(19)国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 116610752 A
(43)申请公布日 2023.08.18

(21)申请号 202310579061.7

(22)申请日 2023.05.19

(71)申请人 新华三技术有限公司
地址 310052 浙江省杭州市滨江区长河路466号

(72)发明人 高飞

(51)Int.Cl.
G06F 16/27(2019.01)
G06F 16/23(2019.01)

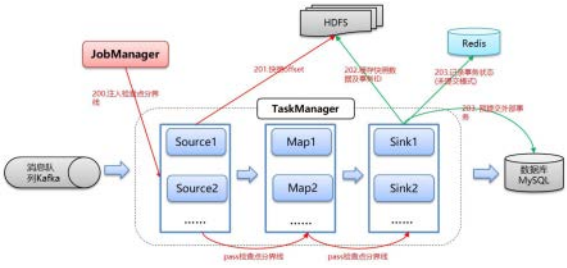
权利要求书3页 说明书11页 附图4页

(54)发明名称

事务性分布式数据同步方法、装置、系统及存储介质

(57)摘要

本发明提供一种事务性分布式数据同步方法、装置、系统及存储介质，用于解决使用Flink从数据源同步数据到无主键目的数据库表的过程中无法保证数据一致性的技术问题。本发明将Flink集群进行分布式数据同步的过程分成预提交阶段和正式提交阶段，在预提交阶段创建快照并以数据库事务的预提交模式将快照数据提交到目的数据库，在快照创建成功后通过正式提交阶段将快照数据正式提交到目的数据库表。本发明技术方案中不仅记录快照的处理状态还会记录事务的提交状态，并在Flink故障的情况下，根据记录的快照和事务状态进行数据同步任务的恢复。通过本发明能够实现Flink从分布式数据源到无主键关系数据库表的数据同步的一致性。



CN 116610752 A

1.一种事务性分布式数据同步方法,其特征在于,该方法应用于Flink集群,该方法包括:

通过Flink集群将分布式数据源中的数据同步到目的无主键关系数据库表即目的数据库表中的过程分为预提交阶段和正式提交阶段;

在预提交阶段中,完成快照创建以及快照数据的预提交,输出算子任务sink负责将快照数据预提交到目的数据库表中并记录快照完成状态及事务预提交状态;

在正式提交阶段中,作业管理器Jobmanager负责在快照创建成功后指令sink算子任务正式提交快照数据到目的数据库表中并记录事务提交结果。

2.根据权利要求1所述的方法,其特征在于,所述预提交阶段的步骤包括:

作业管理器Jobmanager通过注入检查点分界线标记当前同步的快照数据;

输入算子任务source在检测到检查点分界线后记录快照偏移;

sink算子任务在检测到检查点分界线后,负责缓存当前快照数据、预提交当前快照数据到目的数据库表以及记录事务预提交状态;

各算子任务在成功完成快照数据的处理后向Jobmanager发送快照完成反馈,sink算子任务在成功完成快照数据的预提交的处理后向Jobmanager发送预提交结果反馈;

当Jobmanager接收到所有算子任务的快照完成反馈和sink算子任务的预提交结果反馈后,记录快照成功状态。

3.根据权利要求2所述的方法,其特征在于,所述正式提交阶段的步骤包括:

在Jobmanager根据快照完成反馈和预提交结果反馈判定创造创建成功及预提交成功后,向所有算子任务发送快照完成消息;

输出算子任务sink接收到快照完成消息后正式将快照数据提交到目的数据库表中并记录事务正式提交成功与否的事务提交结果。

4.根据权利要求2所述的方法,其特征在于,若Flink集群在预提交阶段产生故障且导致快照创建未成功时,所述方法还包括如下故障恢复步骤:

Jobmanager从状态存储后端获取最近一次创建成功的快照的快照信息;

Jobmanager协调调度Taskmanager恢复数据同步任务;

sink算子任务根据所述快照信息从状态存储后端获取最近一次创建成功的快照的事务标识ID及快照数据;

sink算子任务根据事务标识ID获取事务提交状态,验证所述最近一次创建成功的快照的快照数据是否成功提交到目的数据库表中,若成功提交,则source算子任务获取下一次待消费的快照数据偏移,重新开始数据同步过程。

5.根据权利要求3所述的方法,其特征在于,若Flink集群在预提交阶段产生故障且最近一次快照创建成功时,所述方法还包括如下故障恢复步骤:

Jobmanager获取最近一次创建成功的快照的快照信息;

Jobmanager协调调度Taskmanager恢复数据同步任务的执行;

sink算子任务根据所述快照信息获取最近一次创建成功的快照的事务标识ID及快照数据;

sink算子任务根据事务ID获取事务提交状态,判断对应快照数据是否正式提交到目的数据库表中;

在事务未正式提交的状态下，sink算子任务将最近一次创建成功的快照的快照数据提交到目的数据库表中；

source算子任务获取下一待消费的快照数据偏移继续进行数据同步。

6.根据权利要求1所述的方法，其特征在于，

所述Jobmanager和各算子任务在状态存储后端中记录快照信息，所述状态存储后端为分布式文件系统；

所述分布式数据源为分布式发布订阅消息系统。

7.根据权利要求1所述的方法，其特征在于，

所述sink算子任务将事务状态记录在分布式内存数据库中。

8.一种事务性分布式数据同步系统，其特征在于，该系统包括Flink集群、状态存储后端、分布式内存数据库；

所述Flink集群包括作业管理器Jobmanager、任务管理器Taskmanager，Taskmanager中至少运行有输入算子任务source和输出算子任务sink；

所述Flink集群用于将分布式数据源中的数据同步到目的无主键关系数据库表即目的数据库表中，所述数据同步过程分为预提交阶段和正式提交阶段；

在预提交阶段中，完成快照创建以及快照数据的预提交，输出算子sink算子任务负责将快照数据预提交到目的数据库表中并记录快照完成状态及事务预提交状态；

在正式提交阶段中，作业管理器Jobmanager负责在快照创建成功后指令sink算子任务正式提交快照数据到目的数据库表中并记录事务提交结果；

所述状态存储后端用于存储快照信息和缓存快照数据；

所述分布式内存数据库用于存储事务状态。

9.根据权利要求8所述的系统，其特征在于，在所述预提交阶段中：

所述Jobmanager通过注入检查点分界线标记当前同步的快照数据；

所述source算子任务在检测到检查点分界线后记录快照偏移；

所述sink算子任务在检测到检查点分界线后，负责缓存当前快照数据、预提交当前快照数据到目的数据库表以及记录事务预提交状态；

各算子任务在成功完成快照数据的处理后向Jobmanager发送快照完成反馈，sink算子任务在成功完成快照数据的预提交的处理后向Jobmanager发送预提交结果反馈；

所述Jobmanager还用于接收到所有算子任务的快照完成反馈和sink算子任务的预提交结果反馈，记录快照成功状态。

10.根据权利要求9所述的系统，其特征在于，在所述正式提交阶段中：

所述Jobmanager还用于在根据快照完成反馈和预提交结果反馈判定创造创建成功及预提交成功后，向所有算子任务发送快照完成消息；

所述输出算子任务sink接收到快照完成消息后正式将快照数据提交到目的数据库表中并记录事务正式提交成功与否的事务提交结果。

11.一种事务性分布式数据同步装置，其特征在于，该装置用于通过Flink集群将分布式数据源中的数据同步到目的无主键关系数据库表即目的数据库表中，该装置包括：

预提交模块，用于完成所述数据同步过程中的预提交阶段的处理步骤，完成快照创建以及快照数据的预提交；其中，通过输出算子任务sink将快照数据预提交到目的数据库表

中并记录快照完成状态及事务预提交状态；

正式提交模块，用于完成所述数据同步过程中的正式提交阶段的处理步骤，将快照数据以事务模式正式提交的目的数据库表中；其中，通过作业管理器Jobmanager在快照创建成功后指令sink算子任务正式提交快照数据到目的数据库表中并记录事务提交状态并记录事务提交结果。

12.一种电子设备，其特征在于，包括处理器、通信接口、存储介质和通信总线，其中，处理器、通信接口、存储介质通过通信总线完成相互间的通信；

存储介质，用于存放计算机程序；

处理器，用于执行存储介质上所存放的计算机程序时，实施权利要求1-7中任一项所述的方法步骤。

13.一种存储介质，其上存储有计算机程序，其特征在于，所述计算机程序当被处理器执行时实施如权利要求1至7中任一项所述的方法。

事务性分布式数据同步方法、装置、系统及存储介质

技术领域

[0001] 本发明涉及通信及云计算技术领域，尤其涉及一种事务性分布式数据同步方法、装置、系统及存储介质。

背景技术

[0002] Kafka是由Apache软件基金会开发的一个开源流处理平台，由Scala和Java编写。Kafka是一种高吞吐量的分布式发布订阅消息系统。实现从分布式发布订阅消息系统(例如Kafka)到关系数据库(例如MySQL)的数据同步是一种常见的需求。

[0003] Flink是一个框架和分布式处理引擎，用于在无界(流处理)和有边界(批处理)数据流上进行有状态的计算。Flink是实现Kafka到关系数据库的数据同步的常用工具。Flink中通用的数据处理流程包括一个或多个输入算子任务(source)、数据处理算子任务(transform)以及输出算子任务(sink)。

[0004] 在使用Flink将Kafka集群中的数据同步到关系数据库的处理流程中，由于Flink集群的分布式特性、人工开发的处理逻辑等原因通常很容易发生集群的重启、任务的重新调度等异常现象，通常集群发生异常后会带来源和目的数据一致性问题，例如输出数据库中存在重复业务数据，输出数据库中存在丢失业务数据等。

[0005] 通过Flink的流处理能力和内在机制，如检查点(checkpoint)快照等技术可提升Kafka到MySQL数据同步的一致性，尤其在目的端MySQL库表中有对应的主键的情况下，这样可以通过MySQL中的幂等性SQL实现同步数据的一致性。但是在目的端MySQL库表中没有定义主键，使用Flink集群从数据源Kafka中同步数据至MySQL时在集群异常情况下则会容易产生数据缺失或数据重复的问题。

发明内容

[0006] 有鉴于此，本发明提供一种事务性分布式数据同步方法、装置、系统及存储介质，用于解决使用Flink从数据源同步数据到无主键目的数据库表的过程中，无法保证数据一致性的技术问题。

[0007] 基于本发明实施例的一方面，本发明提供了一种事务性分布式数据同步方法，该方法应用于Flink集群，该方法包括：

[0008] 通过Flink集群将分布式数据源中的数据同步到目的无主键关系数据库表即目的数据库表中的过程分为预提交阶段和正式提交阶段；

[0009] 在预提交阶段中，完成快照创建以及快照数据的预提交，输出算子任务sink负责将快照数据预提交到目的数据库表中并记录快照完成状态及事务预提交状态；

[0010] 在正式提交阶段中，作业管理器Jobmanager负责在快照创建成功后指令sink算子任务正式提交快照数据到目的数据库表中并记录事务提交结果。

[0011] 进一步地，所述预提交阶段的步骤包括：

[0012] 作业管理器Jobmanager通过注入检查点分界线标记当前同步的快照数据；

[0013] 输入算子任务source在检测到检查点分界线后记录快照偏移；

[0014] sink算子任务在检测到检查点分界线后，负责缓存当前快照数据、预提交当前快照数据到目的数据库表以及记录事务预提交状态；

[0015] 各算子任务在成功完成快照数据的处理后向Jobmanager发送快照完成反馈，sink算子任务在成功完成快照数据的预提交的处理后向Jobmanager发送预提交结果反馈；

[0016] 当Jobmanager接收到所有算子任务的快照完成反馈和sink算子任务的预提交结果反馈后，记录快照成功状态。

[0017] 进一步地，所述正式提交阶段的步骤包括：

[0018] 在Jobmanager根据快照完成反馈和预提交结果反馈判定创造创建成功及预提交成功后，向所有算子任务发送快照完成消息；

[0019] 输出算子任务sink接收到快照完成消息后正式将快照数据提交到目的数据库表中并记录事务正式提交成功与否的事务提交结果。

[0020] 进一步地，若Flink集群在预提交阶段产生故障且导致快照创建未成功时，所述方法还包括如下故障恢复步骤:

[0021] Jobmanager从状态存储后端获取最近一次创建成功的快照的快照信息；

[0022] Jobmanager协调调度Taskmanager恢复数据同步任务；

[0023] sink算子任务根据所述快照信息从状态存储后端获取最近一次创建成功的快照的事务标识ID及快照数据；

[0024] sink算子任务根据事务标识ID获取事务提交状态，验证所述最近一次创建成功的快照的快照数据是否成功提交到目的数据库表中，若成功提交，则source算子任务获取下一次待消费的快照数据偏移，重新开始数据同步过程。

[0025] 进一步地，若Flink集群在预提交阶段产生故障且最近一次快照创建成功时，所述方法还包括如下故障恢复步骤:

[0026] Jobmanager获取最近一次创建成功的快照的快照信息；

[0027] Jobmanager协调调度Taskmanager恢复数据同步任务的执行；

[0028] sink算子任务根据所述快照信息获取最近一次创建成功的快照的事务标识ID及快照数据；

[0029] sink算子任务根据事务ID获取事务提交状态，判断对应快照数据是否正式提交到目的数据库表中；

[0030] 在事务未正式提交的状态下，sink算子任务将最近一次创建成功的快照的快照数据提交到目的数据库表中；

[0031] source算子任务获取下一待消费的快照数据偏移继续进行数据同步。

[0032] 进一步地，所述Jobmanager和各算子任务在状态存储后端中记录快照信息，所述状态存储后端为分布式文件系统；所述分布式数据源为分布式发布订阅消息系统；所述sink算子任务将事务状态记录在分布式内存数据库中。

[0033] 基于本发明实施例的另一方面，本发明还提供一种事务性分布式数据同步系统，该系统包括Flink集群、状态存储后端、分布式内存数据库；

[0034] 所述Flink集群包括作业管理器Jobmanager、任务管理器Taskmanager，Taskmanager中至少运行有输入算子任务source和输出算子任务sink；

[0035] 所述Flink集群用于将分布式数据源中的数据同步到目的无主键关系数据库表即目的数据库表中,所述数据同步过程分为预提交阶段和正式提交阶段;

[0036] 在预提交阶段中,完成快照创建以及快照数据的预提交,输出算子sink算子任务负责将快照数据预提交到目的数据库表中并记录快照完成状态及事务预提交状态;

[0037] 在正式提交阶段中,作业管理器Jobmanager负责在快照创建成功后指令sink算子任务正式提交快照数据到目的数据库表中并记录事务提交结果;

[0038] 所述状态存储后端用于存储快照信息和缓存快照数据;

[0039] 所述分布式内存数据库用于存储事务状态。

[0040] 进一步地,在所述预提交阶段中:

[0041] 所述Jobmanager通过注入检查点分界线标记当前同步的快照数据;

[0042] 所述source算子任务在检测到检查点分界线后记录快照偏移;

[0043] 所述sink算子任务在检测到检查点分界线后,负责缓存当前快照数据、预提交当前快照数据到目的数据库表以及记录事务预提交状态;

[0044] 各算子任务在成功完成快照数据的处理后向Jobmanager发送快照完成反馈,sink算子任务在成功完成快照数据的预提交的处理后向Jobmanager发送预提交结果反馈;

[0045] 所述Jobmanager还用于接收到所有算子任务的快照完成反馈和sink算子任务的预提交结果反馈,记录快照成功状态。

[0046] 进一步地,在所述正式提交阶段中:

[0047] 所述Jobmanager还用于在根据快照完成反馈和预提交结果反馈判定创造创建成功及预提交成功后,向所有算子任务发送快照完成消息;

[0048] 所述输出算子任务sink接收到快照完成消息后正式将快照数据提交到目的数据库表中并记录事务正式提交成功与否的事务提交结果。

[0049] 基于本发明实施例的一方面,本发明还提供一种事务性分布式数据同步装置,该装置用于通过Flink集群将分布式数据源中的数据同步到目的无主键关系数据库表即目的数据库表中,该装置包括:

[0050] 预提交模块,用于完成所述数据同步过程中的预提交阶段的处理步骤,完成快照创建以及快照数据的预提交;其中,通过输出算子任务sink将快照数据预提交到目的数据库表中并记录快照完成状态及事务预提交状态;

[0051] 正式提交模块,用于完成所述数据同步过程中的正式提交阶段的处理步骤,将快照数据以事务模式正式提交的目的数据库表中;其中,通过作业管理器Jobmanager在快照创建成功后指令sink算子任务正式提交快照数据到目的数据库表中并记录事务提交状态并记录事务提交结果。

[0052] 本发明提供的该装置可以软件、硬件或软硬结合的方式实现。当以软件模块方式实现时,该软件模块的程序代码被加载到设备的存储介质中,由处理器读取存储介质中的程序代码并执行,从而实现该装置中各组成模块的功能。

[0053] 基于本发明实施例的一方面,本发明还提供一种电子设备,包括处理器、通信接口、存储介质和通信总线,其中,处理器、通信接口、存储介质通过通信总线完成相互间的通信;

[0054] 存储介质,用于存放计算机程序;

[0055] 处理器，用于执行存储介质上所存放的计算机程序时，实施如前所述的事务性分布式数据同步方法中的一个或多个步骤。

[0056] 本发明提供一种事务性分布式数据同步方法、装置、系统及存储介质，用于解决使用Flink从数据源同步数据到无主键目的数据库表的过程中无法保证数据一致性的技术问题。本发明将Flink集群进行分布式数据同步的过程分成预提交阶段和正式提交阶段，在预提交阶段创建快照并以数据库事务的预提交模式将快照数据提交到目的数据库，在快照创建成功后通过正式提交阶段将快照数据正式提交到目的数据库表。本发明技术方案中不仅记录快照的处理状态还会记录事务的提交状态，并在Flink故障的情况下，根据记录的快照和事务状态进行数据同步任务的恢复。通过本发明能够实现Flink从分布式数据源到无主键关系数据库表的数据同步的一致性。

附图说明

[0057] 为了更加清楚地说明本发明实施例或者现有技术中的技术方案，下面将对本发明实施例或者现有技术描述中所需要使用的附图作简单地介绍，显而易见地，下面描述中的附图仅仅是本发明中记载的一些实施例，对于本领域普通技术人员来讲，还可以根据本发明实施例的这些附图获得其他的附图。

[0058] 图1为本发明一实施例中基于Flink两阶段分布式事务状态机制实现数据同步的过程示意图；

[0059] 图2A为本发明一实施例中预提交阶段发起预提交及执行预提交过程的示意图；

[0060] 图2B为本发明一实施例中预提交阶段确认完成预提交阶段的示意图；

[0061] 图2C为本发明一实施例中执行第二阶段即正式提交阶段的示意图；

[0062] 图3A为本发明一实施例中第一阶段快照创建未成功情况下的异常恢复步骤示意图；

[0063] 图3B为本发明一实施例中第一阶段快照创建成功情况下的异常恢复步骤示意图；

[0064] 图4为本发明一实施例提供的用于实现事务性分布式数据同步方法的电子设备结构示意图。

具体实施方式

[0065] 在本发明实施例使用的术语仅仅是出于描述特定实施例的目的，而非限制本发明实施例。本发明实施例中所使用的单数形式的“一种”、“所述”和“该”也旨在包括多数形式，除非上下文清楚地表示其它含义。应当理解，尽管在本发明实施例可能采用术语第一、第二、第三等来描述各种信息，但这些信息不应限于这些术语。这些术语仅用于区别类似的信息、实体或步骤，而不是用于描述特定的顺序或先后次序。例如，在不脱离本发明实施例范围的情况下，第一信息也可以被称为第二信息，类似地，第二信息也可以被称为第一信息。此外，所使用的词语“如果”可以被解释成为“在……时”或“当……时”或“响应于确定”。本发明中的“和/或”仅仅是一种描述关联对象的关联关系，表示可以存在三种关系，例如，A和/或B，可以表示：单独存在A，同时存在A和B，单独存在B这三种情况，其中A，B可以是单数或者复数。并且，在本发明的描述中，除非另有说明，“多个”是指两个或多于两个。“以下至少一项(个)”或其类似表达，是指的这些项中的任意组合，包括单项(个)或复数项(个)的任

意组合。例如,a,b,或c中的至少一项(个),可以表示:a,b,c,a-b,a-c,b-c,或a-b-c,其中a,b,c可以是单个,也可以是多个。

[0066] Flink是一种分布式流处理框架,可以并行和流水线方式执行任意流数据程序,Flink的流水线运行时系统可以执行批处理和流处理程序。Flink的检查点Checkpoint快照机制是其可靠性的基石。当一个任务在运行过程中出现故障时,可以根据Checkpoint快照信息恢复到故障之前的状态,然后从该状态恢复任务的运行。在Flink中,Checkpoint快照机制采用的是分布式快照算法,通过Checkpoint快照机制,保证了Flink程序内部的恰好一次(Exactly Once)的语义,保证Flink内部数据一致性。

[0067] Flink框架中包括两个重要组件,分别是管理协调组件JobManager(作业管理器)和执行进程组件TaskManager(任务管理器),Flink架构也遵循Master-Slave主从架构设计原则,JobManager相当于整个集群的Master节点,且整个集群有且只有一个活跃的JobManager,TaskManager相当于整个集群的Slave节点。

[0068] JobManager负责整个Flink集群任务的调度以及资源的管理,TaskManager负责具体的任务执行和对应任务在每个节点上的资源申请和管理。JobManager从客户端中获取提交的应用,然后根据集群中TaskManager上任务槽TaskSlot的使用情况,为提交的应用分配相应的TaskSlot资源并指令TaskManager启动从客户端中获取的应用。

[0069] 客户端通过将编写好的Flink应用编译打包,提交到JobManager,然后JobManager会根据已注册在JobManager中的TaskManager的资源情况,将任务分配给有资源的TaskManager节点,然后启动并运行任务。

[0070] 在使用Flink实现Kafka到关系数据库的数据同步的场景中,基于Flink框架原生快照检查点Checkpoint机制同时增加“状态存储后端”用于存储Flink集群中source算子任务所处理的业务数据的快照偏移offset,快照偏移为source算子任务从kafka消费的数据起始位置,sink算子任务与数据库连接,将处理后的业务数据输出写入至数据库表中。再每次快照对应的一批业务数据处理完成时,Jobmanager会将快照偏移offset写入状态存储后端,在遇到集群或任务异常时,Flink恢复至上次快照成功时的状态,继续进行下一快照数据的处理,但是该实现过程会面临如下两个问题:

[0071] 问题一:如果本次快照未成功,但是已经写入部分数据至数据库表中,此时发生Flink集群异常,任务重启,会再次从上次快照恢复,这时候就可能重复处理已入库的数据,致使数据库表中插入重复数据。

[0072] 问题二:如果本次快照成功,但快照数据未完全写入数据库,此时发生Flink集群异常,当任务重启后,会从上次成功的快照之后继续进行下一次快照数据的处理,这种情况下,由于部分快照数据未入库,就会导致丢失一部业务数据。

[0073] 可见,基于Flink原生的checkpoint快照机制,在上述情况下,只能保障数据源端和Flink内部的数据一致性,而对于sink输出端,在目的数据库表没有主键的情况下,则无法保证数据源端和目的数据库表中的业务数据的一致性。

[0074] 本发明为解决使用Flink从数据源同步数据到无主键目的数据库表的过程中,无法保证数据一致性的技术问题,提出了一种事务性分布式数据同步方案。本发明技术方案的核心思想是:将使用Flink集群进行分布式数据同步的过程分成预提交阶段和正式提交阶段,sink算子任务在预提交阶段以数据库事务的预提交模式将快照数据提交到目的数据

库,JobManager在接收到所有算子任务的快照成功反馈后记录快照成功状态并通知sink算子任务正式提交快照数据到目的数据库表,sink算子任务在提交数据成功后记录事务提交状态。本发明技术方案中不仅记录快照的处理状态还会记录事务的提交状态,并在Flink故障的情况下,根据记录的快照和事务状态进行数据同步任务的恢复。通过本发明能够实现Flink从分布式数据源(例如分布式发布订阅消息系统,Kafka等)到无主键关系数据库(例如MySQL)表的数据同步的一致性。

[0075] 基于本发明的基本思想,以下结合附图和具体实施例来描述本发明的具体实现过程。

[0076] 图1为本发明一实施例中基于Flink两阶段分布式事务状态机制实现数据同步的过程示意图。该实施例使用Flink将实现分布式数据源到无主键关系数据库表的基于分布式事务状态机制的数据同步过程分为两个阶段,第一阶段为预提交阶段,第二阶段为正式提交阶段。Flink集群中的协调管理组件JobManager(作业管理器)作为协调者在两个阶段中协调快照的创建和事务的提交,Flink集群中的执行进程组件TaskManager(任务管理器)所管理的各算子任务作为参与者负责数据同步(数据获取、数据处理、数据输出)。在完成第一阶段的预提交阶段后由JobManager发起第二阶段的处理步骤。JobManager和TaskManager协作完成基于事务机制的分布式数据同步。

[0077] 本发明通过两阶段的事务性写入过程,既能够保证同步数据在Flink内部的一致性又能够保证Sink输出端的数据一致性,从而保证数据从数据源到无主键关系数据库表的一致性。

[0078] 以下分别对两个阶段的处理过程进行详细描述。

[0079] 图2A为本发明一实施例中预提交阶段发起预提交及执行预提交过程的示意图;图2B为本发明一实施例中预提交阶段确认完成预提交阶段的示意图。图中数据同步的数据源为Kafka,本发明不限制数据源的类型和具体形式,例如也可以是其它分布式消息订阅发布组件,也可以是其它数据采集系统等。图中数据同步的目的端为无主关键字索引(无主键)的MySQL数据库表,本发明不限制目的关系数据库的类型,例如可以是MySQL、SQLServer、Oracle等。图中JobManager和TaskManager为Flink引擎自身组件,分布式文件系统HDFS作为状态存储后端用来缓存快照偏移、快照数据等,分布式内存数据库Redis为新引入组件,用于缓存事务状态信息等。

[0080] 基于图2A和图2B执行第一阶段的处理过程包括:

[0081] 步骤200、Flink集群中作业管理器向数据输入算子Source注入检查点分界线,触发快照创建过程;

[0082] Flink集群中的作业管理器Jobmanager负责调度执行数据同步作业,客户端向Jobmanager提交数据同步作业后,Jobmanager将数据同步作业的作业任务调度给任务管理器Taskmanager的各算子(Source、Map、Sink)去运行这些算子任务。

[0083] Flink自身提供Checkpoint检查点快照功能,该快照功能通过注入检查点分界线来实现,检查点分界线可以看作一种快照标记,其主要作用是用来区分前后两个checkpoint检查点快照,一个快照对应一批能否分别被多个任务算子处理的业务数据。检查点分界线是一种特殊的数据流,由Jobmanager组件触发,Jobmanager从数据输入算子(source)任务开始注入检查点分界线,该分界线会随着整个数据处理流程和普通数据流一

样进行处理流转,因此每个算子都会检测到该检查点分界线并进行相应的处理。

[0084] 步骤201、Flink集群数据输入算子任务source在执行过程中,在检测到检查点分界线后,记录快照偏移;

[0085] 数据输入source算子任务从Kafka消费数据,在接收到Jobmanager的注入检查点分界线的指令后,对消费的数据进行快照标记,并将该检查点分界线对应的快照偏移(即当前消费的kafka数据的位置信息)记录到状态存储后端。

[0086] 本发明实施例中所使用的状态存储后端可以是分布式文件系统(例如Hadoop Distributed File System,HDFS),也可以是其它类型的分布式文件系统或数据库,本发明不做限制。

[0087] 步骤202、Flink集群中的数据输出算子任务sink在检测到检查点分界线后,开启事务,缓存快照数据及事务ID;

[0088] Flink集群中可能会有多个Sink算子任务协作完成数据同步作业,每个sink算子任务在检测到检查点分界线后即开启一个针对当前快照数据的数据库事务,对该检查点分界线标记的快照数据进行序列化,将快照数据及事务标识ID序列化缓存到状态存储后端,以备在集群故障恢复时使用。sink算子任务也可以使用其它存储位置缓存序列号的快照数据,不与快照偏移等快照状态信息一起存储。

[0089] 步骤203、sink算子任务将当前快照数据预提交到目的数据库表中并记录事务的预提交状态;

[0090] Flink集群中可能会有多个Sink算子任务协作完成数据同步作业,每个sink算子任务在检测到检查点分界线后即开启一个针对当前快照数据的事务,将检查点分界线标记的快照数据及事务标识ID序列化缓存到状态存储后端之后,将快照数据预提交到目的数据库中,在完成快照数据的预提交后,sink算子任务还需要在分布式内存数据库redis中记录快照数据的预提交状态即标记自己负责的那部分快照数据的事务状态为预提交状态(未提交模式),需要记录事务ID和预提交的成功或失败的结果。

[0091] 本发明中的事务和数据库中事务的概念一致,数据库中在开始一个事务时,在执行Commit提交指令之前,对数据库表的增/删/改操作并不会实际生效,通过Rollback回滚指令可随时将当前事务回滚到事务开始前的状态,本发明中的预提交阶段的处理过程相当于数据库事务中执行Commit提交指令之前的处理过程。

[0092] 本发明中Sink算子任务以数据库事务的预写入模式将快照数据写入到目的数据库中,虽然这部分快照数据已经实际写入数据库,但是处于未正式Commit提交到目的数据库表中的状态。只有所有sink算子任务都完成了当前批次的所有快照数据的预提交并记录了事务预提交状态后,Jobmanager才可以进一步触发sink算子任务执行正式提交快照数据到目的数据库表的Commit提交操作。

[0093] 步骤204、Flink集群中的各算子任务在完成各自负责的快照数据的预提交阶段的处理后,向管理协调组件发送快照完成反馈和预提交结果反馈;

[0094] 步骤205、管理协调组件在接收到所有算子任务反馈的快照完成反馈和预提交结果反馈后,记录所注入的检查点分界线标记的快照成功状态;

[0095] 该步骤对应图2B的示例,在各算子任务完成当前快照相关的预提交阶段的处理后,都会向管理协调组件Jobmanager发送快照完成反馈和预提交结果反馈。

[0096] 例如，source算子任务在完成向状态存储后端的快照偏移的记录后，会向Jobmanager发送快照完成反馈；sink算子任务在完成向状态存储后端的快照数据的缓存后也会向Jobmanager发送快照完成反馈，在sink算子任务完成快照数据的预提交并记录预提交状态后也会向Jobmanager发送预提交结果反馈。

[0097] 管理协调组件Jobmanager在接收到所有算子有关当前快照及预提交状态的反馈，根据反馈结果判定所有算子任务都成功完成各自的处理步骤后，将快照成功状态记录到状态存储后端，快照成功状态信息可包括检查点分界线标记、快照标识、快照时间等。只有当各算子任务成功完成快照创建以及Sink算子任务成功完成同步数据的预提交并记录事务状态后，Jobmanager才可确认第一阶段完成。若有算子任务未成功完成快照创建或快照数据的预提交，则说明当前快照的第一阶段即预提交阶段处理失败，进而导致当前快照的两阶段事务状态的数据同步失败，此时需要执行相应的故障处理步骤。

[0098] 在本发明一些实施例中，source和sink算子任务一般都会有，正常情况下，一般在source和sink之间可能会有多个Map数据处理算子用于处理数据同步。Map算子任务一般属于无状态的算子，可不记录状态信息。

[0099] 图2C为本发明一实施例中执行第二阶段即正式提交阶段的示意图，基于图2C执行第二阶段的处理过程包括：

[0100] 步骤206、作业管理器向所有算子任务发送快照完成消息；

[0101] 当作业管理器Jobmanager将快照创建成功的状态写入状态存储后端后，向所有算子(Source、Map、Sink)发送快照完成消息，指示各算子任务预提交阶段完成。

[0102] 步骤207、sink算子任务在接收到快照完成消息后，将当前快照数据正式提交到目的数据库表中并记录事务提交状态为提交完成状态；

[0103] 当sink算子任务接收到预提交阶段完成通知消息后，正式将自己负责的快照数据Commit提交的目的数据库表中，提交成功后向分布式内存数据库redis记录事务提交成功的状态信息，至此，与注入的检查点分界线对应的快照数据就成功被同步到目的关系数据库表中了，基于分布式事务状态机制的第二阶段过程就成功完成了。

[0104] 本发明采用基于事务机制的二阶段分布式数据同步的一个重要的技术效果是能够实现在Flink集群故障情况下，从Kafka到目的数据库的数据一致性，若Flink集群发生故障，包括Jobmanager在预设时间内未收到所有算子有关当前快照的正常反馈、有算子反馈创建快照失败、sink算子任务反馈预提交失败等情况。检查点分界线标记的快照数据同步失败的情况可能发生在预提交阶段，也可能发生在正式提交阶段，以下针对两种情况说明故障恢复的方法。

[0105] 异常情况一：数据同步作业过程故障中断时处于第一阶段预提交阶段，并且本次Checkpoint快照未成功

[0106] 创建快照是否成功是以Jobmanager是否接收到所有算子任务发送的表示快照创建成功的快照完成反馈消息来判断，若接收到所有算子任务的快照完成反馈则Jobmanager判定为快照创建成功，此时Jobmanager会向状态存储后端记录快照创建成功状态记录信息。有些情况下Map算子任务是无状态的，可不向Jobmanager发送快照完成反馈消息，Jobmanager主要判断是否接收到所有source和sink算子任务发送的快照完成反馈即可判定快照创建是否成功。

[0107] 图3A为本发明一实施例中第一阶段快照创建未成功情况下的异常恢复步骤示意图，若在Jobmanager记录快照创建成功的状态记录信息之前，Flink集群发生故障，则在集群重新启动后执行如下步骤：

[0108] 步骤310、Jobmanager从状态存储后端获取最近一次创建成功的快照(最近失败快照的前一个成功的快照)的快照信息；

[0109] 最近一次成功创建的快照的快照信息可包括对应的检查点分界线信息(快照标识)、快照偏移、快照创建时间等。

[0110] 步骤311、Jobmanager协调调度Taskmanager恢复数据同步任务的执行；

[0111] 步骤312、sink算子任务根据所述快照信息从状态存储后端获取最近一次创建成功的快照的事务ID及快照数据；

[0112] 步骤313、sink算子任务根据事务ID获取事务提交状态，验证事务提交状态为已提交；

[0113] 该步骤可包括sink算子任务从状态存储后端获取缓存的序列化的快照数据，通过反序列化处理恢复出事务ID及快照数据，然后根据事务ID从redis中判断最近一次创建成功的快照对应的事务提交是否成功，若成功则继续执行后续步骤，若不成功则需要将对应的快照数据提交到目的数据库表中，然后执行后续步骤。

[0114] 步骤314、source算子任务从状态存储后端获取下一次待消费的快照数据(最近失败快照数据)偏移，重新开始创建快照等数据同步过程。

[0115] 图3B为本发明一实施例中第一阶段快照创建成功情况下的异常恢复步骤示意图，若在Jobmanager记录最近一次快照创建成功后，Flink集群发生故障，此种情况下可能有部分sink算子任务提交事务成功，但第二阶段还未成功完成，则在集群重新启动后执行如下步骤：

[0116] 步骤320、Jobmanager从状态存储后端获取最近一次创建成功的快照的快照信息；

[0117] 步骤321、Jobmanager协调调度Taskmanager恢复数据同步任务的执行；

[0118] 步骤322、sink算子任务根据所述快照信息从状态存储后端获取最近一次创建成功的快照的事务ID及快照数据；

[0119] 步骤323、sink算子任务根据事务ID获取事务提交状态；

[0120] 步骤324、在事务未提交的状态下(快照数据未被正式提交的目的数据库表中)，sink算子任务将本次恢复后的未提交事务提交到目的数据库表中；

[0121] 步骤325、source算子任务从状态存储后端(通过反序列化操作)获取下一待消费的起点offset(同一快照不同数据块的偏移)继续进行数据同步。

[0122] 一个检查点分界线标识的快照对应Kafka中的一批数据，多个source可并行消费Kafka中的这一批数据，一个source算子任务可能会处理属于同一快照的这一批数据中的多个数据块，各算子任务可根据检查点分界线来识别一个快照，通过数据块标识来识别所处理的是哪个数据块。

[0123] 若Flink集群在快照数据同步过程的第二阶段产生故障，则说明快照创建已经成功，因此处理的过程与图3B的过程相似，sink算子任务根据事务ID判断事务提交是否成功，若未成功则正式提交事务，若成功则继续消费Kafka中的同一批次的快照数据执行数据同步。

[0124] 图4为本发明一实施例提供的用于实现本发明提供的事务性分布式数据同步方法的电子设备结构示意图，该设备400包括：诸如中央处理单元(CPU)的处理器410、通信总线420、通信接口440以及存储器430。其中，处理器410与存储器430可以通过通信总线420相互通信。存储器430内存储有计算机程序，当该计算机程序被处理器410执行时即可实现本发明提供的事务性分布式数据同步方法中的一个或多个步骤的功能。

[0125] 存储器是指基于某种存储介质用于存储计算机程序和/或数据的装置，它可以是易失性存储器(Volatile Memory，VM，常称为内存)，也可以是非易失性存储器(Non-Volatile Memory，NVM)。内存是指与处理器直接交换数据的内部存储器，它可以随时读写数据，而且速度很快，作为操作系统和其它运行程序的临时数据的存储介质。内存可以是同步动态随机存取内存(Synchronous Dynamic Random Access Memory，SDRAM)、动态随机存取内存(Dynamic Random Access Memory，DRAM)等。非易失性存储器是指采用持久化存储介质的存储器，具有容量大和可持久保存数据的特性，可以是存储级存储器(Storage Class Memory，SCM)、固态硬盘(Solid State Disk，SSD)、NAND闪存、磁盘等。SCM是业界对介于内存与闪存之间的新存储介质的统称，是一种同时结合持久化存储特性与内存特性的复合型储存技术，存取速度慢于DRAM快于SSD硬盘。

[0126] 处理器可以是通用处理器，包括中央处理器(Central Processing Unit，CPU)、网络处理器(Network Processor，NP)等；还可以是数字信号处理器(Digital Signal Processing，DSP)、专用集成电路(Application Specific Integrated Circuit，ASIC)、现场可编程门阵列(Field-Programmable Gate Array，FPGA)或其他可编程逻辑器件、分立门或者晶体管逻辑器件、分立硬件组件。

[0127] 应当认识到，本发明的实施例可以由计算机硬件、硬件和软件的组合、或者通过存储在非暂时性(或称为非持久性)存储器中的计算机指令来实现或实施。所述方法可以使用标准编程技术，包括配置有计算机程序的非暂时性存储介质在计算机程序中实现，其中如此配置的存储介质使得计算机以特定和预定义的方式操作。每个程序可以以高级过程或面向对象的编程语言来实现以与计算机系统通信。然而，若需要，该程序可以以汇编或机器语言实现。在任何情况下，该语言可以是编译或解释的语言。此外，为此目的该程序能够在编程的专用集成电路上运行。此外，可按任何合适的顺序来执行本发明描述的过程的操作，除非本发明另外指示或以其他方式明显地与上下文矛盾。本发明描述的过程(或变型和/或其组合)可在配置有可执行指令的一个或多个计算机系统的控制下执行，并且可作为共同地在一个或多个处理器上执行的代码(例如，可执行指令、一个或多个计算机程序或一个或多个应用)、由硬件或其组合来实现。所述计算机程序包括可由一个或多个处理器执行的多个指令。

[0128] 进一步，所述方法可以在可操作地连接至合适的任何类型的计算平台中实现，包括但不限于个人电脑、迷你计算机、主框架、工作站、网络或分布式计算环境、单独的或集成的计算机平台、或者与带电粒子工具或其它成像装置通信等等。本发明的各方面可以以存储在非暂时性存储介质或设备上的机器可读代码来实现，无论是可移动的还是集成至计算平台，如硬盘、光学读取和/或写入存储介质、RAM、ROM等，使得其可由可编程计算机读取，当存储介质或设备由计算机读取时可用于配置和操作计算机以执行在此所描述的过程。此外，机器可读代码，或其部分可以通过有线或无线网络传输。当此类媒体包括结合微处理器

或其他数据处理器实现上文所述步骤的指令或程序时，本发明所述的发明包括这些和其他不同类型的非暂时性计算机可读存储介质。当根据本发明所述的方法和技术编程时，本发明还包括计算机本身。

[0129] 以上所述仅为本发明的实施例而已，并不用于限制本发明。对于本领域技术人员来说，本发明可以有各种更改和变化。凡在本发明的精神和原理之内所作的任何修改、等同替换、改进等，均应包含在本发明的保护范围之内。

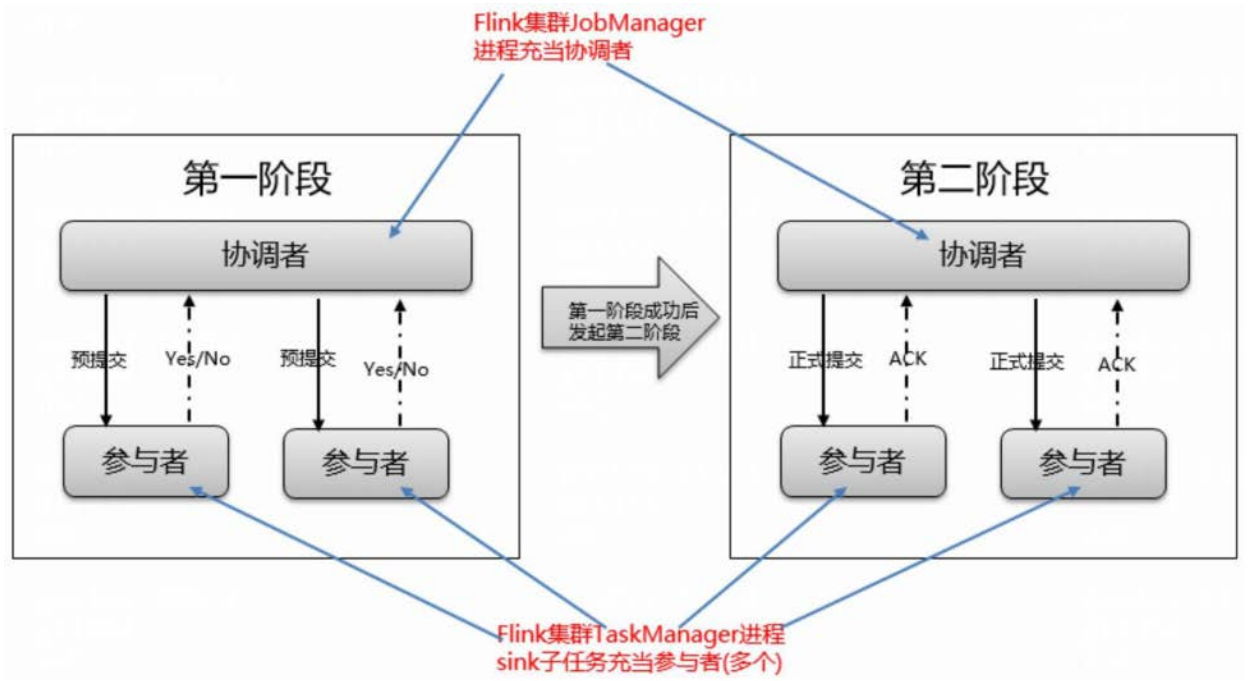

Flink集群JobManager
进程充当协调者
第一阶段
协调者
预提交
Yes/No
预提交
Yes/No
参与者
参与者
第一阶段成功后
发起第二阶段
第二阶段
协调者
正式提交
ACK
正式提交
ACK
参与者
参与者
Flink集群TaskManager进程
sink子任务充当参与者(多个)

图1

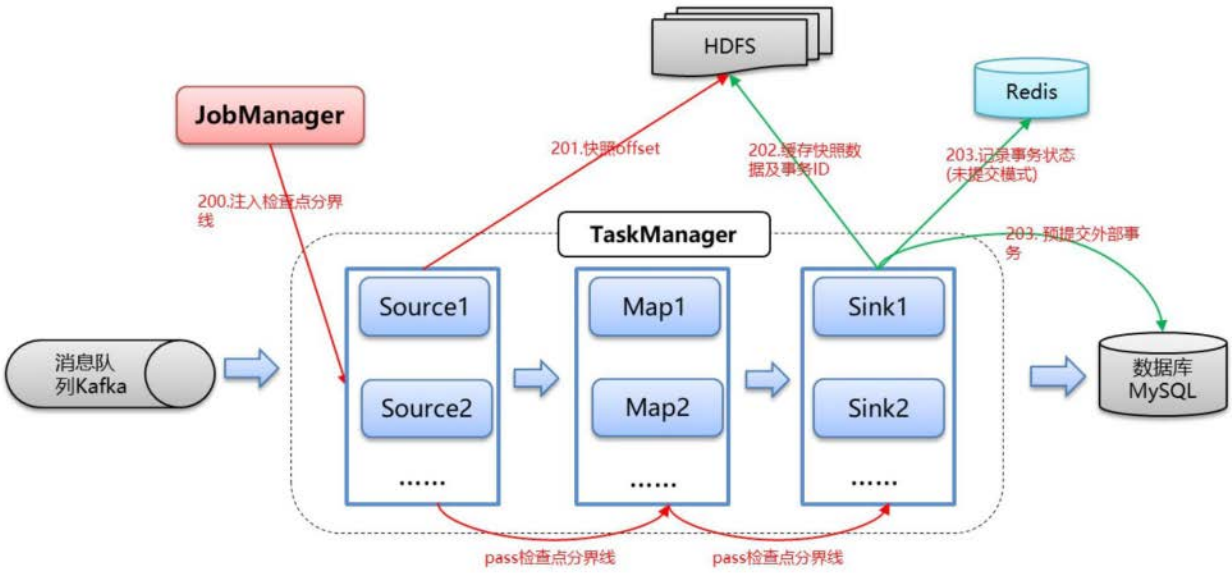

HDFS
Redis
JobManager
201.快照offset
202.缓存快照数据及事务ID
203.记录事务状态(未提交模式)
200.注入检查点分界线
TaskManager
203. 预提交外部事务
Source1
Source2
.....
Map1
Map2
.....
Sink1
Sink2
.....
消息队列Kafka
数据库MySQL
pass检查点分界线
pass检查点分界线

图2A

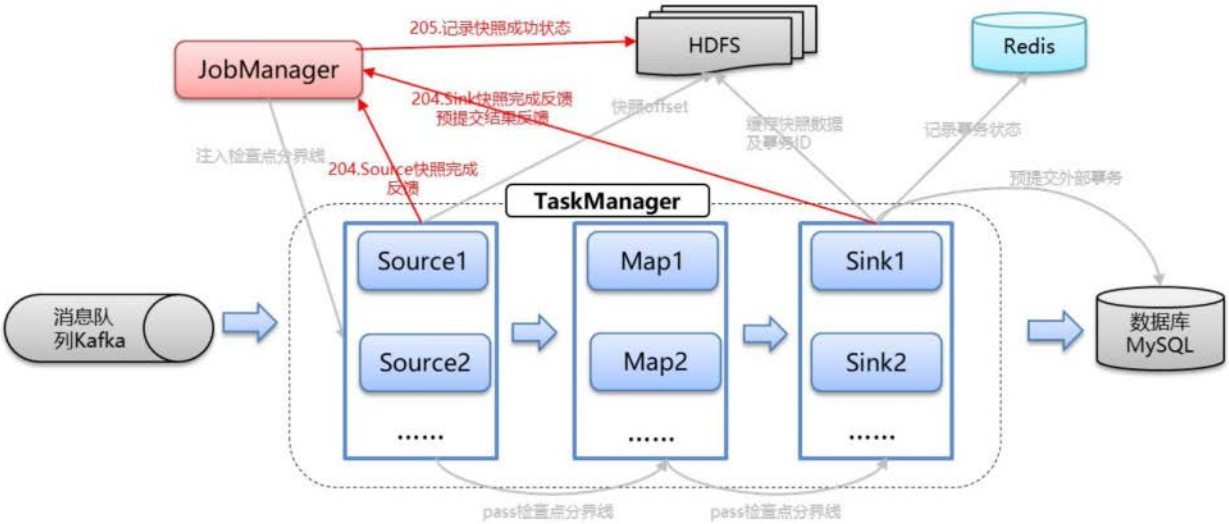

JobManager
205.记录快照成功状态
HDFS
Redis
204.Sink快照完成反馈
预提交结果反馈
快照offset
缓存快照数据
及事务ID
记录事务状态
注入检查点分界线
204.Source快照完成
反馈
预提交外部事务
TaskManager
Source1
Source2
.....
Map1
Map2
.....
Sink1
Sink2
.....
消息队
列Kafka
数据库
MySQL
pass检查点分界线
pass检查点分界线

图2B

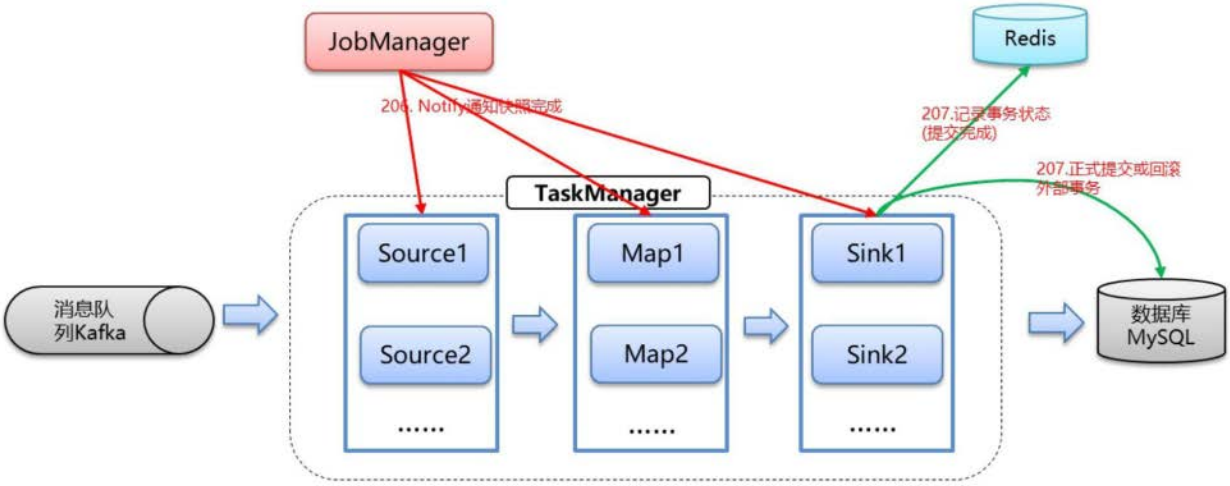

JobManager
Redis
206. Notify通知快照完成
207.记录事务状态
(提交完成)
207.正式提交或回滚
外部事务
TaskManager
Source1
Source2
.....
Map1
Map2
.....
Sink1
Sink2
.....
消息队
列Kafka
数据库
MySQL

图2C

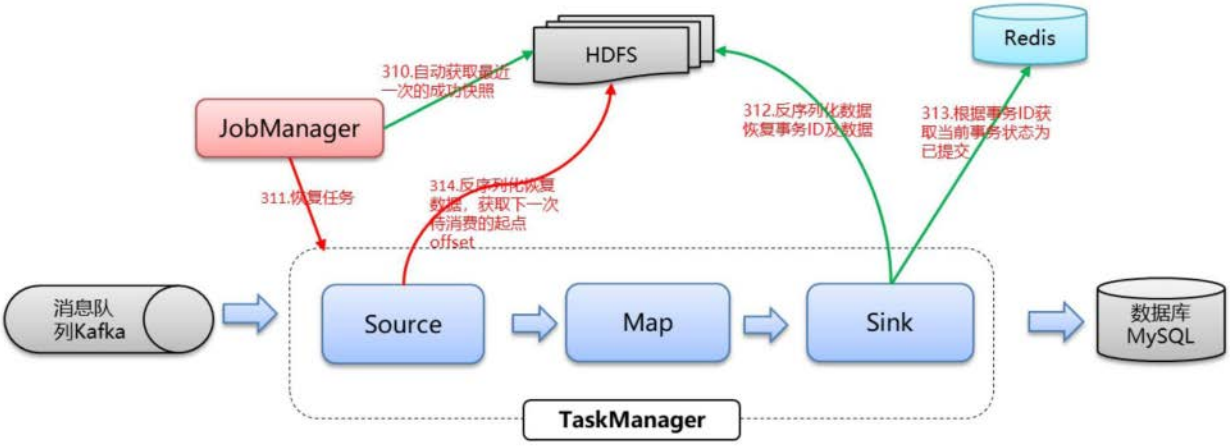

HDFS
Redis
310.自动获取最近一次的成功快照
JobManager
312.反序列化数据恢复事务ID及数据
313.根据事务ID获取当前事务状态为已提交
311.恢复任务
314.反序列化恢复数据，获取下一次待消费的起点offset
消息队列Kafka
Source
Map
Sink
数据库MySQL
TaskManager

图3A

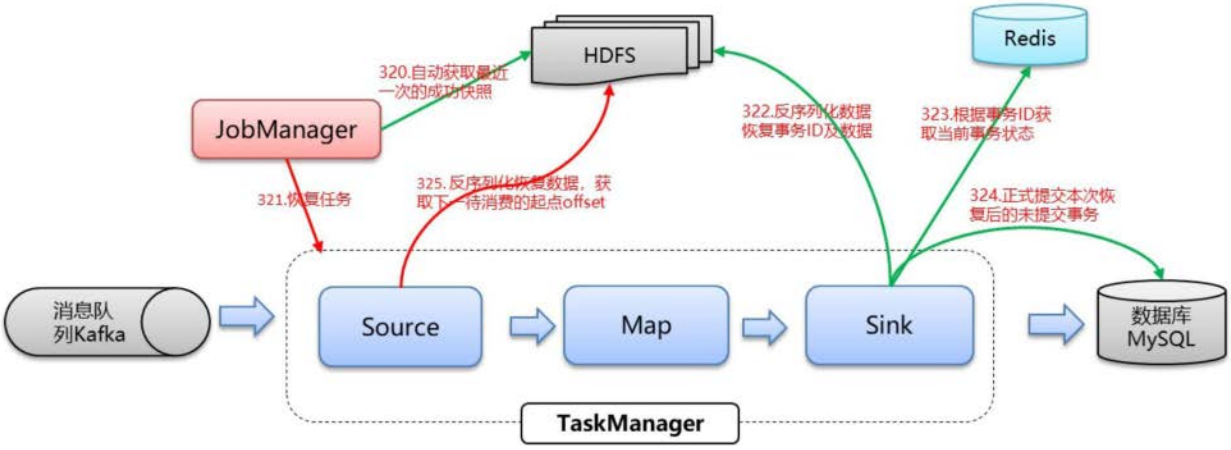

HDFS
Redis
320.自动获取最近一次的成功快照
JobManager
322.反序列化数据恢复事务ID及数据
323.根据事务ID获取当前事务状态
321.恢复任务
325.反序列化恢复数据，获取下一待消费的起点offset
324.正式提交本次恢复后的未提交事务
消息队列Kafka
Source
Map
Sink
数据库MySQL
TaskManager

图3B

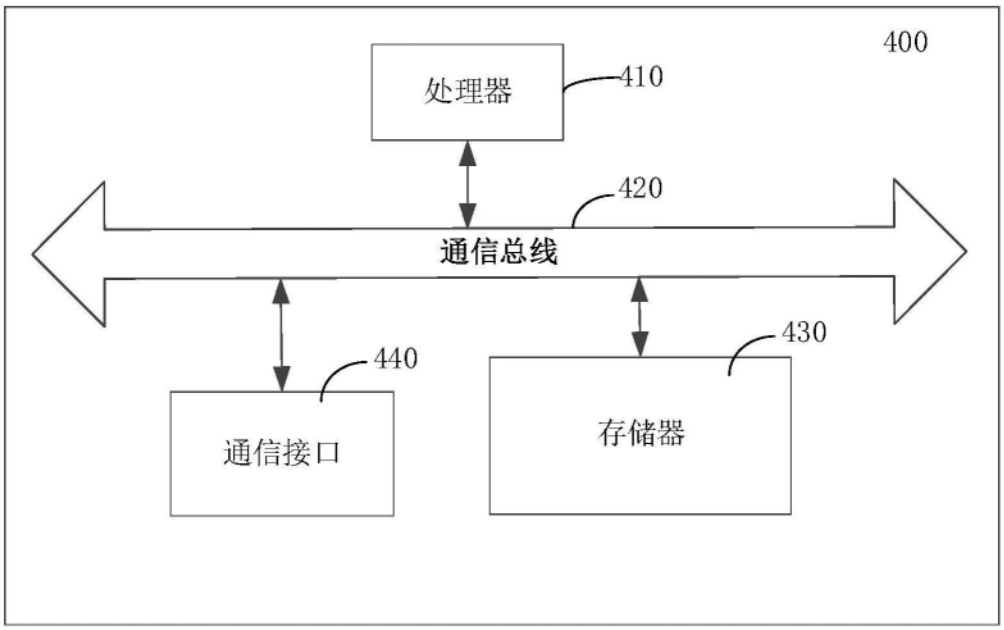


图4