分任务类型

步骤：历史任务实例 > 历史任务列表 > 任务类型、资源组名称，结合任务类型及资源组名称是否有值来区分细分任务类型。

细分任务类型	任务类型	资源组名称
SQL 任务	SQL	资源组名称有值
交互式 SQL 任务	SQL	资源组名称无值（--）
Spark 批流任务	作业	资源组名称无值（--）

2. SQL 任务排查步骤

2.1 判断引擎和网关状态

1. 进入引擎列表，确认引擎状态是否为“运行中”或“就绪”。
2. 通过管理页面查看网关状态，确认网关是否“运行中”。
3. 如果引擎或网关状态异常，说明正在执行其他流程，请等待流程结束。
4. 若状态超过10分钟未恢复，请 提交工单 联系技术支持。

2.2 判断计算资源是否处于冷启动

1. 如果任务提交到挂起状态的资源组，资源组可能正在启动，通常需要3~5分钟。
2. 进入资源组页面确认资源组状态是否为“启动中”。
3. 若超过5分钟仍未启动，继续排查其他原因。

4. 如果资源组状态非“运行中”或“启动中”，说明资源组正在执行其他操作，请等待，超过10分钟未恢复请 提交工单 。

2.3 资源组是否分配到足够资源

1. 资源组启动超过10分钟仍未成功，可能是引擎资源被占满，资源组无法分配到足够资源。
2. 在资源组列表查看资源需求（如CU数），确认资源组最低所需资源。
3. 在引擎列表点击引擎名称，进入“集群监控”查看资源使用情况。
4. 若“已占用集群规格”达到总规格，剩余资源不足，资源组无法启动。
5. 解决方案：释放其他资源（暂停或取消其他资源组或批流作业任务）。

2.4 是否等待依赖任务完成

1. DLC 默认一次提交的多个 SQL 任务串行执行，前置任务未完成，后续任务会处于初始化状态。
2. 请确认是否存在依赖任务未完成导致排队。

2.5 是否达到资源组任务并发数上限

1. 资源组默认任务并发数上限为5。
2. 超过并发数的任务会处于排队状态。
3. 在资源组详情页面查看并发数配置，可根据需要调整。
4. 在历史任务实例页面查看当前运行任务数，确认是否达到上限。

3. 交互式 SQL 任务（BatchSQL 任务）排查步骤

3.1 判断引擎和网关状态

参考 SQL 任务排查步骤 > 判断引擎和网关状态 的操作步骤。

3.2 判断资源是否充足

1. 在引擎列表“集群监控”查看资源使用情况。
2. 若资源不足，任务无法启动。
3. 解决方案同 SQL 任务，释放其他占用资源。

3.3 冷启动时间

- 交互式 SQL 任务每次提交均需拉起新资源，冷启动时间一般为3~5分钟。
- 超过5分钟仍未启动，排查是否存在异常。

4. Spark 批流作业排查步骤

4.1 判断引擎和网关状态

1. 参考 SQL 任务排查步骤 > 判断引擎和网关状态 的操作步骤。
2. 资源启动一般需要3~5分钟，等待后仍未运行，继续排查。

4.2 判断引擎资源是否充足

1. 在历史任务实例页面查看任务所需资源（Driver 和 Executor 的 CU 数）。
2. 在引擎列表“集群监控”查看资源使用情况。
3. 若资源不足，无法启动批流作业。
4. 解决方案同 SQL 任务，释放其他占用资源。

功能类常见问题

最近更新时间：2023-07-26 17:00:41

外表和原生表的区别是什么?

- 原生表：是您存放于 DLC 托管存储上的表，默认为 Iceberg 格式。使用原生表无需关注 Iceberg 底层文件，而且具备数据优化等能力帮助构建数据湖。
- 外表：文件为您自己账号下的 COS 桶或其他第三方数据存储的表。DLC 可以直接建立外表进行分析，无需额外加载数据。

数据湖计算 DLC 支持数据调度吗?

如您有数据调度的需求，可前往咨询并开通 数据开发治理平台 Wedata 。

数据湖计算 DLC 是否支持自助上传 jar 包，自定义 spark 函数?

支持。

元数据加速桶支持 CDN 吗?

加速桶不支持，普通桶支持。

元数据加速桶与普通 COS 桶的区别是什么?

元数据加速桶与普通桶的区别在于：对大数据分析中常见的 rename，list 文件操作等会有明显的加速效果。

数据存储需要绑定引擎权限吗?

元数据加速桶需要绑定引擎权限，普通的对象存储不需要绑定引擎权限。

一个 COS 的 bucket 可以绑定多个 DLC 数据引擎么?

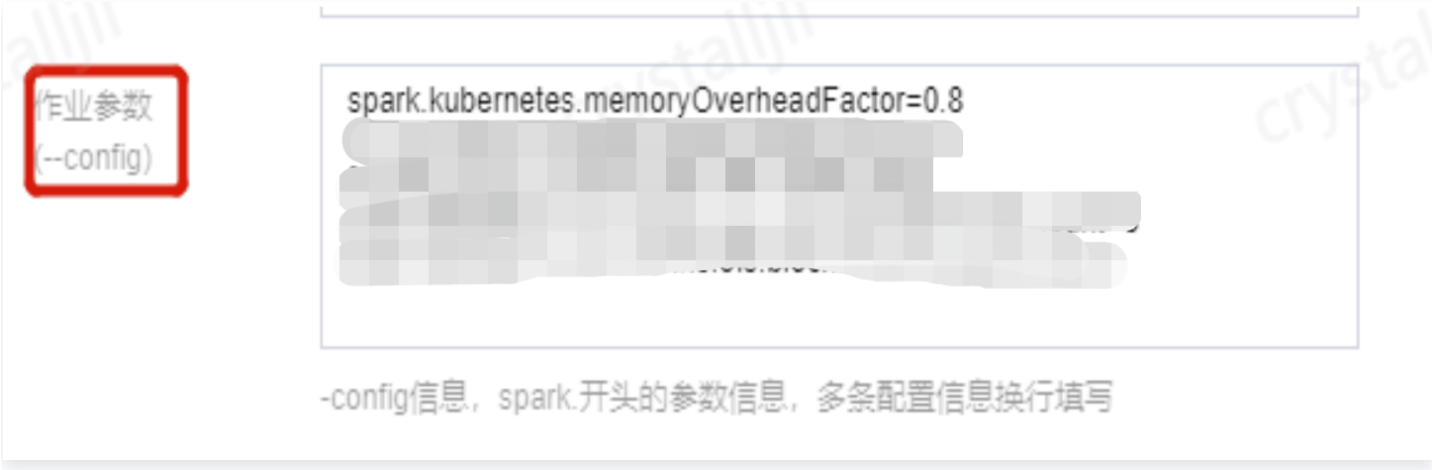
可以。

Spark 作业类常见问题

最近更新时间：2024-11-15 15:09:52

PySpark 任务数据倾斜导致 python+jvm 内存占用超过 K8s request 内出现 OOMkilled?

问题描述：PySpark 任务执行时，executor 日志出现“k8s执行OOMKiiled”，使用内存超过 k8s 限制的内存。
原因分析：K8s 申请的内存是根据 Spark executor 内存乘以 memoryOerheadFactor计算出来的，如果 Python 处理的数据有倾斜，或单条数据过大，可能导致使用内存超过 K8s 分配的内存。
解决方案：添加任务参数 spark.kubernetes.memoryOverheadFactor=0.8，默认值0.4。
操作步骤：登录 DLC 控制台，进入**数据作业（Spark 作业）> 编辑作业**，按照如下配置：



Insert into/overwrite 后如何自动添加 repartition 命令对数据做分区以减少小文件数量?

解决方案：开启自动重分区，配置如下参数：

```
spark.sql.adaptive.enabled: true
spark.sql.adaptive.insert.repartition: true
spark.sql.adaptive.insert.repartition.forceNum: 300 （指定了具体需要分区的值）
```

操作步骤：

- 程序内配置到 SparkConf：

```
val spark: SparkSession = args(0) match {
  case "1" =>
    SparkSession.builder().getOrCreate()
  case "2" =>
    SparkSession.builder().config("spark.sql.adaptive.insert.repartition", "true").getOrCreate()
  case "3" =>
    SparkSession.builder().config("spark.sql.adaptive.insert.repartition", "true")
      .config("spark.sql.adaptive.coalescePartitions.enabled", "true")
      .config("spark.sql.adaptive.coalescePartitions.initialPartitionNum", "300")
      .getOrCreate()
  case "4" =>
    SparkSession.builder().config("spark.sql.adaptive.insert.repartition.forceNum", "300").getOrCreate()
  case _ =>
    SparkSession.builder().getOrCreate()
}
```

- 程序内设置 SQL SET：

PySpark 任务高并发写 COS 存储桶时返回503错误?

问题描述：PySpark 任务高并发写 COS 存储桶时，executor 有非常多的 COS 返回503错误。
问题原因：Spark 任务写 COS 时的并行核数为 *fs.cosn.trsf.fs.ofs.data.transfer.thread.count 指定 。例如，4096核下不做调优，默认并发度为4096*32=131072，导致 COS 瓶颈。
解决方案：

1. COS 新建一个元数据加速桶，避免 spark 任务写时的 list 和 rename 超频。
2. 通过 COS 调整元数据加速桶带宽限制。
3. 任务添加以下参数降低高并行度时对 COS 访问压力过大。

```
fs.cosn.trsf.fs.ofs.data.transfer.thread.count=8
fs.cosn.trsf.fs.ofs.block.max.file.cache.mb=0
spark.hadoop.fs.cosn.trsf.fs.ofs.data.transfer.thread.count=8
```

```
spark.hadoop.fs.cosn.trsf.fs.ofs.block.max.file.cache.mb=0
```

常用数据治理 SQL 有哪些?

- 关闭库治理 SQL

```
ALTER DATABASE DataLakeCatalog.demo_db
SET
  DBPROPERTIES (
    'dlc.ao.data.govern.inherit' = 'none',
    'dlc.ao.merge.data.enable' = 'disable',
    'dlc.ao.expired.snapshots.enable' = 'disable',
    'dlc.ao.remove.orphan.enable' = 'disable',
    'dlc.ao.merge.manifests.enable' = 'disable'
  )
```

- 开启库治理 SQL

```
ALTER DATABASE DataLakeCatalog.db_name
SET
  DBPROPERTIES (
    'dlc.ao.data.govern.inherit' = 'none',
    'dlc.ao.merge.data.enable' = 'enable',
    'dlc.ao.merge.data.engine' = 'bda-sinker',
    'dlc.ao.merge.data.min-input-files' = '10',
    'dlc.ao.merge.data.target-file-size-bytes' = '536870912',
    'dlc.ao.merge.data.interval-min' = '90',
    'dlc.ao.expired.snapshots.enable' = 'enable',
    'dlc.ao.expired.snapshots.engine' = 'bda-sinker',
    'dlc.ao.expired.snapshots.retain-last' = '5',
    'dlc.ao.expired.snapshots.before-days' = '2',
    'dlc.ao.expired.snapshots.max-concurrent-deletes' = '4',
    'dlc.ao.expired.snapshots.interval-min' = '150',
    'dlc.ao.remove.orphan.enable' = 'enable',
    'dlc.ao.remove.orphan.engine' = 'bda-sinker',
    'dlc.ao.remove.orphan.before-days' = '3',
    'dlc.ao.remove.orphan.max-concurrent-deletes' = '4',
    'dlc.ao.remove.orphan.interval-min' = '600',
    'dlc.ao.merge.manifests.enable' = 'enable',
    'dlc.ao.merge.manifests.engine' = 'bda-sinker',
    'dlc.ao.merge.manifests.interval-min' = '1440'
  )
```

- 关闭表治理 SQL

```
  ALTER TABLE
  `DataLakeCatalog`.`db_name`.`tb_name`
SET
  TBLPROPERTIES (
    'dlc.ao.data.govern.inherit' = 'none',
    'dlc.ao.merge.data.enable' = 'disable',
    'dlc.ao.expired.snapshots.enable' = 'disable',
    'dlc.ao.remove.orphan.enable' = 'disable',
    'dlc.ao.merge.manifests.enable' = 'disable'
  )
```

- 开启继承库治理 SQL

```
  ALTER TABLE `DataLakeCatalog`.`db_name`.`tb_name`
  SET TBLPROPERTIES ('dlc.ao.data.govern.inherit' = 'default')
```

- 开启表治理 SQL

```
  ALTER TABLE
  `DataLakeCatalog`.`db_name`.`tb_name`
SET
  TBLPROPERTIES (
    'dlc.ao.data.govern.inherit' = 'none',
    'dlc.ao.merge.data.enable' = 'enable',
    'dlc.ao.merge.data.engine' = 'bda-sinker',
    'dlc.ao.merge.data.min-input-files' = '10',
    'dlc.ao.merge.data.target-file-size-bytes' = '536870912',
    'dlc.ao.merge.data.interval-min' = '90',
    'dlc.ao.expired.snapshots.enable' = 'enable',
    'dlc.ao.expired.snapshots.engine' = 'bda-sinker',
    'dlc.ao.expired.snapshots.retain-last' = '5',
    'dlc.ao.expired.snapshots.before-days' = '2',
    'dlc.ao.expired.snapshots.max-concurrent-deletes' = '4',
    'dlc.ao.expired.snapshots.interval-min' = '150',
    'dlc.ao.remove.orphan.enable' = 'enable',
    'dlc.ao.remove.orphan.engine' = 'bda-sinker',
    'dlc.ao.remove.orphan.before-days' = '3',
    'dlc.ao.remove.orphan.max-concurrent-deletes' = '4',
    'dlc.ao.remove.orphan.interval-min' = '600',
    'dlc.ao.merge.manifests.enable' = 'enable',
```

```
    'dlc.ao.merge.manifests.engine' = 'bda-sinker',
    'dlc.ao.merge.manifests.interval-min' = '1440'
  )
```

- 不指定 Where 条件全表合并 SQL

```
  CALL `DataLakeCatalog`.`system`.`rewrite_data_files`(
  `table` => 'tb_name',
  `options` => map(
    'min-input-files',
    '10',
    'target-file-size-bytes',
    '536870912',
    'delete-file-threshold',
    '1',
    'max-concurrent-file-group-rewrites',
    '20'
  )
)
```

- 支持 Where 条件增量合并 SQL

```
 CALL `DataLakeCatalog`.`system`.`rewrite_data_files`(
  `table` => 'tb_name',
  `options` => map(
    'min-input-files',
    '10',
    'target-file-size-bytes',
    '536870912',
    'delete-file-threshold',
    '1',
    'max-concurrent-file-group-rewrites',
    '20'
  ),
  `where` => 'field_date > "2022-01-01" and field_date <= "2023-01-
01"'
)
```

- 快照过期 SQL

```
  CALL `DataLakeCatalog`.`system`.`expire_snapshots`(
  `table` => 'tb_name',
```

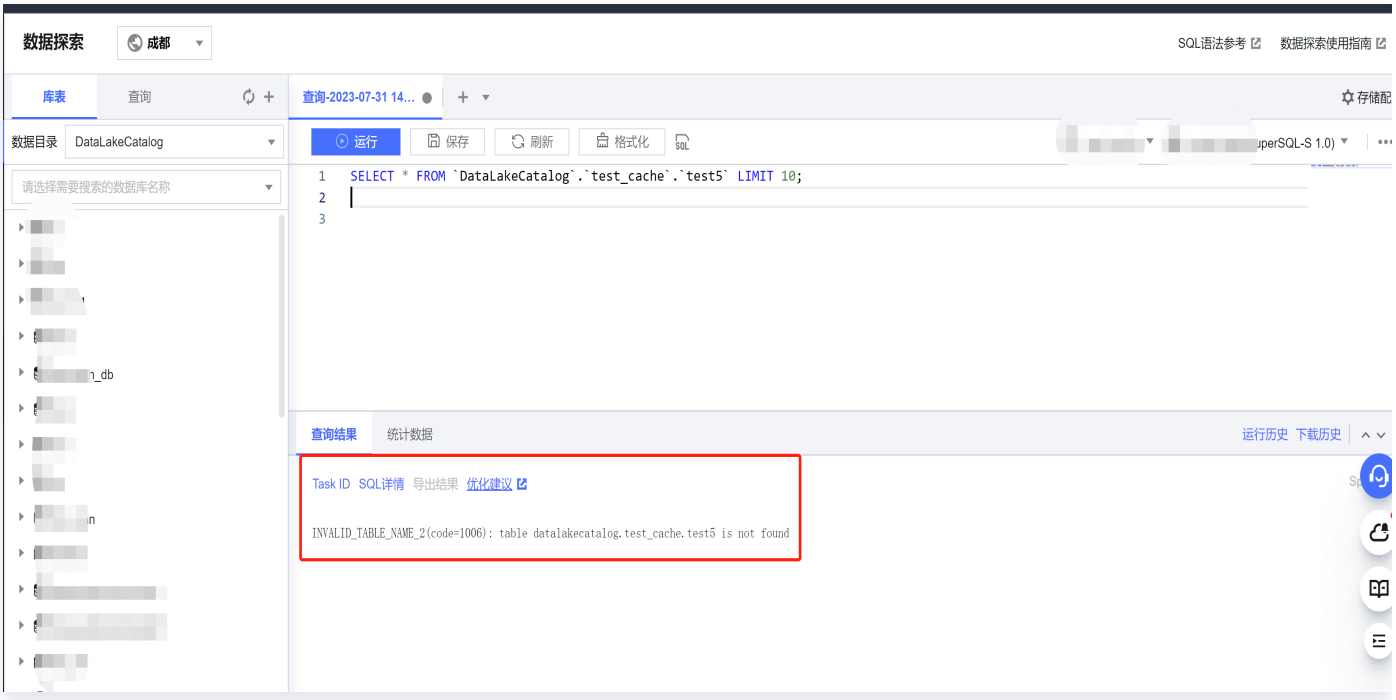
```
  older_than => TIMESTAMP '2023-02-28 16:06:35.000',
  retain_last => 1,
  max_concurrent_deletes => 4,
  stream_results => true
)
```

如何查看 SQL 执行计划和 SQL 执行的日志?

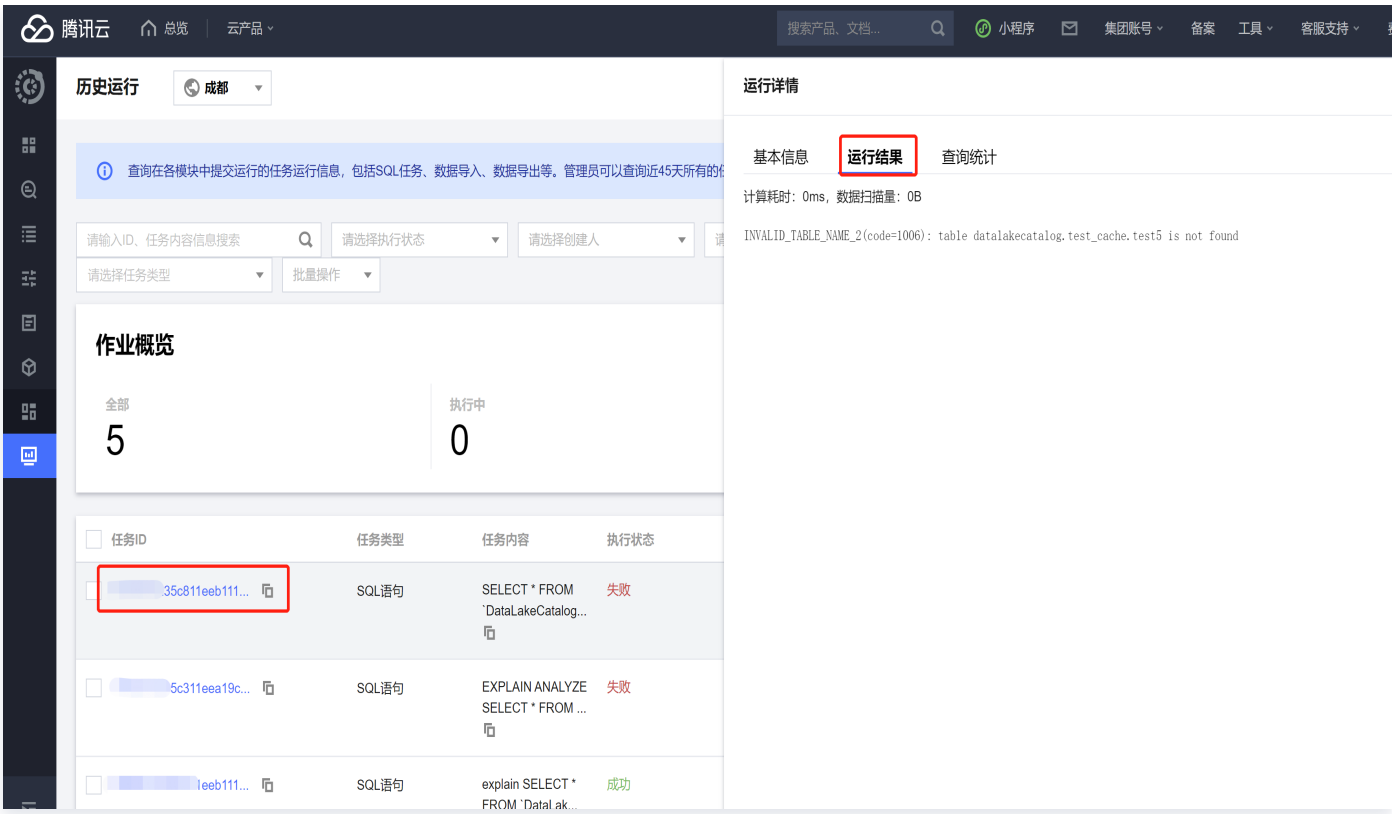
查看 SQL 执行计划：用 explain 关键字在数据探索中查看 SQL 执行的物理计划，explain 详细使用请参考 SQL 统一语法 > EXPLAIN 。

查看SQL执行日志：

1. 数据探索执行 SQL，运行结果展示 SQL 执行日志。



2. DLC 控制台 > 数据运维 > 历史运行可以查看 SQL 执行日志。



CAST 未自动转换精度导致数据写入失败?

问题描述：hive sql 迁移 spark sql 时，报错 Cannot safely cast 'class_type': string to bigint。
问题定位：Spark 3.0.0 开始，Spark SQL 在处理类型转换时有 3 种安全策略：

- ANSI：不允许 Spark 进行某些不合理的类型转换，如：string 转换成 timestamp。
- LEGACY ： 允许 Spark 进行类型强制转换，只要它是有效的 Cast 操作。
- STRICT ： 不允许 Spark 进行任何可能有损精度的转换。默认策略是 ANSI。

解决方案：修改策略为 LEGACY，设置 spark.sql.storeAssignmentPolicy=LEGACY。

QUERY_PROGRESS_UPDATE_ERROR(code=3060): Failed to update statement progress 错误

问题描述：数据探索中提交 spark sql 任务，执行过程中，提示 Failed to Update statement progress 错误。
问题定位：当有多个 Spark SQL 任务提交时，需要持续的异步跟进每个 SQL 的执行进度，这里异步处理的队列有限制，默认值是100（2024.1.14以后的版本更新为300）。所以当某个任务被提交后一直没有执行完成，而后续新增的任务超过了队列上限会导致该错误。如果您出现这种错误，一般表示该 SQL 任务可能是个长尾任务，需要关注对其他任务资源占用是否合理。
解决方案：可以在引擎上调整配置 livy.rsc.retained-statements，调整到大于默认值的值。注意调整后引擎会重启。具体值可以根据任务的并发量来设置，该参数对集群影响较小，同时提交的 SQL 并发量达到100-200/min 时，该参数调整到6000也实测没有影响。