

数据表的处理

技术领域

[001] 本申请涉及数据处理技术领域，尤其涉及数据表的处理。

背景技术

[002] 随着大数据时代的到来，分布式的计算方式作为一种应对大数据的计算算力不足的有效方案，逐渐变得流行起来，具体的，分布式的计算通常需要将数据以及计算任务分摊到多个分区进行处理，以提升数据处理的质量和效率。

[003] 然而，由于不同的分区中存储有不同的数据，当数据处理任务涉及多个分区中的数据时，往往需要依次对涉及的每个分区进行数据处理操作，以获得数据处理结果，这样极大地增加了数据处理的流程，同时降低了数据处理的效率。

发明内容

[004] 本申请实施例提供了一种数据表的处理方法、装置、设备及计算机程序产品，可以对存储在分布式的多个分区中的不同数据表进行关联操作，获得关联数据表，而后可以基于关联数据表进行相对应的数据处理操作，能够提高数据处理的质量和效率。

[005] 第一方面，本申请实施例提供一种数据表的处理方法，包括：获取第一数据表和第二数据表，所述第一数据表和第二数据表均存储在分布式的多个分区中，且所述第一数据表和第二数据表中包括相同的属性项；基于所述第一数据表中的属性项对所述第二数据表进行扩展，获得第二扩展表；基于所述第二数据表中的属性项对所述第一数据表进行扩展，获得第一扩展表；在所述第一扩展表中增加一数据列，获得调整后扩展表，所述数据列中包括的参数值用于对所述第一扩展表进行重新排序，以使得所述第一扩展表与所述第二扩展表中的数据对齐；对所述第二扩展表和所述调整后扩展表进行关联，获得与所述第一数据表和第二数据表相对应的关联数据表。

[006] 第二方面，本申请实施例提供一种数据表的处理装置，包括：获取模块，用于获取第一数据表和第二数据表，所述第一数据表和第二数据表存储在分布式的多个分区中，且所述第一数据表和第二数据表中包括相同的属性项；扩展模块，用于基于所述第一数据表中的属性项对所述第二数据表进行扩展，获得第二扩展表；所述扩展模块，还用于基于所述第二数据表中的属性项对所述第一数据表进行扩展，获得第一扩展表；处理模块，用于在所述第一扩展表中增加一数据列，获得调整后扩展表，所述数据列中包括的参数值用于对所述第一扩展表进行重新排序，以使得所述第一扩展表与所述第二扩展表中的数据对齐；所述处理模块，还用于对所述第二扩展表和所述调整后扩展表进行关联，获得与所述第一数据表和第二数据表相对应的关联数据表。

[007] 第三方面，本申请实施例提供一种电子设备，包括：存储器、处理器；其中，所述存储器用于存储一条或多条计算机指令，其中，所述一条或多条计算机指令被所述处理器执行时实现上述第一方面中的数据表的处理方法。

[008] 第四方面，本申请实施例提供了一种计算机存储介质，用于储存计算机程序，所述计算机程序使计算机执行时实现上述第一方面中的数据表的处理方法。

[009] 第五方面，本申请实施例提供了一种计算机程序产品，包括：计算机程序，当所述计算机程序被电子设备的处理器执行时，使所述处理器执行上述第一方面中的数据表的处理方法中的步骤。

[010] 本实施例提供的数据表的处理方法、装置、设备及计算机程序产品，通过获取第一

数据表和所述第二数据表,基于所述第一数据表中的属性项对所述第二数据表进行扩展,获得第二扩展表;基于所述第二数据表中的属性项对所述第一数据表进行扩展,获得第一扩展表;而后在所述第一扩展表中增加一数据列,获得调整后扩展表,并对所述第二扩展表和所述调整后扩展表进行关联,获得与所述第一数据表和所述第二数据表相对应的关联数据表,有效地实现了可以对存储在分布式的多个分区中的不同数据表进行关联操作,而后可以基于所获得的关联数据表进行相对应的数据处理操作,这样极大地提高了数据处理的质量和效率,并保证了该方法的实用性,有利于市场的推广与应用。

附图说明

[11] 为了更清楚地说明本申请实施例或相关技术中的技术方案,下面将对实施例或相关技术描述中所需要使用的附图作一简单地介绍,显而易见地,下面描述中的附图是本申请的一些实施例,对于本领域普通技术人员来讲,在不付出创造性劳动的前提下,还可以根据这些附图获得其他的附图。

[12] 图 1 为本申请实施例提供一种数据表的处理方法的原理示意图;

[13] 图 2 为本申请实施例提供一种数据表的处理方法的流程示意图;

[14] 图 3 为本申请实施例提供的对所述第二连接表进行扩展,获得第二扩展表的流程示意图;

[15] 图 4 为本申请实施例提供的对所述扩展次数进行前缀和计算,获得计算后参数的流程示意图;

[16] 图 5 为本申请实施例提供一种数据表的处理装置的结构示意图;

[17] 图 6 为与图 5 所示实施例提供的数据表的处理装置对应的电子设备的结构示意图。

具体实施方式

[18] 为使本申请实施例的目的、技术方案和优点更加清楚,下面将结合本申请实施例中的附图,对本申请实施例中的技术方案进行清楚、完整地描述,显然,所描述的实施例是本申请一部分实施例,而不是全部的实施例。基于本申请中的实施例,本领域普通技术人员在没有做出创造性劳动前提下所获得的所有其他实施例,都属于本申请保护的范围。

[19] 在本申请实施例使用的术语是仅仅出于描述特定实施例的目的,而非旨在限制本申请。在本申请实施例和所附权利要求书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他含义,“多种”一般包含至少两种,但是不排除包含至少一种的情况。

[20] 应当理解,本文中使用的术语“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。另外,本文中字符“/”,一般表示前后关联对象是一种“或”的关系。

[21] 取决于语境,如在此所使用的词语“如果”、“若”可以被解释成为“在……时”或“当……时”或“响应于确定”或“响应于检测”。类似地,取决于语境,短语“如果确定”或“如果检测(陈述的条件或事件)”可以被解释成为“当确定时”或“响应于确定”或“当检测(陈述的条件或事件)时”或“响应于检测(陈述的条件或事件)”。

[22] 还需要说明的是,术语“包括”、“包含”或者其任何其他变体意在涵盖非排他性的包含,从而使得包括一系列要素的商品或者系统不仅包括那些要素,而且还包括没有

明确列出的其他要素，或者是还包括为这种商品或者系统所固有的要素。在没有更多限制的情况下，由语句“包括一个……”限定的要素，并不排除在包括所述要素的商品或者系统中还存在另外的相同要素。

[23] 另外，下述各方法实施例中的步骤时序仅为一种举例，而非严格限定。

术语定义

[24] 可信执行环境 (Trusted Execution Environment, 简称 TEE)：是有安全机能的隔离的执行环境，数据可以在其中安全地进行计算而无需担心被窃取。

[25] 基于 TEE 的加密数据库：一种常见的加密数据库方案。用户将数据加密后上传至数据库，将密钥上传给 TEE。在计算时，数据库中的数据在 TEE 中被安全地解密后进行运算，运算结果在离开 TEE 前被重新加密再发送给用户。由于数据在流通过程中，除了在 TEE 内部都以加密形式存在，从而数据安全得到了一定程度上的保护。

[26] 数据重分布 (Shuffle)：在分布式下根据数据处理需求将数据按着某种方式重新分配的操作，目的是便于后续的数据处理，实现方式一般是各个分区将结果按需发送到其他分区。

[27] 数据聚合操作 (reduceByKey)：用于将具有相同键的值合并到一起。

[28] 不经意算法 (oblivious)：常用来形容算法。在单机场景下，一个 oblivious 算法是指算法对内存位置的读或写与算法的输入完全无关，只与一些公开信息（例如输入或输出数据的长度）有关。而在分布式场景下，一个 oblivious 算法是指各个分区之间交换数据的时候，发送的数据量与输入无关，而是否要求各个分区在本地计算过程中对内存位置的读或写与算法的输入完全无关则需要根据上下文语境而定。在本提案中，不对分区本地计算的算法有 oblivious 要求。

[29] 占位数据 (dummy)：在 oblivious 算法中用来作为填充位的无意义元素，目的是将数据量填充为指定大小，以防数据量的实际大小泄露隐私信息。

[30] 为了能够理解本实施例中技术方案的具体实现过程，下面先对相关技术进行简要说明。

[31] 在基于可信执行环境 TEE 的加密数据库领域中，数据通过加密进行安全保护，在进行运算时，通常需要在可信区域内被安全地解密再计算。在大数据时代，单台机器难以完成大量数据的计算操作，而新增的加解密运算无疑让计算操作的负荷进一步加重，为此，分布式的计算方式作为一种应对大数据计算算力不足的有效方案，逐渐变得流行起来。分布式的计算通常需要将数据以及计算任务分摊到多个分区进行处理，以提升数据处理的质量和效率。

[32] 然而，即使数据已被加密数据库进行加密保护，分布式下各个分区在进行数据计算时，相互通信传输的数据量仍然存在严重的隐私泄露问题，例如：在分布式开源处理系统执行数据聚合算子 (reduceByKey) 时，第一步会将具有相同键的值通过数据重分布算子 (shuffle) 集中到同一个分区，以便后续的运算。通过观察，在 shuffle 进行数据重分布的过程中，各个分区接收到的数据量可以大致估计原数据在键上的数据分布。因此，区别于非加密数据库，这些算子需要重新设计成不经意算法 (oblivious) 的，以使得运算过程中各分区之间通信发送的数据量与数据输入无关，从而实现对数据的全方位保护。

[33] 对于数据库查询的常见算子可以包括：过滤(filter)算子、分组聚合(group by aggregate)算子、投影(project)算子、排序(order by)算子等，改造不经意算法的难度都不大。

[34] 相关技术 1 实现了一种满足 oblivious 要求的分布式数据库查询系统，该分布式数据

库查询系统能够支持排序、过滤、聚合操作，然而，该查询系统并不支持一般的自然连接算子，只支持主键-外键类型的自然连接算子，缺乏通用性。

[35] 相关技术 2 实现了一种高效的分布式数据库算子计算方案，该计算方案对过滤、聚合操作进行了优化，移除了对排序操作的依赖，同时也支持了一般的自然连接的操作。然而，该方案在安全性和高效性上存在以下缺点。

[36] 在安全性上，算法依赖于公开的 α 和 β 两个数值，它们分别代表参与自然连接的两个表在对应连接条件的键上的最大重复度，若这两个数值本身也属于隐私信息，那么这个算法将不适用，否则会违背安全要求。

[37] 在高效性上，该算法会将每个分区填充一些占位数据 (dummy) 至总计 $M/p + \alpha\beta$ 条，这里 M 是自然连接算子的输出结果行数， p 是分区总数量。一个达到计算平衡的算法在每个分区的数据总量应恰好为 M/p 条，意味着该方案给出的算子为实现不经意算法的要求而填充的数据多达 $\alpha\beta$ 条。在填充前后，该算法对应有两轮完整通信，总的通信量约为 $N + 2M + 2\alpha\beta p$ 。最坏情况下， $\alpha\beta = N^2$ ，其中， N 是参与自然连接的两个表的行数，因此，该方案在很多场景下的实际上非常低效；并且，对于常见的自然连接算子，要改造成不经意算法不是一件显然的事情。若算法设计不当，则会导致通信量的大幅增长，计算效率下降，甚至于背离了分布式计算的初衷。

[38] 为了解决上述技术问题，本实施例提供了一种数据表的处理方法、装置及设备，参考附图 1 所示，本实施例提供的数据表的处理方法的执行主体可以为数据表的处理装置，需要注意的是，该数据表的处理装置可以实现为终端设备、个人电脑、平板电脑、本地服务器或者云端的服务器，此时，在数据表的处理装置实现为云端的服务器时，该数据表的处理方法可以在云端来执行，在云端可以部署有若干计算节点（云服务器），每个计算节点中都具有计算、存储等处理资源。在云端，可以组织由多个计算节点来提供某种服务，当然，一个计算节点也可以提供一种或多种服务。云端提供该服务的方式可以是对外提供服务接口，用户调用该服务接口以使用相应的服务。服务接口包括软件开发工具包 (Software Development Kit, 简称 SDK)、应用程序接口 (Application Programming Interface, 简称 API) 等形式。

[39] 该数据表的处理装置与客户端通信连接，其中，客户端用于供用户进行应用，以能够实现数据表的处理操作，上述的客户端可以是任何具有一定数据传输能力的计算设备，具体实现时，客户端可以是手机、个人电脑 PC、平板电脑、设定应用程序等等。此外，客户端的基本结构可以包括：至少一个处理器。处理器的数量取决于客户端的配置和类型。客户端也可以包括存储器，该存储器可以为易失性的，例如：随机存取存储器 (Random Access Memory, 简称 RAM)，也可以为非易失性的，例如：只读存储器 (Read-Only Memory, 简称 ROM)、闪存等，或者也可以同时包括两种类型。存储器内通常存储有操作系统 (Operating System, 简称 OS)、一个或多个应用程序，也可以存储有程序数据等。除了处理单元和存储器之外，客户端还包括一些基本配置，例如网卡芯片、IO 总线、显示组件以及一些外围设备等。可选地，一些外围设备可以包括，例如键盘、鼠标、输入笔、打印机等。其它外围设备在本领域中是众所周知的，在此不做赘述。

[40] 数据表的处理装置是指可以在网络虚拟环境中提供数据表的处理操作的设备，通常是指利用网络进行信息规划以及数据表的处理操作的装置。在物理实现上，数据表的处理装置可以是任何能够提供计算服务，响应数据表的处理请求，并可以基于数据表的处理请求进行数据表的处理操作的设备，例如：可以是集群服务器、常规服务器、云服务器、云主机、虚拟中心等。数据表的处理装置的构成主要包括处理器、硬盘、内存、系统总线等，和通用的计算机架构类似。

[41]在上述本实施例中，客户端与数据表的处理装置进行网络连接，该网络连接可以是无线或有线网络连接。若客户端可以与数据表的处理装置是通信连接，该移动网络的网络制式可以为2G(GSM)、2.5G(GPRS)、3G(WCDMA、TD-SCDMA、CDMA2000、UTMS)、4G(LTE)、4G+(LTE+)、WiMax、5G、6G等中的任意一种。

[42]在本申请实施例中，客户端，用于生成或者获取数据表的处理请求，具体的，客户端可以显示人机交互界面，获取用户在人机交互界面中输入的执行操作，基于执行操作生成或者获取数据表的处理请求，为了能够实现数据表的处理操作，可以将数据表的处理请求发送至数据表的处理装置。

[43]数据表的处理装置，用于获取客户端发送的数据表的处理请求，而后可以确定与数据表的处理请求相对应的第一数据表和第二数据表，其中，第一数据表和第二数据表分别为完整的数据表，且均存储在分布式的多个分区中，即构成第一数据表的多个数据子表均匀地分布存储在各个分区中，构成第二数据表的多个数据子表均匀的分布存储在各个分区中，并且，对于第一数据表和第二数据表而言，第一数据表和第二数据表中可以包括相同的属性项，相同的属性项可以包括一类、两类或者更多类等等。

[44]由于第一数据表和第二数据表存储在分布式的多个分区中，为了能够提高数据处理的质量和效率，并尽量降低数据通信量，在获取到与数据表的处理请求相对应的第一数据表和第二数据表之后，可以基于第一数据表中的属性项对第二数据表进行扩展操作，获得第二扩展表，相类似的，可以基于第二数据表中的属性项对第一数据表进行扩展操作，获得第一扩展表。

[45]在对第一数据表和第二数据表进行扩展操作之后，所获得的第一扩展表和第二扩展表中的数据并未对齐，为了能够对第一数据表和第二数据表进行稳定地处理操作，可以在第一扩展表中增加一数据列，从而可以获得与第二扩展表中的数据相对齐的调整后扩展表，在获取到第二扩展表和调整后扩展表之后，可以对第二扩展表和调整后扩展表进行关联操作，从而可以获得与第一数据表和第二数据表相对应的关联数据表，而后可以基于关联数据表进行相对应的数据处理操作，这样在需要基于第一数据表和第二数据表进行数据处理操作时，可以直接基于关联数据表进行数据处理操作，从而有效地提高了数据处理的质量和效率，有利于市场的推广与应用。

[46]下面结合附图，对本申请的一些实施方式作详细说明。在各实施例之间不冲突的情况下，下述的实施例及实施例中的特征可以相互组合。另外，下述各方法实施例中的步骤时序仅为一种举例，而非严格限定。

[47]图2为本申请实施例提供的一种数据表的处理方法的流程示意图；参考附图2所示，本实施例提供了一种数据表的处理方法，该方法的执行主体可以为数据表的处理装置，可以理解的是，该数据表的处理装置可以实现为软件、或者软件和硬件的组合，具体的，在数据表的处理装置实现为硬件时，其具体可以是能够实现数据表的处理操作的各种电子设备，包括但不限于平板电脑、个人电脑PC、服务器等等。当数据表的处理装置实现为软件时，其可以安装在上述所列举的电子设备中。基于上述的数据表的处理装置，本实施例中的数据表的处理方法可以包括以下步骤S201至步骤S205。

[48]步骤S201：获取第一数据表和第二数据表，第一数据表和第二数据表均存储在分布式的多个分区中，且第一数据表和第二数据表中包括相同的属性项。

[49]步骤S202：基于第一数据表中的属性项对第二数据表进行扩展，获得第二扩展表。

[50]步骤S203：基于第二数据表中的属性项对第一数据表进行扩展，获得第一扩展表。

[51]步骤S204：在第一扩展表中增加一数据列，获得调整后扩展表，数据列中包括的参

数值用于对第一扩展表进行重新排序，以使得第一扩展表与第二扩展表中的数据对齐。

[52] 步骤 S205: 对第二扩展表和调整后扩展表进行关联，获得与第一数据表和第二数据表相对应的关联数据表。

[53] 下面对上述各个步骤的具体实现原理和实现效果进行详细说明。

[54] 步骤 S201: 获取第一数据表和第二数据表，第一数据表和第二数据表均存储在分布式的多个分区中，且第一数据表和第二数据表中包括相同的属性项。

[55] 其中，在用户存在数据表的处理需求时，可以使得数据表的处理装置获取到第一数据表和第二数据表，在一些实例中，第一数据表和第二数据表可以通过人机交互操作所获得，此时，获取第一数据表和第二数据表可以包括：显示用于实现数据表的处理操作的人机交互界面；在人机交互界面中显示多个数据表；获取用户针对多个数据表输入的执行操作，基于执行操作获取第一数据表和第二数据表。举例来说，在数据表的处理装置存储有多个数据表包括：数据表 a、数据表 b、数据表 c、数据表 d、数据表 e，用户可以针对上述的数据表 b 和数据表 c 存在数据表的处理需求时，则可以针对上述的数据表 b 和数据表 c 输入选中操作，从而可以获得待处理的数据表 b 和数据表 c。

[56] 在另一些实例中，第一数据表和第二数据表不仅可以通过人机交互操作所获得，还能够通过分析处理所获得，此时，获取第一数据表和第二数据表可以包括：获取与数据表的处理装置通信连接的客户端或者预设设备；通过客户端或者预设设备主动或者被动地获取数据表的处理请求；确定与数据表的处理请求相对应的第一数据表和第二数据表，从而有效地保证了对第一数据表和第二数据表进行获取的准确可靠性。

[57] 需要注意的是，第一数据表和第二数据表可以分别是完整的数据表，上述的第一数据表和第二数据表均存储在分布式的多个分区中，其中，分区可以是分布式下的计算单元，其可以是一台机器，例如：服务器；或者，分区也可以是机器处理器的一个核心，每个分区拥有独立的计算、存储、网络资源，分区之间通过网络通信来协同完成计算任务。

[58] 另外，对于第一数据表和第二数据表而言，第一数据表和第二数据表中包括相同的属性项，举例来说，第一数据表为 $R(A,B)$ ，上述的第一数据表可以由子表 R1、子表 R2、子表 R3 中包括的数据所构成，子表 R1 可以存储在分区 a 中，子表 R2 可以存储在分区 b 中，子表 R3 可以存储在分区 c 中，即第一数据表 $R(A,B)$ 可以分布式存储在上述的多个分区中；相类似的，第二数据表为 $S(B,C)$ ，第二数据表可以由子表 S1、子表 S2、子表 S3 中包括的数据所构成，子表 S1 可以存储在分区 a 中，子表 S2 可以存储在分区 b 中，子表 S3 可以存储在分区 c 中。并且，上述的第一数据表 $R(A,B)$ 和第二数据表 $S(B,C)$ 可以包括相同的属性项 B，可以理解的是，第一数据表与第二数据表之间所存在的相同的属性项可以不仅为 1 类，本领域技术人员可以根据需求对第一数据表与第二数据表之间所存在的相同属性项的数量进行灵活调整操作。

[59] 当第一数据表与第二数据表之间具有相同的属性项时，则可以对第一数据表和第二数据表进行关联处理操作。当第一数据表和第二数据表之间不存在相同的属性项时，则无法实现对第一数据表和第二数据表进行关联处理操作。

[60] 步骤 S202: 基于第一数据表中的属性项对第二数据表进行扩展，获得第二扩展表。

[61] 在获取到第一数据表和第二数据表之后，为了能够实现数据表的关联操作以及保证第一数据表和第二数据表中的数据安全性，可以基于第一数据表中的属性项来对第二数据表进行扩展操作，其中，第一数据表中的属性项为第一数据表和第二数据表中包括的共有属性项，从而可以获得第二扩展表。

[62] 在一些实例中，扩展操作可以通过预设的机器学习模型、神经网络模型或者预设算法所实现，此时，基于第一数据表中的属性项对第二数据表进行扩展，获得第二扩展表可以包括：获取预先训练好的机器学习模型或者神经网络模型；将第一数据表中的属性项和第二数据表输入至机器学习模型或者神经网络模型中，获得机器学习模型或者神经网络模型所输出的第二扩展表。

[63] 在另一些实例中，扩展操作不仅可以通过机器学习模型或者神经网络模型来实现，还能够基于第一数据表中的属性项的重复度来实现扩展操作，此时，基于第一数据表中的属性项对第二数据表进行扩展，获得第二扩展表可以包括：基于第一数据表中的属性项对第一数据表进行分组聚合处理，获得第一聚合表，第一聚合表中包括位于属性项下的各个属性值的重复度；对第一聚合表和第二数据表进行自然连接，获得第二连接表；对第二连接表进行扩展，获得第二扩展表。

[64] 具体的，在获取到第一数据表和第二数据表之后，可以基于第一数据表中的属性项对第一数据表进行分组聚合处理，获得第一聚合表，该第一聚合表中可以包括位于属性项下的各个属性值的重复度。

[65] 举例来说，第一数据表 $R(A,B)$ 可以为 $\{(a1,b1),(a1,b2),(a2,b2),(a1,b3),(a2,b4)\}$ ，第二数据表 $S(B,C)$ 可以为 $\{(b1,c1),(b1,c2),(b2,c2),(b2,c3),(b3,c4)\}$ ，由上可知，第一数据表和第二数据表之间具有相同的属性项 B ，即第一数据表中的属性项 B 为 b 列所对应的数值，而后通过对第一数据表中的属性项的重复度进行统计操作，可以获得上述属性项中各个属性值的重复度，即 $b1$ 的重复度为 1， $b2$ 的重复度为 2， $b3$ 的重复度为 1， $b4$ 的重复度为 1，基于上述的第一数据表中的属性值的重复度来对第一数据表进行分组聚合处理，获得第一聚合表 $R(B,D)$ ，该第一聚合表 $R(B,D)$ 可以为 $\{(b1,1),(b2,2),(b3,1),(b4,1)\}$ 。

[66] 在获取到第一聚合表和第二数据表之后，可以对第一聚合表和第二数据表进行自然连接，获得第二连接表；在一些实例中，自然连接操作可以通过预设的机器学习模型或者神经网络模型来实现，举例来说，在第一聚合表 $R(B,D)$ 为 $\{(b1,1),(b2,2),(b3,1),(b4,1)\}$ ，第二数据表 $S(B,C)$ 为 $\{(b1,c1),(b1,c2),(b2,c2),(b2,c3),(b3,c4)\}$ 时，通过对第一聚合表和第二数据表进行自然连接操作，从而可以获得第二连接表 $S2(B,C,D)$ ，该第二连接表 $S2(B,C,D)$ 可以为 $\{(b1,c1,1),(b1,c2,1),(b2,c2,2),(b2,c3,2),(b3,c4,1)\}$ 。在获取到第二连接表之后，可以对第二连接表进行扩展操作，这样有效地实现了对第二数据表进行扩展操作，从而可以稳定地获取到第二扩展表。

[67] 步骤 S203：基于第二数据表中的属性项对第一数据表进行扩展，获得第一扩展表。

[68] 在获取到第一数据表和第二数据表之后，为了能够实现数据表的关联操作以及保证第一数据表和第二数据表中的数据安全性，可以基于第二数据表中的属性项来对第一数据表进行扩展操作，其中，第二数据表中的属性项为第一数据表和第二数据表中包括的共有属性项，从而可以获得第一扩展表。

[69] 在一些实例中，基于第二数据表中的属性项对第一数据表进行扩展，获得第一扩展表可以包括：基于第二数据表中的属性项对第二数据表进行分组聚合处理，获得第二聚合表，第二聚合表中包括位于属性项下的各个属性值的重复度；对第二聚合表和第一数据表进行自然连接，获得第一连接表；对第一连接表进行扩展，获得第一扩展表。

[70] 需要注意的是，本实施例中对第一数据表进行扩展操作的具体实现方式、实现效果和实现原理与上述步骤 S202 中对第二数据表进行扩展操作的具体实现方式、实现效果和实现原理相类似，具体可参考上述陈述内容，在此不再赘述。

[71] 步骤 S204：在第一扩展表中增加一数据列，获得调整后扩展表，数据列中包括的参数值用于对第一扩展表进行重新排序，以使得第一扩展表与第二扩展表中的数据对齐。

[72]对于第一数据表和第二数据表而言，经过对第一数据表和第二数据表进行扩展操作之后，所获得的第一扩展表中的数据与第二扩展表中的数据被打乱，此时，第一扩展表中的数据与第二扩展表中的数据往往并未对齐。因此，为了能够实现数据表的处理操作，在获取到第一扩展表之后，可以在第一扩展表中增加一数据列，获得调整后扩展表，数据列中包括的参数值用于对第一扩展表进行重新排序操作，以使得第一扩展表与第二扩展表中的数据相对齐。

[73]在一些实例中，在第一扩展表中增加一数据列，获得调整后扩展表可以包括：在第一扩展表增加初始值为1的数据列，获得中间扩展表；对中间扩展表中的相邻两组数据的两个属性值进行比较；在相邻两组数据所对应的两个属性值均相同时，则对相邻两组数据中的数据列进行求和处理，获得求和数据列，并将两个属性值与求和数据列确定用于构成调整后扩展表的数据；在相邻两组数据所对应的两个属性值不同时，则将相邻两组数据中的后一个数据确定为用于构成调整后扩展表的数据。

[74]举例来说，在第一扩展表为 $\begin{Bmatrix} a1 & b1 \\ a2 & b2 \end{Bmatrix}$ 时，通过对第一扩展表增加初始值为1的数据列时，则可以获取到中间扩展表，该中间扩展表可以为 $\begin{Bmatrix} a1 & b1 & 1 \\ a2 & b2 & 1 \end{Bmatrix}$ ，而后可以对中间扩展表中的相邻两组数据的两个属性值进行比较，即将数据 $\{a1, b1, 1\}$ 与数据 $\{a2, b2, 1\}$ 中的属性值“a”和“b”进行比较，在 $a1 = a2$ 、且 $b1 = b2$ 时，则可以对相邻两组数据中的数据列进行求和处理，从而可以获得求和数据列，该求和数据列即为 $\{a1, b1, 2\}$ ，而后则可以将两个属性值与求和数据列确定为用于构成调整后扩展表的数据，这样则可以保证对调整后扩展表进行获取的准确可靠性。

[75]相对应的，在相邻两组数据所对应的两个属性值不同时，即 $a1 \neq a2$ 或者 $b1 \neq b2$ 时，则可以将相邻两组数据中的后一个数据确定为用于构成调整后扩展表的数据，即此时用于构成调整后扩展表的数据为 $\{a2, b2, 1\}$ ，这样有效地保证了对调整后扩展表进行获取的准确可靠性。

[76]需要注意的是，为了能够使得第一扩展表和第二扩展表中的数据相对齐，不仅可以对第一扩展表中增加数据列，还可以对第二扩展表进行数据列的增加操作，从而可以获得与第二扩展表相对应的调整后扩展表，此时，步骤 S204 可以变为“在第二扩展表中增加一数据列，获得调整后扩展表，数据列中包括的参数值用于对第二扩展表进行重新排序，以使得第一扩展表和第二扩展表中的数据对齐”。

[77]步骤 S205：对第二扩展表和调整后扩展表进行关联，获得与第一数据表和第二数据表相对应的关联数据表。

[78]由于调整后扩展表中的数据与第二数据表中的数据相对齐，为了能够准确地实现对第一数据表和第二数据表进行处理操作，在获取到调整后扩展表和第二扩展表之后，可以对第二扩展表和调整后扩展表进行关联操作，从而可以获得与第一数据表和第二数据表相对应的关联数据表。

[79]在一些实例中，关联操作可以通过预先训练好的机器学习模型或者神经网络模型所实现，此时，对第二扩展表和调整后扩展表进行关联，获得与第一数据表和第二数据表相对应的关联数据表可以包括：获取预先训练好的机器学习模型或者神经网络模型；将第二扩展表和调整后扩展表输入至机器学习模型或者神经网络模型中，获得机器学习模型或者神经网络模型所输出的与第一数据表和第二数据表相对应的关联数据表，该关联数据表即为对第一数据表和第二数据表进行自然连接操作之后所获得的结果，这样有效地保证了对关联数据表进行确定的准确可靠性。对于所获得的关联数据表而言，关联数据表可以存储在分布式的多个分区中，这样可以保证关联数据表存储的质量和效果。

[80]在另一些实例中，不仅可以通过预先训练好的机器学习模型或者神经网络模型来实现数据表的关联操作，还能够利用预设规则来实现数据表的关联操作，此时，对第一扩展表和调整后扩展表进行关联，获得与第一数据表和第二数据表相对应的关联数据表，包括：获取用于对第一扩展表和调整后扩展表进行排序处理的预设规则；利用预设规则分别对第一扩展表和调整后扩展表进行排序，获得第一排序表和第二排序表；对第一排序表和第二排序表进行关联，获得与第一数据表和第二数据表相对应的关联数据表。

[81]在获取到第一扩展表之后，可以先获取用于对第一扩展表进行排序处理的预设规则，该预设规则可以为存储在预设区域或者预设设备中的字典序或者其他序列；相类似的，在获取到调整后扩展表之后，可以先获取用于对调整后扩展表进行排序处理的预设规则，该预设规则可以为预先存储在预设区域或者预设设备中的字典序或者其他序列；需要注意的是，用于对第一扩展表进行排序处理的预设规则与用于对调整后扩展表进行排序处理的预设规则可以同步获取或者异步获取。

[82]在获取到用于对第一扩展表和调整后扩展表进行排序处理的预设规则之后，可以利用预设规则分别对第一扩展表和调整后扩展表进行排序，从而可以获得第一排序表和第二排序表，在获取到第一排序表和第二排序表之后，可以对第一排序表和第二排序表进行关联操作，从而可以获得与第一数据表和第二数据表相对应的关联数据表。举例来说，在第一数据表中的第*i*数据为(*b,c*)，第二数据表中的第*i*数据为(*a,b1,f*)，其中，*b=b1*，通过对第一数据表和第二数据表进行关联操作时，则可以获取关联数据表，该关联数据表中的第*i*项数据为(*a,b,c*)，从而有效地实现了数据表的关联操作。

[83]在又一些实例中，在获得与第一数据表和第二数据表相对应的关联数据表之后，为了提高该方法的实用性，可以基于所获得的关联数据表实现数据查询操作，此时，本实施例中的方法还可以包括：获取数据查询请求；确定与数据查询请求相对应的关联数据表；基于关联数据表，获得数据查询结果。

[84]具体的，在用户存在数据查询需求时，可以使得数据表的处理装置获取到数据查询请求，在一些实例中，可以通过人机交互操作获得数据查询请求，或者，通过预设设备主动或者被动地获取到数据查询请求。在获取到数据查询请求之后，可以对数据查询请求进行分析处理，以确定与数据查询请求相对应的关联数据表，在获取到关联数据表之后，可以基于关联数据表进行相对应的数据查询操作，从而可以获得数据查询结果，这样有效地实现了基于关联数据表进行数据查询操作，进一步提高了该方法的实用性。

[85]本实施例提供的数据表的处理方法，通过获取第一数据表和第二数据表，基于所述第一数据表中的属性项对所述第二数据表进行扩展，获得第二扩展表；基于所述第二数据表中的属性项对所述第一数据表进行扩展，获得第一扩展表；而后在所述第一扩展表中增加一数据列，获得调整后扩展表，并对所述第二扩展表和所述调整后扩展表进行关联，获得与所述第一数据表和第二数据表相对应的关联数据表，有效地实现了可以对存储在分布式的多个分区中的不同数据表进行关联操作，而后可以基于所获得的关联数据表进行相对应的数据处理操作，这样极大地提高了数据处理的质量和效率，并保证了该方法的实用性，有利于市场的推广与应用。

[86]图3为本申请实施例提供的对第二连接表进行扩展，获得第二扩展表的流程示意图；在上述实施例的基础上，参考附图3所示，本实施例对于对第二连接表进行扩展的实现方式不做限定，在一些实例中，可以利用预先训练好的机器学习模型或者神经网络模型对第二连接表进行扩展操作，或者，可以基于每个分区所对应的数据输出量来进行第二连接表的扩展操作，此时，本实施例中的对第二连接表进行扩展，获得第二扩展表可以包括步骤S301至步骤S305。

[87]步骤S301：获取每个分区所对应的数据输出量。

[88] 由于对第二连接表的扩展操作与每个分区所对应的数据输出量相关，因此，为了能够准确地实现对第二连接表进行扩展操作，可以获取每个分区所对应的数据输出量，在一些实例中，每个分区所对应的数据输出量可以是用户预先配置的、存储在预设区域或者预设设备中的数据，此时，通过访问预设区域或者预设设备可以获取每个分区所对应的数据输出量。

[89] 或者，在另一些实例中，每个分区所对应的数据输出量不仅可以是用户预先配置好的，还可以基于分区的数量和输出数据总量来确定，此时，获取每个分区所对应的数据输出量可以包括：获取输出数据总量 M 以及分区的数量 p ；基于输出数据总量 M 以及分区的数量 p ，确定每个分区所对应的数据输出量 $m=M/p$ ，举例来说，在输出数据总量 M 为 18 时，分区的数量 p 为 3 时，则可以确定每个分区所对应的数据输出量为 $18/3=6$ ，此时，多个分区是采用均衡算法对数据进行相对应的数据处理操作。

[90] 步骤 S302：将各个属性值的重复度确定为用于对第二连接表中各个数据进行扩展操作的扩展次数。

[91] 通过对第一聚合表和第二数据表进行自然连接操作之后，可以获得第二连接表，该第二连接表中可以包括各个属性值的重复度，而后可以将各个属性值的重复度确定为用于对第二连接表中各个数据进行扩展操作的扩展次数。举例来说，在第二连接表 $S2(B,C,D)$ 为 $\{(b1,c1,1),(b1,c2,1),(b2,c2,2),(b2,c3,2),(b3,c4,1)\}$ 时，则可以将 $(b1,c1,1)$ 中的“1”确定为用于对数据“(b1,c1)”进行扩展操作的扩展次数，即需要对数据“(b1,c1)”扩展 1 次，相类似的，将 $(b1,c2,1)$ 中的“1”确定为对数据“(b1,c2)”进行扩展操作的扩展次数，将 $(b2,c2,2)$ 中的“2”确定为对数据“(b2,c2)”进行扩展操作的扩展次数，将 $(b2,c3,2)$ 中的“2”确定为对数据“(b2,c3)”进行扩展操作的扩展次数，将 $(b3,c4,1)$ 中的“1”确定为对数据“(b3,c4)”进行扩展操作的扩展次数，这样有效地保证了对用于对第二连接表中各个数据进行扩展操作的扩展次数进行确定的准确可靠性。

[92] 步骤 S303：基于扩展次数和数据输出量，确定各个数据所对应的目标分区。

[93] 由于数据表中的各个数据位于不同的分布式分区中，在进行扩展操作时，则需要将各个数据发送至相对应的目标分区，因此，为了能够实现对第二连接表进行扩展操作，在获取到扩展次数和数据输出量之后，可以对扩展次数和数据输出量进行分析处理，以确定各个数据所对应的目标分区。在一些实例中，目标分区可以通过预先训练好的机器学习模型或者神经网络模型对扩展次数和数据输出量进行分析处理所确定，此时，基于扩展次数和数据输出量，确定各个数据所对应的目标分区可以包括：获取预先训练好的机器学习模型或者神经网络模型；将扩展次数、数据输出量以及第二连接表输入至机器学习模型或者神经网络模型中，获得机器学习模型或者神经网络模型所输出的各个数据所对应的目标分区。

[94] 在另一些实例中，不仅可以通过预先训练好的机器学习模型或者神经网络模型来确定各个数据所对应的目标分区，还能够利用前缀和算法对扩展次数和数据输出量进行分析处理来确定各个数据所对应的目标分区，此时，基于扩展次数和数据输出量，确定用于对第二连接表中各个数据所对应的目标分区可以包括：对扩展次数进行前缀和计算，获得计算后参数；将 0 值与计算后参数，确定为处理后参数；获取处理后参数除以数据输出量的商值；基于商值与 1 的和值，确定数据所对应的目标分区。

[95] 举例来说，在扩展次数为 $(1,3,1,1,5,2,1,1,3)$ 时，通过对扩展次数进行前缀和计算之后，获得的计算后参数可以为 $(1,4,5,6,11,13,14,15)$ ，而后可以将 0 值与计算后参数确定为处理后参数，该处理后参数即为 $(0,1,4,5,6,11,13,14,15)$ ，在分区的数量为 3 个，且每个分区的输出数据量为 6 时，则可以获取上述各个处理后参数除以数据输出量的商值，即 $0/6$ 的商值为 0， $1/6$ 的商值为 0， $4/6$ 的商值为 0， $5/6$ 的商值为 0， $6/6$ 的商值为

1, 11/6 的商值为 1, 13/6 的商值为 2, 14/6 的商值为 2, 15/6 的商值为 2。

[96] 在获取到上述各个处理后参数所对应的商值之后, 可以获取商值与 1 的和值, 从而可以获得与多个计算后参数相对应的和值序列 (1,1,1,1,2,2,3,3,3), 由上可以确定上述和值“1”所对应的数据所对应的目标分区为预设标识为“1”的分区, 上述和值“2”所对应的数据所对应的目标分区为预设标识为“2”的分区, 上述和值“3”所对应的数据所对应的目标分区为预设标识为“3”的分区, 从而有效地保证了对数据所对应的目标分区进行确定的准确可靠性。

[97] 需要注意的是, 数据所对应的目标分区不仅可以通过商值与 1 的和值来确定, 还可以先获取处理后参数与 1 的差值, 之后获取差值与数据输出量的比值, 并对比值进行向上取整操作, 最后可以基于取整操作后的参数来确定数据所对应的目标分区, 这样同样能够保证对目标分区进行确定的准确可靠性。

[98] 步骤 S304: 基于数据输出量对各个数据进行扩展, 获得扩展数据。

[99] 为了能够实现数据扩展操作, 在获取到数据输出量之后, 可以基于数据输出量对各个数据进行扩展操作, 以获得扩展数据, 在一些实例中, 扩展操作可以通过预先训练好的机器学习模型或者神经网络模型来实现, 此时, 基于数据输出量对各个数据进行扩展, 获得扩展数据可以包括: 获取预先训练好的机器学习模型或者神经网络模型; 将数据输出量和各个数据输入至机器学习模型或者神经网络模型中, 获得机器学习模型或者神经网络模型输出的扩展数据。

[100] 在另一些实例中, 不仅可以通过预先训练好的机器学习模型或者神经网络模型来实现数据扩展操作, 还可以基于数据输出总位数来实现数据扩展操作, 此时, 基于数据输出量对各个数据进行扩展, 获得扩展数据可以包括: 基于数据输出量, 确定数据输出总位数; 基于数据输出总位数对各个数据进行扩展, 获得扩展数据。

[101] 其中, 由于数据扩展操作与数据输出总位数相关, 因此, 为了能够准确地实现数据扩展操作, 在获取到数据输出量之后, 可以基于数据输出量来确定数据输出总位数, 在一些实例中, 基于数据输出量, 确定数据输出总位数可以包括: 获取预先配置有数据输出量与数据输出总位数之间的映射关系, 基于数据输出量和映射关系来确定数据输出总位数。

[102] 在另一些实例中, 不仅可以通过预设的映射关系来确定数据输出总位数, 还可以通过预设公式来确定数据输出总位数, 此时, 基于数据输出量, 确定数据输出总位数可以包括: 对数据输出量进行平方根处理操作, 获得处理后数值; 确定数据输出量与处理后数值之间的和值, 并对和值进行取整操作, 获得数据输出总位数。举例来说, 在数据输出量为 6 时, 则可以通过 $\lceil 6 + \sqrt{6} \rceil$ 的计算, 可以获得数据输出总位数=9, 这样有效地保证了对数据输出总位数进行确定的准确可靠性。

[103] 在获取到输出数据总位数之后, 可以基于数据输出总位数对各个数据进行扩展, 获得扩展数据, 在一些实例中, 不仅可以基于预先训练好的机器学习模型或者神经网络模型对各个数据进行扩展操作, 还可以结合预设填充位来实现数据扩展操作, 此时, 基于数据输出总位数对各个数据进行扩展, 获得扩展数据可以包括: 获取需要发送同一目标分区的所有数据; 检测所有数据的数据位总和是否满足数据输出总位数; 若不满足, 则基于所有数据的数据位总和以及预设填充位进行数据扩展操作, 获得扩展数据, 扩展数据的数据位数满足数据输出总位数。

[104] 举例来说, 在分区的数量为 3 个时, 上述各个数据所对应的数据分别为: 第一分区对应有{(a,0)、(c,4)、(f,11)、(h,14)}, 第二分区对应有{(b,1)、(e,6)、(g,13)}, 第三分区对应有{(d,5)、(i,15)}, 上述的第一分区中的数据 a 所对应的处理后参数为 0, 通过处理

后参数和数据输出量即可确定第一分区为数据 a 所对应的目标分区，相类似的，经过分析处理即可确定：数据 c 所对应的目标分区为第一分区，数据 f 所对应的目标分区为第二分区，数据 h 所对应的目标分区为第三分区，数据 b 所对应的目标分区为第一分区，数据 e 所对应的目标分区为第二分区，数据 g 所对应的目标分区为第三分区，数据 d 所对应的目标分区为第一分区，数据 i 所对应的目标分区为第三分区。

[105] 经过统计之后，第一分区中的数据 a 和数据 c 的目标分区均为第一分区，此时，由于数据 a 和数据 c 的数据位数（2 位）并不能满足数据输出总位数（9 位），因此则可以基于所有数据的数据位数总和以及预设填充位进行数据扩展操作，获得扩展数据，即将数据（a,c）扩展为数据（a,c,x,x,x,x,x,x,x），上述的“x”为预设占位数据（或称为预设填充位），这样有效地实现了数据扩展操作。

[106] 相类似的，在第一分区中的数据 f 的目标分区为第二分区，此时，由于数据 f 并不能满足数据输出总位数（9 位），因此，则可以基于所有数据的数据位数总和以及预设填充位进行数据扩展操作，获得扩展数据，即将数据（f）扩展为数据（f,x,x,x,x,x,x,x,x），上述的“x”为预设填充位。第一分区中的数据 h 的目标分区为第三分区，而后可以将数据（h）扩展为数据（h,x,x,x,x,x,x,x,x），从而有效地实现了数据扩展操作。

[107] 在第二分区中的数据 b 所对应的目标分区为第一分区，而后可以将数据（b）扩展为数据（b,x,x,x,x,x,x,x,x），第二分区中的数据 e 所对应的目标分区为第二分区，而后可以将数据（e）扩展为数据（e,x,x,x,x,x,x,x,x），第二分区中的数据 g 所对应的目标分区为第三分区，而后可以将数据（g）扩展为（g,x,x,x,x,x,x,x,x）；在第三分区中的数据 d 所对应的目标分区为第一分区，而后可以将数据（d）扩展为（d,x,x,x,x,x,x,x,x）；在第三分区中的数据 i 所对应的目标分区为第三分区，而后可以将数据（e）扩展为（e,x,x,x,x,x,x,x,x），这样有效地实现了对各个数据进行扩展操作，从而可以稳定地获取到扩展数据。

[108] 步骤 S305：将扩展数据发送至目标分区，并对扩展数据进行改写，获得用于构成第二扩展表的扩展后数据。

[109] 在获取到扩展数据，并确定扩展数据所对应的目标分区之后，可以将扩展数据发送至目标分区，由于目标分区中所接收到的数据中包括很多的预设填充位，为了保证数据扩展的质量和效果，在将扩展数据发送至目标分区之后，可以对扩展数据进行改写操作，从而可以获得构成第二扩展表的扩展后数据。在一些实例中，可以利用预先训练好的机器学习模型或者神经网络模型对扩展数据进行改写操作，从而可以稳定地获取到第二扩展表。

[110] 或者，在另一些实例中，不仅可以通过预先训练好的机器学习模型或者神经网络模型来实现对扩展数据的改写操作，还可以通过删除多余的预设填充位来实现改写操作，此时，对扩展数据进行改写，获得用于构成第二扩展表的扩展后数据可以包括：获取处理后参数与数据输出量之间进行除法运算时所对应的余数；基于余数确定数据存储在目标分区中的位置；基于位置和数据输出量删除多余的预设填充位，获得排序后数据；将排序后数据中包括的预设填充位改写为前一相邻数据，获得用于构成第二扩展表的目标数据。

[111] 举例来说，在处理参数为(0,1,4,5,6,11,13,14,15)时，上述的处理后参数所对应的数据包括：第一分区对应为{(a,0)、(c,4)、(f,11)、(h,14)}，第二分区对应为{(b,1)、(e,6)、(g,13)}，第三分区对应为{(d,5)、(i,15)}，输出数据量为 6 时，则可以获取处理后参数与数据输出量之间进行除法运算时所对应的余数，具体的，上述数据 a、b、c、d、e、f、g、h 所对应处理后参数的余数为：0、1、4、5、1、5、1、2、3，上述的余数用于表示数据存储在目标分区中的位置，而数据所对应的目标分区可以通过处理后参数与数据输

出量之间进行除法运算时所对应的商值来确定。

[112] 经过分析处理可知，上述数据 a 位于目标分区（即第一分区）中的第 0 位，数据 b 位于目标分区（即第一分区）中的第 1 位，数据 c 位于目标分区（即第一分区）中的第 4 位，数据 d 位于目标分区（即第一分区）中的第 5 位，而后可以基于上述的位置和数据输出量来删除多余的预设填充位，从而可以获得位于目标分区（第一分区）中的排序后数据为：a,b,x,x,c,d，上述的 x 为剩余的预设填充位。相类似的，可以获得位于目标分区（第二分区）中的排序后数据为：e,x,x,x,x,f，上述的 x 为剩余的预设填充位；可以获得位于目标分区（第三分区）中的排序后数据为：x,g,h,i,x,x，上述的 x 为剩余的预设填充位。

[113] 在获取到排序后数据之后，可以将排序后数据中包括的预设填充位改写位前一相邻数据，即将位于第一分区中的“x,x”改为“b,b”，获得位于第一分区中的第一目标数据“a,b,b,b,c,d”，相类似的，可以将位于第二分区中的“x”改写为“e”，获得位于第二分区中的第二目标数据为“e,e,e,e,e,f”，对于第三分区中的第一个“x”而言，其与第二分区中的最后一个数据“f”相邻，进而可以将第三分区中的第一个“x”改写为“f”，并将第三分区中的第五个以及第六个“x”改写为“i”，经过上述改写操作，可以获得位于第三分区中的第三目标数据为“f,g,h,i,i,i”。上述的第一目标数据、第二目标数据以及第三目标数据均为构成第二扩展表的目标数据，这样有效地保证了对第二扩展表进行获取的稳定可靠性。

[114] 本实施例中，通过获取每个分区所对应的数据输出量，将各个属性值的重复度确定为用于对第二连接表中各个数据进行扩展操作的扩展次数，而后基于扩展次数和数据输出量确定各个数据所对应的目标分区，并基于数据输出量对各个数据进行扩展，获得扩展数据，之后将扩展数据发送至目标分区，并对扩展数据进行改写，从而可以稳定地获得用于构成第二扩展表的扩展后数据，这样有效地实现了对第二连接表进行稳定地扩展操作。

[115] 图 4 为本申请实施例提供的对扩展次数进行前缀和计算，获得计算后参数的流程示意图；在上述实施例的基础上，参考附图 4 所示，本实施例提供了一种对扩展次数进行前缀和计算的实现方式，具体的，本实施例中的对扩展次数进行前缀和计算，获得计算后参数可以包括步骤 S401 至步骤 S404。

[116] 步骤 S401：获取扩展次数所对应的当前分区。

[117] 在获取到扩展次数之后，可以基于各个扩展次数所对应的数据来确定扩展次数所对应的当前分区，在一些实例中，可以将各个数据所对应的当前分区确定为扩展次数所对应的当前分区。举例来说，在扩展次数为（1,3,1,1,5,2,1,1,3）时，确定上述的扩展次数（1,3,1）所对应的当前分区可以为第一分区，上述的扩展次数（1,5,2）所对应的当前分区可以为第二分区，上述的扩展次数（1,1,3）所对应的当前分区可以为第三分区。

[118] 步骤 S402：对位于同一分区中扩展次数进行前缀和计算处理，获得各个分区所对应的第一计算处理序列，第一计算处理序列包括多个处理结果。

[119] 在获取到扩展次数相对应的当前分区之后，可以对位于同一分区中扩展次数进行前缀和计算处理操作，从而可以获得位于各个分区中的第一计算处理序列，第一计算处理序列包括多个处理结果。在一些实例中，可以利用预先训练好的机器学习模型或者神经网络模型对位于同一分区中扩展次数进行前缀和计算处理，获得各个分区所对应的第一计算处理序列。

[120] 举例来说，在第一分区中存储扩展次数（1,3,1）时，通过对上述的扩展次数进行前缀和计算处理操作之后，则可以获得与第一分区相对应的第一计算处理序列，该第一

计算处理序列可以为：1,4,5。相类似的，在第二分区中存储扩展次数（1,5,2）时，通过对上述的扩展次数进行前缀和计算处理操作之后，则可以获得与第二分区相对应的第一计算处理序列，该第一计算处理序列可以为：1,6,8；在第三分区中存储扩展次数（1,1,3）时，通过对上述的扩展次数进行前缀和计算处理操作之后，则可以获得与第三分区相对应的第一计算处理序列，该第一计算处理序列可以为：1,2,5，这样有效地保证了对各个分区所对应的第一计算处理序列进行确定的准确可靠性。

[121] 步骤 S403：将各个分区所对应的第一计算处理序列中的最后一个处理结果发送至预设分区，获得存储在预设分区中的处理结果序列。

[122] 在获取到各个分区所对应的第一计算处理序列之后，可以将各个分区所对应的第一计算处理序列中的最后一个处理结果发送至预设分区，其中，预设分区可以为所配置的多个分区中的第一个分区、第二个分区或者第 N 个分区等等。以第一个分区作为预设分区为例，在获取到各个分区所对应的第一计算处理序列中的最后一个处理结果发送至第一个分区，从而可以获得存储在预设分区中的处理结果序列。

[123] 举例来说，在第一个分区为预设分区时，若第一分区所对应的第一计算处理序列为：1,4,5；第二分区所对应的第一计算处理序列为：1,6,8；第三分区所对应的第一计算处理序列为：1,2,5，而后可以将上述各个分区所对应的第一计算处理序列中的最后一个处理结果（5,8,5）发送至第一分区，从而可以获得存储在第一分区中的处理结果序列：5,8,5。

[124] 步骤 S404：对处理结果序列进行前缀和计算，获得计算后参数。

[125] 在获取到处理结果序列之后，可以对处理结果序列进行前缀和计算处理，从而可以获得计算后参数，在一些实例中，对处理结果序列进行前缀和计算，获得计算后参数可以包括：对处理结果序列进行前缀和计算，获得第二计算处理序列，第二计算处理序列包括多个计算处理结果；确定第二计算处理序列中各个计算处理结果所对应的目标分区；将计算处理结果发送至目标分区，并对存储在目标分区中的计算处理结果和扩展次数进行求和处理，获得计算后参数。

[126] 其中，确定第二计算处理序列中各个计算处理结果所对应的目标分区可以包括：基于各个计算处理结果所对应的分区标识对第二计算处理序列中包括的多个计算处理结果进行排序，获得结果排序信息；将各个计算处理结果所在的当前排序位置与 1 的和值确定为计算处理结果所对应的目标分区。

[127] 举例来说，在处理结果序列为：5,8,5 时，则可以对处理结果序列进行前缀和计算，获得第二计算处理序列，该第二计算处理序列可以为：5,13,18，而后可以基于上述各个处理结果序列中的处理结果所对应的源分区的标识对上述的第二计算处理序列进行排序，由于 5 对应于第一分区，13 对应于第二分区，18 对应于第三分区，因此，通过对第二计算处理序列进行排序之后，则可以获得结果排序信息，即 5,13,18，即结果排序信息与第二计算处理序列刚好相同。

[128] 而后可以将上述的 5 所对应的目标分区确定为第二分区，将 13 所对应的目标分区确定为第三分区，并将上述的 5 发送至第二分区，将 13 发送至第三分区，而后可以对存储在第二分区中的原始数据与数据“5”进行求和处理，从而可以获得存储在各个分区中的计算后参数，例如：第一分区中存储有：1,3,1；第二分区中存储有：6,8,7；第三分区中存储有：14,14,16，这样稳定地实现了对处理结果序列进行前缀和计算操作，并保证了对计算后参数进行获取的准确可靠性。

[129] 本实施例中，通过获取扩展次数所对应的当前分区，对位于同一分区中扩展次数进行前缀和计算处理，获得各个分区所对应的第一计算处理序列，而后将各个分区所对

应的第一计算处理序列中的最后一个处理结果发送至预设分区，获得存储在预设分区中的处理结果序列，并对处理结果序列进行前缀和计算，从而可以稳定地获得计算后参数，而后则可以基于计算后参数来实现数据表的处理操作，进一步保证了该方法的实用性。

[130] 具体应用时，以分布式分区的数量为三个为例，本应用实施例提供了一种分布式场景下的自然连接算子的不经意高效算法，该自然连接算子的不经意高效算法依赖如下基础算子来实现：主键-外键自然连接算子、排序算子、分组聚合算子、前缀和算子、扩展算子。其中，主键-外键自然连接、排序算子以及分组聚合算子可以直接沿用相关技术中的实现方式，为了便于理解本应用实施例中的不经意高效算法的实现原理和实现效果，下面先对前缀和算子和扩展算子的实现原理进行说明：

[131] 以输入参数为 $\{x_1, x_2, \dots, x_N\}$ 、输出参数为 $\{x_1, x_1 + x_2, \dots, x_1 + x_2 + \dots + x_N\}$ 为例对前缀和算子的实现原理进行说明，其中，上述的“+”运算并不局限于算数加法，可以是任何满足结合率的二元运算，例如：算数乘法、逻辑异或、取较大者、取较小者、取后者（若后者不为预设占位数据 dummy）或者前者（若后者为预设占位数据 dummy）等等，具体的，前缀和算子的实现过程可以包括以下步骤 1 至步骤 3。

[132] 步骤 1：各个分区（第一分区、第二分区和第三分区）对本地的存储数据进行前缀和计算，并将计算结果的最后一个数发送给第一分区（或者预设分区）。

[133] 举例来说，分区的数量 $p=3$ ，分别为第一分区（对应有标识 1）、第二分区（对应有标识 2）以及第三分区（对应有标识 3），数据总量 $N=9$ ，每个分区所存储的数据总量 $n=N/p=3$ 。输入的 N 个数依次为 1,2,3,1,2,1,3,4,2，在执行前缀和算子之前，各个分区所对应的起始数据可以为：第一分区中存储有 1,2,3；第二分区中存储有 1,2,1；第三分区中存储有 3,4,2；在对上述各个分区本地中的存储数据进行前缀和计算之后，则可以获取到以下数据：第一分区-1,3,6；第二分区-1,3,4；第三分区-3,7,9。而后可以针对上述三个分区进行前缀和计算之后的数据 6、4、9 发送至第分区，从而使得第一分区可以获取到 6、4、9。

[134] 步骤 2：第一分区可以按照分区的编号顺序将接收到的 p 个数排列在一起，并进行前缀和的计算操作，而后将第 i 个数发送给标识为 $i+1$ 的分区，其中， $i=1,2,\dots,p-1$ 。

[135] 其中，在第一分区获取到 6、4、9 之后，可以对接收到的 p 个数排列在一起，而后可以对上述的数据进行前缀和计算，获得 6、10、19。而后可以将上述的 6 发送至第 $i+1$ 个分区（即第二分区），上述的 10 发送至第 $i+1$ 个分区（即第三分区）。

[136] 步骤 3：在各个分区接收到相对应的数据之后，可以将所接收到的数据加上各个分区内的经过前缀和计算之后的数值。

[137] 由于第一分区并没有接收到任何数据，因此，第一分区中的数据为 1,3,6；而第二分区接收到数据 6，第二分区中经过前缀和计算之后的数值为 1,3,4，经过加法求和之后可以获得存储在第二分区中的 7,9,10；第三分区接收到数据 10，第三分区中经过前缀和计算之后的数值为 3,7,9，经过加法求和之后可以获得存储在第三分区中的 13,17,19，从而有效地实现了数据的前缀和操作。

[138] 需要注意的是，上述的前缀和计算操作所对应的通信量分别为：上述的步骤 1 的通信量为 $p-1$ ，步骤 2 的通信量为 $p-1$ ，总的通信量为 $2p-2$ 。

[139] 另一方面，以输入参数为 $\{x_1, x_2, \dots, x_N\}$ 、 N 个总和为 M 的正整数 $\{d_1, d_2, \dots, d_N\}$ 作为扩展次数、输出参数为 $\{x_1, \dots, x_1, x_2, \dots, x_2, \dots, x_N, \dots, x_N\}$ 为例对扩展算子的实现原理进行说明，其中， x_i 连续出现 d_i 次， $i=1,2,\dots,N$ 。其中， $n=N/p$ ， $m=M/p$ ，需要注意的是，若 N 或者 M 不是 p 的倍数，补充预设占位数据 dummy 数据使得 N 和 M 都是 p 的倍数。

具体的，扩展算子的实现过程可以包括以下步骤 11 至步骤 14。

[140] 步骤 11: 对数列 $\{0, d_1, d_2, \dots, d_{N-1}\}$ 调用前缀和算子，上述的前缀和算子中的加法为普通的整数加法，记结果为 $\{e_1, e_2, \dots, e_N\}$ 。

[141] 举例来说，分区的数量 $p=3$ ，数据总量 $N=9$ ，每个分区的数据总量 $n=N/p=3$ ，输出数据总量 $M=18$ ，每个分区的输出数据量 $m=M/p=6$ ，此外，为了便于说明，可以将用于控制失败率的常数参数设置为 1。在输入的 N 个数依次为 $a, b, c, d, e, f, g, h, i$ ，对应正整数依次为 $1, 3, 1, 1, 5, 2, 1, 1, 3$ 时，通过对 $\{0, 1, 3, 1, 1, 5, 2, 1, 1\}$ 调用前缀和算子之后，可以获取到结果 $\{0, 1, 4, 5, 6, 11, 13, 14, 15\}$ 。

[142] 步骤 12: 将数据 $\{x_1, x_2, \dots, x_N\}$ 改写为 $(x_1, e_1), (x_2, e_2), \dots, (x_N, e_N)$ 的结构，为每个数据独立地、均匀随机地从 p 个分区中随机选取一个分区作为目标分区，然后将所有数发送到各自指定的目标分区。

[143] 经过上述改写操作后，各个分区存储的本地数据会被打乱，假设打乱后各个分区中的数据可以分别为：

第一分区: $(a, 0), (c, 4), (f, 11), (h, 14)$;

第二分区: $(b, 1), (e, 6), (g, 13)$;

第三分区: $(d, 5), (i, 15)$ 。

[144] 步骤 13: 对于每个分区接收的数据，将数据 $(x_1, e_1), (x_2, e_2), \dots, (x_N, e_N)$ 改写为 $(x_1, r_1), (x_2, r_2), \dots, (x_N, r_N)$ 。

[145] 假设其数据内容为 $(y_1, f_1), (y_2, f_2), \dots, (y_K, f_K)$ ，对于 (y_i, f_i) 而言，假设 $f_i = mq_i + r_i$ ，即 f_i 除以 m 的商是 q_i ，余数是 r_i 。将数据内容改写为 (x_i, r_i) ，并为该数据指定目标分区为第 $q_i + 1$ 个。对于每个分区 j ，若发送到分区 j 的数据总数少于 $\lceil m + c\sqrt{m} \rceil = 9$ 个，其中， c 为预先配置的用于控制失败率的常数参数，若发送至分区 j 的数据总数少于 9 个时，则可以补充预设占位数据 dummy，直至数据总数等于该值，然后将所有数发送到指定的分区。

[146] 若发送至分区 j 的数据总数大于该值，则整个算法失败，此时，可以重新进行数据表的处理操作，或者，可以生成提示信息，以提醒用户基于提示信息对于控制失败率的常数参数进行配置操作。

[147] 经过上述过程，每个分区对分区所发送的数据量会被填充为恰好为 9 个，并将数据发送给对应的分区之后，各个分区的数据可以分别为（ x 表示预设填充的占位数据 dummy）：

第一分区: $(a, 0), (c, 4), x, x, x, x, x, x, x, (b, 1), x, x, x, x, x, x, x, (d, 5), x, x, x, x, x, x, x$;

第二分区: $(f, 5), x, x, x, x, x, x, x, x, (e, 0), x, x, x, x, x, x, x, x, x, x, x, x, x, x, x$;

第三分区: $(h, 2), x, x, x, x, x, x, x, x, (g, 1), x, x, x, x, x, x, x, x, (i, 3), x, x, x, x, x, x, x$ 。

[148] 另外，对于上述的常数参数 c 而言，其用来控制算法的失败概率，通过理论分析，可以证明算法失败的概率小于 $e^{-c^2/2}$ 。在实际场景中，用户可以根据具体的应用需求以及应用要求的失败率来选择或者配置合适的参数 c 。

[149] 步骤 14: 在每个分区接收的数据中，假设非预设占位数据 dummy 的数据为 $(z_1, g_1), \dots, (z_i, g_i)$ ，将数据 z_i 置于第 $g_i + 1$ 个位置，并将空余位置填充 dummy，总数据量为 m 。多余的 dummy 数据丢弃。最后对数据调用前缀和算法，这里前缀和算法中的加法定义为 $x+y=x$ (若 y 为 dummy 数据) 或 y (若 y 不为 dummy 数据)。

[150] 在经过上述处理操作，可以将数据移动到相应位置，并移除多余的 dummy，结果如下：

第一分区：a,b,x,x,c,d;

第二分区：e,x,x,x,x,f;

第三分区：x,g,h,i,x,x。

[151] 而后再调用前缀和算法，所有 dummy 的数据会被改写为其前方相邻的非 dummy 数。注意，此处调用的是全局的前缀和算子，因此，第三分区中的第一个 x 会识别到前方相邻的或者最近的非 dummy 数为 f，而后则可以将第三分区中的“x”改写为 f，从而可以获得处理后的结果，具体如下：

第一分区：a,b,b,b,c,d;

第二分区：e,e,e,e,e,f;

第三分区：f,g,h,i,i,i。

[152] 需要注意的是，上述的扩展操作所对应的通信量分别为：上述的步骤 11 的通信量为 $2p-2$ ，步骤 12 的通信量为 N ，步骤 13 的通信量约为 M (可以忽略)，步骤 14 的通信量为 $2p-2$ ，从而可以获得总通信量约为 $N+M$ 。

[153] 基于上述的前缀和算子、扩展算子，并以 $R(A,B)$ 与 $S(B,C)$ 的两个数据表作为输入、输出为共有 M 行数据的形如 $T(A,B,C)$ 的数据表为例，其中， $R(A,B)$ 与 $S(B,C)$ 中的每个表各有 N 行数据，经过自然连接操作之后，所获得的输出结果共有 M 行，具体的，自然连接算子的不经意高效算法的实现过程可以包括以下步骤 111 至步骤 11110。

[154] 步骤 111：对表 $R(A,B)$ 调用分组聚合算子，计算每个 b 在 $R(A,B)$ 中的重复度，将结果记为表 $R1(B,D)$ ，其中，列 B 在表 $R1$ 中为主键， D 表示主键 B 在 R 中的重复度。

[155] 举例来说， $N=5$ ，表 $R(A,B)$ 的数据为 $\{(a_1, b_1), (a_1, b_2), (a_2, b_2), (a_1, b_3), (a_2, b_4)\}$ ，表 $S(B,C)$ 的数据为 $\{(b_1, c_1), (b_1, c_2), (b_2, c_2), (b_2, c_3), (b_3, c_4)\}$ ，则 R 与 S 的正确自然连接结果 $T(A,B,C)$ 应为 $\{(a_1, b_1, c_1), (a_1, b_1, c_2), (a_1, b_2, c_2), (a_2, b_2, c_2), (a_1, b_2, b_3), (a_2, b_2, c_3), (a_1, b_3, c_4)\}$ ，即 $M=7$ 。针对上述的表 $R(A,B)$ ，计算每个 b 在 $R(A,B)$ 中的重复度，得到 $R1(B,D)$ 为 $\{(b_1, 1), (b_2, 2), (b_3, 1), (b_4, 1)\}$ 。

[156] 步骤 112：对表 $R1$ 和表 S 调用主键-外键自然连接算子，记结果为表 $S2(B,C,D)$ 。

[157] 具体的，通过主键-外键自然连接算子对表 $R1$ 和表 S 进行自然连接操作，获得表 $S2(B,C,D)$ ，上述的表 $S2(B,C,D)$ 可以为 $\{(b_1, c_1, 1), (b_1, c_2, 1), (b_2, c_2, 2), (b_2, c_3, 2), (b_3, c_4, 1)\}$ 。

[158] 步骤 113：对表 $S2$ 调用扩展算子，待扩展数据为列组合 (B,C) ，扩展次数为列 D 。记扩展结果为表 $S3(B,C)$ 。

[159] 具体的，在获取到表 $S2$ 之后，可以利用扩展算子对表 $S2$ 进行扩展处理，得到扩展结果表 $S3(B,C)$ 为 $\{(b_1, c_1), (b_1, c_2), (b_2, c_2), (b_2, c_2), (b_2, c_3), (b_2, c_3), (b_3, c_4)\}$ 。

[160] 步骤 114：调用排序算子，对表 $S3$ 按照列 (B,C) 以字典序的方式进行排序。

[161] 其中，排序算子能够接收 N 个元素（或一个行数为 N 的表）作为输入，按照一定的大小关系对 N 个元素（或 N 行）重新进行排列，然后再输出。当这些元素为包含多个分量的元组时，有时会为大小关系指定多个分量(列)，称为按照多列进行字典序排序，即在比较两个元组时，优先按照指定的第一列进行比较，若元组在此列的数值相同，则按第二列进行比较，如此继续，通过计算可知，上述的排序操作的通信量大约为 $3N$ 。

[162] 步骤 115: 对表 S 调用分组聚合算子, 计算对于每个 b 在 S(B,C)中的重复度, 将结果记为表 S1(B,E), 其中, 列 B 在 S1 表中为主键, E 表示主键 B 在 S 中的重复度。

[163] 其中, 分组聚合算子对应于数据库查询的"group by"查询语句, 该算子可以接收一个表作为输入, 将表的行按指定的列组合进行分组, 对相同组内的另一个指定列进行聚合操作, 如求和、求最大值、求平均值等, 通过计算可知, 上述的分组聚合操作的通信量约为 N。

[164] 具体的, 在获取到表 S 之后, 可以利用分组聚合算子对表 S 进行分析处理, 获得表 S 中与数据列 B 中各个数值 b 所对应的重复度, 从而可以获得结果 S1(B,E)为 $\{(b_1, 2), (b_2, 2), (b_3, 1)\}$ 。

[165] 步骤 116: 对表 S1 和 R 调用主键-外键自然连接算子, 记结果为 R2(A,B,E)。

[166] 其中, 主键-外键自然连接算子为自然连接算子的一种特殊情况, 它接收表 R(A,B)与表 S(B,C)的两个表作为输入, 每个表各有 N 行数据, 且 B 列在 S 表中是唯一的, 即重复度为 1。输出为至多 N 行数据的形如 T(A,B,C)的表, 表的内容为集合, 由于主键-外键自然连接算子需要将两个表合并到一起, 调用一次排序和一次前缀和算子, 因此, 其通信量较大, 具体约为 6N。

[167] 具体的, 在表 S1(B,E) 为 $\{(b_1, 2), (b_2, 2), (b_3, 1)\}$, 表 R 为 $\{(a_1, b_1), (a_1, b_2), (a_2, b_2), (a_1, b_3), (a_2, b_4)\}$ 时, 而后可以利用主键-外键自然连接算子对表 S1 和表 R 进行自然连接操作, 从而可以获得结果 R2(A,B,E) 为 $\{(a_1, b_1, 2), (a_1, b_2, 2), (a_2, b_2, 2), (a_1, b_3, 1)\}$ 。

[168] 步骤 117: 对表 R2 调用扩展算子, 待扩展数据为列组合(A,B), 扩展次数为列 E, 获得扩展结果为 R3(A,B)。

[169] 在获取到表 R2 之后, 可以利用扩展算子对表 R2 进行扩展操作, 获得扩展结果 R3(A,B), 具体的, R3(A,B)可以为 $\{(a_1, b_1), (a_1, b_1), (a_1, b_2), (a_1, b_2), (a_2, b_2), (a_2, b_2), (a_1, b_3)\}$ 。

[170] 步骤 118: 在表 R3 中增加列 F, 初始值为 1, 记结果为 R4(A,B,F)。对 R4 调用前缀和算子, 其中, 加法定义为: 假定待求和的两个元组分别为 (a_1, b_1, f_1) 和 (a_2, b_2, f_2) , 若 $a_1 = a_2$ 且 $b_1 = b_2$, 则它们的和为 $(a_2, b_2, f_1 + f_2)$, 否则为 (a_2, b_2, f_2) 。记结果为 R5(A,B,F)。

[171] 其中, 所增加的 F 列用于确定对表 R3 进行调整的预设顺序, 以将表 R3 与 S 表进行数据对齐操作。另外, 对 R4 调用前缀和算子之后, 所获得的结果 R5(A,B,F)可以为 $\{(a_1, b_1, 1), (a_1, b_1, 2), (a_1, b_2, 1), (a_1, b_2, 2), (a_2, b_2, 1), (a_2, b_2, 2), (a_1, b_3, 1)\}$ 。

[172] 步骤 119: 调用排序算子, 对 R5 按照列(B,F,A)字典序进行排序。

[173] 步骤 1110: 在获取到表 S3 和表 R5 之后, 可以对表 S3 和表 R5 进行合并操作, 获得计算表 T(A,B,C)。

[174] 举例来说, 假设表 S3 中的第 i 行为 (b, c) , 表 R5 中的第 i 行为 (a, b', f) , 对表 S3 和表 R5 进行合并之后, 可以获得计算表 T 中的第 i 行为 (a, b, c) , 具体的, 表 T 的内容可以为: $\{(a_1, b_1, c_1), (a_1, b_1, c_2), (a_1, b_2, c_2), (a_2, b_2, c_2), (a_1, b_2, c_3), (a_2, b_2, c_3), (a_1, b_3, c_4)\}$, 从而有效地保证了数据表处理的质量和效果。

[175] 需要注意的是, 上述实施例中的步骤 111-步骤 113 与步骤 114-步骤 116 是完全对称的, 并且上述实施例中的参数 $b=b'$ 。另外, 上述的扩展操作所对应的通信量分别为: 步骤 111 所对应的通信量为 N, 步骤 112 所对应的通信量为 6N, 步骤 113 所对应的通信量为 N+M, 步骤 114 所对应的通信量为 3M, 步骤 115 所对应的通信量为 N, 步骤 116 所对应的通信量为 6N, 步骤 117 所对应的通信量为 N+M, 步骤 118 所对应的通信量为

p-1, 步骤 119 所对应的通信量为 3M, 步骤 11110 所对应的通信量为 0, 总通信量约为 $16N+8M$ 。

[176] 本应用实施例提供的技术方案, 实现了一种分布式场景下自然连接算子的不经意高效算法, 具体的, 通过设置前缀和算子和扩展算子, 有效地实现了以一种高效、简洁的方式将每个表的数据扩展重复多次, 次数恰好等于它在另一个表的重复度, 再通过排序进行数据的对齐, 从而可以高效地完成自然连接操作, 并且由于每个分区发送的数据量均是相同或者相似的, 这样有效地满足了不经意的要求, 整个数据表的处理操作的通信量较低, 进而有效地解决了相关技术中实现的不经意算法存在的难度较高且效率很低, 在加密数据领域中还存在额外的信息泄露风险的问题, 此外, 上述实现过程也提高了数据表处理的质量和效率, 进一步提高了该方法的实用性, 有利于市场的推广与应用。

[177] 图 5 为本申请实施例提供的一种数据表的处理装置的结构示意图; 参考附图 5 所示, 本实施例提供了一种数据表的处理装置, 该数据表的处理装置用于执行上述图 2 所示的数据表的处理操作, 具体的, 该数据表的处理装置可以包括获取模块 11、扩展模块 12 和处理模块 13。

[178] 获取模块 11, 用于获取第一数据表和第二数据表, 第一数据表和第二数据表位于不同存储在分布式的多个分区中, 且第一数据表和第二数据表中包括相同的属性项。

[179] 扩展模块 12, 用于基于第一数据表中的属性项对第二数据表进行扩展, 获得第二扩展表。

[180] 扩展模块 12, 还用于基于第二数据表中的属性项对第一数据表进行扩展, 获得第一扩展表。

[181] 处理模块 13, 用于在第一扩展表中增加一数据列, 获得调整后扩展表, 数据列中包括的参数值用于对第一扩展表进行重新排序, 以使得第一扩展表与第二扩展表中的数据对齐。

[182] 处理模块 13, 还用于对第二扩展表和调整后扩展表进行关联, 获得与第一数据表和第二数据表相对应的关联数据表。

[183] 在一些实例中, 在扩展模块 12 基于第一数据表中的属性项对第二数据表进行扩展, 获得第二扩展表时, 该扩展模块 12 用于执行: 基于第一数据表中的属性项对第一数据表进行分组聚合处理, 获得第一聚合表, 第一聚合表中包括位于属性项下的各个属性值的重复度; 对第一聚合表和第二数据表进行自然连接, 获得第二连接表; 对第二连接表进行扩展, 获得第二扩展表。

[184] 在一些实例中, 在扩展模块 12 对第二连接表进行扩展, 获得第二扩展表时, 该扩展模块 12 用于执行: 获取每个分区所对应的数据输出量; 将各个属性值的重复度确定为用于对第二连接表中各个数据进行扩展操作的扩展次数; 基于扩展次数和数据输出量, 确定各个数据所对应的目标分区; 基于数据输出量对各个数据进行扩展, 获得扩展数据; 将扩展数据发送至目标分区, 并对扩展数据进行改写, 获得用于构成第二扩展表的扩展后数据。

[185] 在一些实例中, 在扩展模块 12 基于扩展次数和数据输出量, 确定用于对第二连接表中各个数据所对应的目标分区时, 该扩展模块 12 用于执行: 对扩展次数进行前缀和计算, 获得计算后参数; 将 0 值与计算后参数, 确定为处理后参数; 获取处理后参数除以数据输出量的商值; 基于商值与 1 的和值, 确定数据所对应的目标分区。

[186] 在一些实例中, 在扩展模块 12 对扩展次数进行前缀和计算, 获得计算后参数时, 该扩展模块 12 用于执行: 获取扩展次数所对应的当前分区; 对位于同一分区中扩展次

数进行前缀和计算处理,获得各个分区所对应的第一计算处理序列,第一计算处理序列包括多个处理结果;将各个分区所对应的第一计算处理序列中的最后一个处理结果发送至预设分区,获得存储在预设分区中的处理结果序列;对处理结果序列进行前缀和计算,获得计算后参数。

[187] 在一些实例中,在扩展模块 12 对处理结果序列进行前缀和计算,获得计算后参数时,该扩展模块 12 用于执行:对处理结果序列进行前缀和计算,获得第二计算处理序列,第二计算处理序列包括多个计算处理结果;确定第二计算处理序列中各个计算处理结果所对应的目标分区;将计算处理结果发送至目标分区,并对存储在单一目标分区中的计算处理结果和扩展次数进行求和处理,获得计算后参数。

[188] 在一些实例中,在扩展模块 12 确定第二计算处理序列中各个计算处理结果所对应的目标分区时,该扩展模块 12 用于执行:基于各个计算处理结果所对应的分区标识对第二计算处理序列中包括的多个计算处理结果进行排序,获得结果排序信息;将各个计算处理结果所在的当前排序位置与 1 的和值确定为计算处理结果所对应的目标分区。

[189] 在一些实例中,在扩展模块 12 基于数据输出量对各个数据进行扩展,获得扩展数据时,该扩展模块 12 用于执行:基于数据输出量,确定数据输出总位数;基于数据输出总位数对各个数据进行扩展,获得扩展数据。

[190] 在一些实例中,在扩展模块 12 基于数据输出总位数对各个数据进行扩展,获得扩展数据时,该扩展模块 12 用于执行:获取需要发送单一目标分区的所有数据;检测所有数据的数据位总和是否满足数据输出总位数;若不满足,则基于所有数据的数据位总和以及预设填充位进行数据扩展操作,获得扩展数据,扩展数据的数据位数满足数据输出总位数。

[191] 在一些实例中,在扩展模块 12 对扩展数据进行改写,获得用于构成第二扩展表的扩展后数据时,该扩展模块 12 用于执行:获取处理后参数与数据输出量之间进行除法运算时所对应的余数;基于余数确定数据存储在目标分区中的位置;基于位置和数据输出量删除多余的预设填充位,获得排序后数据;将排序后数据中包括的预设填充位改写为前一相邻数据,获得用于构成第二扩展表的目标数据。

[192] 在一些实例中,在扩展模块 12 基于第二数据表中的属性项对第一数据表进行扩展,获得第一扩展表时,该扩展模块 12 用于执行:基于第二数据表中的属性项对第二数据表进行分组聚合处理,获得第二聚合表,第二聚合表中包括位于属性项下的各个属性值的重复度;对第二聚合表和第一数据表进行自然连接,获得第一连接表;对第一连接表进行扩展,获得第一扩展表。

[193] 在一些实例中,在处理模块 13 在第一扩展表中增加一数据列,获得调整后扩展表时,该处理模块 13 用于执行:在第一扩展表增加初始值为 1 的数据列,获得中间扩展表;对中间扩展表中的相邻两组数据的两个属性值进行比较;在相邻两组数据所对应的两个属性值均相同时,则对相邻两组数据中的数据列进行求和处理,获得求和数据列,并将两个属性值与求和数据列确定用于构成调整后扩展表的数据;在相邻两组数据所对应的两个属性值不同时,则将相邻两组数据中的后一个数据确定为用于构成调整后扩展表的数据。

[194] 在一些实例中,在处理模块 13 对第二扩展表和调整后扩展表进行关联,获得与第一数据表和第二数据表相对应的关联数据表时,该处理模块 13 用于执行:获取用于对第一扩展表和调整后扩展表进行排序处理的预设规则;利用预设规则分别对第一扩展表和调整后扩展表进行排序,获得第一排序表和第二排序表;对第一排序表和第二排序表进行关联,获得与第一数据表和第二数据表相对应的关联数据表。

[195] 图 5 所示装置可以执行图 1-图 4 所示实施例的方法, 本实施例未详细描述的部分, 可参考对图 1-图 4 所示实施例的相关说明。该技术方案的执行过程和技术效果参见图 1-图 4 所示实施例中的描述, 在此不再赘述。

[196] 在一个可能的设计中, 图 5 所示数据表的处理装置的结构可实现为一电子设备, 该电子设备可以是控制器、个人电脑、分区等各种设备。如图 6 所示, 该电子设备可以包括: 第一处理器 21 和第一存储器 22。其中, 第一存储器 22 用于存储相对应电子设备执行上述图 1-图 2 所示实施例中提供的数据表的处理方法的程序, 第一处理器 21 被配置为用于执行第一存储器 22 中存储的程序。

[197] 程序包括一条或多条计算机指令, 其中, 一条或多条计算机指令被第一处理器 21 执行时能够实现如下步骤: 获取第一数据表和第二数据表, 第一数据表和第二数据表均存储在分布式的多个分区中, 且第一数据表和第二数据表中包括相同的属性项; 基于第一数据表中的属性项对第二数据表进行扩展, 获得第二扩展表; 基于第二数据表中的属性项对第一数据表进行扩展, 获得第一扩展表; 在第一扩展表中增加一数据列, 获得调整后扩展表, 数据列中包括的参数值用于对第一扩展表进行重新排序, 以使得第一扩展表与第二扩展表中的数据对齐; 对第二扩展表和调整后扩展表进行关联, 获得与第一数据表和第二数据表相对应的关联数据表。

[198] 进一步的, 第一处理器 21 还用于执行前述图 1-图 4 所示实施例中的全部或部分步骤。

[199] 其中, 电子设备的结构中还可以包括第一通信接口 23, 用于电子设备与其他设备或通信网络通信。

[200] 另外, 本申请实施例提供了一种计算机存储介质, 用于储存电子设备所用的计算机软件指令, 其包含用于执行上述图 1-图 4 所示实施例中数据表的处理方法所涉及的程序。

[201] 此外, 本申请实施例提供了一种计算机程序产品, 包括: 存储有计算机指令的计算机可读存储介质, 当计算机指令被一个或多个处理器执行时, 致使一个或多个处理器执行上述图 1-图 4 所示方法实施例中数据表的处理方法中的步骤。

[202] 以上所描述的装置实施例仅仅是示意性的, 其中所述作为分离部件说明的单元可以是或者也可以不是物理上分开的, 作为单元显示的部件可以是或者也可以不是物理单元, 即可以位于一个地方, 或者也可以分布到多个网络单元上。可以根据实际的需要选择其中的部分或者全部模块来实现本实施例方案的目的。本领域普通技术人员在不付出创造性的劳动的情况下, 即可以理解并实施。

[203] 通过以上的实施方式的描述, 本领域的技术人员可以清楚地了解到各实施方式可借助加必需的通用硬件平台的方式来实现, 当然也可以通过硬件和软件结合的方式来实现。基于这样的理解, 上述技术方案本质上或者说对相关技术做出贡献的部分可以以计算机产品的形式体现出来, 本申请可采用在一个或多个其中包含有计算机可用程序代码的计算机可用存储介质 (包括但不限于磁盘存储器、CD-ROM、光学存储器等) 上实施的计算机程序产品的形式。

[204] 本申请是参照根据本申请实施例的方法、设备 (系统)、和计算机程序产品的流程图和/或方框图来描述的。应理解可由计算机程序指令实现流程图和/或方框图中的每一流程和/或方框、以及流程图和/或方框图中的流程和/或方框的结合。可提供这些计算机程序指令到通用计算机、专用计算机、嵌入式处理机或其他可编程设备的处理器以产生一个机器, 使得通过计算机或其他可编程设备的处理器执行的指令产生用于实现在流程图一个流程或多个流程和/或方框图一个方框或多个方框中指定的功能的装

置。

[205] 这些计算机程序指令也可存储在能引导计算机或其他可编程设备以特定方式工作的计算机可读存储器中，使得存储在该计算机可读存储器中的指令产生包括指令装置的制造品，该指令装置实现在流程图一个流程或多个流程和/或方框图一个方框或多个方框中指定的功能。

[206] 这些计算机程序指令也可装载到计算机或其他可编程设备上，使得在计算机或其他可编程设备上执行一系列操作步骤以产生计算机实现的处理，从而在计算机或其他可编程设备上执行的指令提供用于实现在流程图一个流程或多个流程和/或方框图一个方框或多个方框中指定的功能的步骤。

[207] 在一个典型的配置中，计算设备包括一个或多个处理器 (CPU)、输入/输出接口、网络接口和内存。

[208] 内存可能包括计算机可读介质中的非永久性存储器，随机存取存储器 (RAM) 和/或非易失性内存等形式，如只读存储器 (ROM) 或闪存(flash RAM)。内存是计算机可读介质的示例。

[209] 计算机可读介质包括永久性和非永久性、可移动和非可移动媒体可以由任何方法或技术来实现信息存储。信息可以是计算机可读指令、数据结构、程序的模块或其他数据。计算机的存储介质的例子包括，但不限于相变内存 (PRAM)、静态随机存取存储器 (SRAM)、动态随机存取存储器 (DRAM)、其他类型的随机存取存储器 (RAM)、只读存储器 (ROM)、电可擦除可编程只读存储器 (EEPROM)、快闪记忆体或其他内存技术、只读光盘只读存储器 (CD-ROM)、数字多功能光盘 (DVD) 或其他光学存储、磁盒式磁带，磁带磁磁盘存储或其他磁性存储设备或任何其他非传输介质，可用于存储可以被计算设备访问的信息。按照本文中的界定，计算机可读介质不包括暂存电脑可读媒体 (transitory media)，如调制的数据信号和载波。

[210] 最后应说明的是：以上实施例仅用以说明本申请的技术方案，而非对其限制；尽管参照前述实施例对本申请进行了详细的说明，本领域的普通技术人员应当理解：其依然可以对前述各实施例所记载的技术方案进行修改，或者对其中部分技术特征进行等同替换；而这些修改或者替换，并不使相应技术方案的本质脱离本申请各实施例技术方案的精神和范围。

权利要求书

1、一种数据表的处理方法，包括：

获取第一数据表和第二数据表，所述第一数据表和所述第二数据表均存储在分布式的多个分区中，且所述第一数据表和所述第二数据表中包括相同的属性项；

基于所述第一数据表中的属性项对所述第二数据表进行扩展，获得第二扩展表；

基于所述第二数据表中的属性项对所述第一数据表进行扩展，获得第一扩展表；

在所述第一扩展表中增加一数据列，获得调整后扩展表，所述数据列中包括的参数值用于对所述第一扩展表进行重新排序，以使得所述第一扩展表与所述第二扩展表中的数据对齐；

对所述第二扩展表和所述调整后扩展表进行关联，获得与所述第一数据表和第二数据表相对应的关联数据表。

2、根据权利要求1所述的方法，其中，基于所述第一数据表中的属性项对所述第二数据表进行扩展，获得第二扩展表，包括：

基于所述第一数据表中的属性项对所述第一数据表进行分组聚合处理，获得第一聚合表，所述第一聚合表中包括位于所述属性项下的各个属性值的重复度；

对所述第一聚合表和所述第二数据表进行自然连接，获得第二连接表；

对所述第二连接表进行扩展，获得第二扩展表。

3、根据权利要求2所述的方法，其中，对所述第二连接表进行扩展，获得第二扩展表，包括：

获取每个分区所对应的数据输出量；

将各个属性值的重复度确定为用于对所述第二连接表中各个数据进行扩展操作的扩展次数；

基于所述扩展次数和所述数据输出量，确定各个数据所对应的目标分区；

基于所述数据输出量对各个数据进行扩展，获得扩展数据；

将所述扩展数据发送至目标分区，并对所述扩展数据进行改写，获得用于构成所述第二扩展表的扩展后数据。

4、根据权利要求3所述的方法，其中，基于所述扩展次数和所述数据输出量，确定用于对所述第二连接表中各个数据所对应的目标分区，包括：

对所述扩展次数进行前缀和计算，获得计算后参数；

将0值与所述计算后参数，确定为处理后参数；

获取所述处理后参数除以所述数据输出量的商值；

基于所述商值与1的和值，确定数据所对应的目标分区。

5、根据权利要求4所述的方法，其中，对所述扩展次数进行前缀和计算，获得计算后参数，包括：

获取所述扩展次数所对应的当前分区；

对位于同一分区中扩展次数进行前缀和计算处理，获得各个分区所对应的第一计算处理序列，所述第一计算处理序列包括多个处理结果；

将各个分区所对应的第一计算处理序列中的最后一个处理结果发送至预设分区，获得存储在预设分区中的处理结果序列；

对所述处理结果序列进行前缀和计算，获得计算后参数。

6、根据权利要求5所述的方法，其中，对所述处理结果序列进行前缀和计算，获得计算后参数，包括：

对所述处理结果序列进行前缀和计算，获得第二计算处理序列，所述第二计算处理序列包括多个计算处理结果；

确定所述第二计算处理序列中各个计算处理结果所对应的目标分区；

将所述计算处理结果发送至目标分区，并对存储在单一目标分区中的所述计算处理

结果和扩展次数进行求和处理，获得计算后参数。

7、根据权利要求 6 所述的方法，其中，确定所述第二计算处理序列中各个计算处理结果所对应的目标分区，包括：

基于所述各个计算处理结果所对应的分区标识对所述第二计算处理序列中包括的多个计算处理结果进行排序，获得结果排序信息；

将各个计算处理结果所在的当前排序位置与 1 的和值确定为所述计算处理结果所对应的目标分区。

8、根据权利要求 3 所述的方法，其中，基于所述数据输出量对各个数据进行扩展，获得扩展数据，包括：

基于所述数据输出量，确定数据输出总位数；

基于所述数据输出总位数对各个数据进行扩展，获得扩展数据。

9、根据权利要求 8 所述的方法，其中，基于所述数据输出总位数对各个数据进行扩展，获得扩展数据，包括：

获取需要发送同一目标分区的所有数据；

检测所有数据的数据位总和是否满足所述数据输出总位数；

若不满足，则基于所有数据的数据位总和以及预设填充位进行数据扩展操作，获得扩展数据，所述扩展数据的数据位数满足数据输出总位数。

10、根据权利要求 3 所述的方法，其中，对所述扩展数据进行改写，获得用于构成所述第二扩展表的扩展后数据，包括：

获取所述处理后参数与所述数据输出量之间进行除法运算时所对应的余数；

基于所述余数确定所述数据存储在目标分区中的位置；

基于所述位置和所述数据输出量删除多余的预设填充位，获得排序后数据；

将所述排序后数据中包括的预设填充位改写为前一相邻数据，获得用于构成所述第二扩展表的目标数据。

11、根据权利要求 1-10 中任意一项所述的方法，其中，基于所述第二数据表中的属性项对所述第一数据表进行扩展，获得第一扩展表，包括：

基于所述第二数据表中的属性项对所述第二数据表进行分组聚合处理，获得第二聚合表，所述第二聚合表中包括位于所述属性项下的各个属性值的重复度；

对所述第二聚合表和所述第一数据表进行自然连接，获得第一连接表；

对所述第一连接表进行扩展，获得第一扩展表。

12、根据权利要求 1-10 中任意一项所述的方法，其中，在所述第一扩展表中增加一数据列，获得调整后扩展表，包括：

在所述第一扩展表增加初始值为 1 的数据列，获得中间扩展表；

对所述中间扩展表中的相邻两组数据的两个属性值进行比较；

在相邻两组数据所对应的两个属性值均相同时，则对相邻两组数据中的数据列进行求和处理，获得求和数据列，并将两个属性值与求和数据列确定用于构成调整后扩展表的数据；

在相邻两组数据所对应的两个属性值不同时，则将相邻两组数据中的后一个数据确定为用于构成调整后扩展表的数据。

13、根据权利要求 1-10 中任意一项所述的方法，其中，对所述第二扩展表和所述调整后扩展表进行关联，获得与所述第一数据表和第二数据表相对应的关联数据表，包括

获取用于对所述第一扩展表和调整后扩展表进行排序处理的预设规则；

利用预设规则分别对所述第一扩展表和所述调整后扩展表进行排序，获得第一排序表和第二排序表；

对所述第一排序表和第二排序表进行关联，获得与所述第一数据表和所述第二数据

表相对应的关联数据表。

14、一种数据表的处理装置，包括：

获取模块，用于获取第一数据表和第二数据表，所述第一数据表和所述第二数据表存储在分布式的多个分区中，且所述第一数据表和所述第二数据表中包括相同的属性项；

扩展模块，用于基于所述第一数据表中的属性项对所述第二数据表进行扩展，获得第二扩展表；

所述扩展模块，还用于基于所述第二数据表中的属性项对所述第一数据表进行扩展，获得第一扩展表；

处理模块，用于在所述第一扩展表中增加一数据列，获得调整后扩展表，所述数据列中包括的参数值用于对所述第一扩展表进行重新排序，以使得所述第一扩展表与所述第二扩展表中的数据对齐；

所述处理模块，还用于对所述第二扩展表和所述调整后扩展表进行关联，获得与所述第一数据表和第二数据表相对应的关联数据表。

15、一种电子设备，包括：存储器、处理器；其中，所述存储器用于存储一条或多条计算机指令，其中，所述一条或多条计算机指令被所述处理器执行时实现上述权利要求 1-13 中任意一项的方法。

16、一种计算机程序产品，包括：计算机程序，其中，当所述计算机程序被电子设备的处理器执行时，使所述处理器执行上述权利要求 1-13 中任意一项的方法中的步骤。