

# **DATA PROCESSING METHOD AND APPARATUS BASED ON BLOCKCHAIN, DEVICE, AND MEDIUM**

**[0001]** This application claims the priority of Chinese patent No. 202211383918.X, filed on  
5 November 07, 2022 and titled "DATA PROCESSING METHOD AND APPARATUS BASED ON  
BLOCKCHAIN, DEVICE, AND MEDIUM", the content of which is incorporated herein by  
reference in its entirety.

## **FIELD**

10 **[0002]** The present disclosure relates to the technical field of blockchains, and in particular  
to a data processing method and apparatus based on a blockchain, a device, and a medium.

## **BACKGROUND**

**[0003]** In a traditional blockchain system, a database instance is configured to store  
15 blockchain data. Both a blockchain node and the database instance are deployed on the same device,  
and there is a one-to-one correspondence between the blockchain node and the database instance.  
The database instance uses only one hard disk. Data of the blockchain nodes is usually stored on a  
local hard disk of a single machine. When blockchain data has a relatively small data size, this  
single-machine storage method can still meet a storage requirement in short term. However, when  
20 the blockchain is applied in some high-frequency trading scenarios, the storage capacity of the  
blockchain node is limited by the capacity of a single hard disk, and the storage performance is  
limited by a read-write speed of the single hard disk, resulting in the storage requirement being  
difficult to be met.

## **SUMMARY**

**[0004]** According to embodiments of the present disclosure, a data processing method and  
apparatus based on a blockchain, a device, and a medium are provided. The technical solutions are  
described as follows.

**[0005]** According to an aspect of the embodiments of the present disclosure, a data  
30 processing method based on a blockchain is provided. The method includes:

[0006] obtaining a storage instance mapping table based on database configuration information of a blockchain node, the storage instance mapping table including data mapping relationships of N database instances, the data mapping relationships indicating data types of the database instances, and N being a positive integer greater than 1;

5 [0007] determining M candidate database instances from the N database instances in a case that to-be-stored blockchain data is already obtained, data types of the candidate database instances being matched with the blockchain data, and M being a positive integer less than or equal to N; and

[0008] storing the blockchain database to a first database instance among the M candidate database instances according to a data sharding strategy in the database configuration information.

10 [0009] According to an aspect of the embodiments of the present disclosure, a data processing apparatus based on a blockchain is provided. The apparatus includes a mapping table obtaining module, a data type matching module and a data storage module.

[0010] The mapping table obtaining module is configured to obtain a storage instance mapping table based on database configuration information of a blockchain node. The storage  
15 instance mapping table includes data mapping relationships of N database instances, the data mapping relationships indicate data types of the database instances, and N is a positive integer greater than 1.

[0011] The data type matching module is configured to determine M candidate database instances from the N database instances in a case that to-be-stored blockchain data is already  
20 obtained. Data types of the candidate database instances are matched with the blockchain data, and M is a positive integer less than or equal to N.

[0012] The data storage module is configured to store the blockchain database to a first database instance among the M candidate database instances according to a data sharding strategy in the database configuration information.

25 [0013] According to an aspect of the embodiments of the present disclosure, a computer device including a processor, a memory, and a network interface is provided.

[0014] The processor is connected to the memory and the network interface. The network interface is configured to provide a data communication function, the memory is configured to store a computer program, and the processor is configured to call the computer program to cause the  
30 computer device to perform the data processing method based on the blockchain provided by the embodiments of the present disclosure.

[0015] According to an aspect of the embodiments of the present disclosure, a computer-readable storage medium storing a computer program is provided. The computer program is loaded and executed by a processor to cause a computer device with the processor to perform the data processing method based on the blockchain provided by the embodiments of the present disclosure.

[0016] According to an aspect of the embodiments of the present disclosure, a computer program product is provided. The computer program product includes a computer program. The computer program is stored in a computer-readable storage medium. A processor of a computer device reads the computer program from the computer-readable storage medium and executes the computer program, causing the computer device to perform the data processing method based on the blockchain provided by the embodiments of the present disclosure.

[0017] In the embodiments of the present disclosure, a blockchain node obtains a storage instance mapping table based on database configuration information of the blockchain node. The storage instance mapping table includes data mapping relationships of N database instances. One data mapping relationship indicates a data type of one database instance. N is a positive integer greater than 1. The blockchain node in the embodiments of the present disclosure corresponds to multiple database instances, and the multiple database instances are closely related to the data types. When obtaining to-be-stored blockchain data, the blockchain node determines M candidate database instances of which data types are matched with the blockchain data, from the N database instances included in the storage instance mapping table, and then stores the blockchain data to a first database instance among the M candidate database instances according to a data sharding strategy in the database configuration information, where M is a positive integer less than or equal to N. Therefore, with this storage method provided in the embodiments of the present disclosure, multiple database instances can be deployed on one blockchain node according to a service requirement, thereby significantly increasing the storage capacity. Afterwards, according to the data type of the blockchain data, the blockchain data of different data types can be respectively stored to different database instances in a sharding and classification manner, thereby greatly relieving a storage pressure and achieving a distribution effect. It follows that, with the storage method of the embodiments of the present disclosure, the storage capacity and read-write performance of the blockchain node can be multiplied, thereby effectively improving the node storage capability of the blockchain node and meeting a storage requirement.

## BRIEF DESCRIPTION OF THE DRAWINGS

**[0018]** FIG. 1 is a schematic structural diagram of a blockchain node system according to an embodiment of the present disclosure.

**[0019]** FIG. 2 is a schematic diagram of storage extension performed on a single disk according to an embodiment of the present disclosure.

5 **[0020]** FIG. 3 is a schematic diagram of storage extension performed on multiple disks according to an embodiment of the present disclosure.

**[0021]** FIG. 4 is a schematic diagram of distributed classification and sub-database storage extension according to an embodiment of the present disclosure.

10 **[0022]** FIG. 5 is a schematic diagram of a scenario of data storage according to an embodiment of the present disclosure.

**[0023]** FIG. 6 is a schematic flowchart of a data processing method based on a blockchain according to an embodiment of the present disclosure.

**[0024]** FIG. 7 is a schematic flowchart of obtaining a storage instance mapping table according to an embodiment of the present disclosure.

15 **[0025]** FIG. 8 is a schematic structural diagram of sub-database and classification storage of data according to an embodiment of the present disclosure.

**[0026]** FIG. 9 is a schematic diagram of a scenario of storing blockchain data according to an embodiment of the present disclosure.

20 **[0027]** FIG. 10 is a schematic flowchart of a data processing method based on a blockchain according to an embodiment of the present disclosure.

**[0028]** FIG. 11 is a schematic flowchart of accessing to-be-processed data according to an embodiment of the present disclosure.

**[0029]** FIG. 12 is a schematic structural diagram of a data processing apparatus based on a blockchain according to an embodiment of the present disclosure.

25 **[0030]** FIG. 13 is a schematic diagram of a computer device according to an embodiment of the present disclosure.

## DETAILED DESCRIPTION OF EMBODIMENTS

30 **[0031]** The technical solutions in the embodiments of the present disclosure are clearly and completely described below with reference to the accompanying drawings in the embodiments of

the present disclosure. Apparently, the described embodiments are merely some rather than all of the embodiments of the present disclosure. All other embodiments obtained by a person of ordinary skill in the art based on the embodiments of the present disclosure without creative efforts shall fall within the protection scope of the present disclosure.

5   **[0032]**       Reference is made to FIG. 1 which is a schematic structural diagram of a blockchain node system according to an embodiment of the present disclosure. As shown in FIG. 1, the blockchain node system in the embodiments of the present disclosure is a distributed system formed by multiple blockchain nodes connected in the form of network communication. A blockchain network corresponding to the blockchain system is peer to peer networking (P2P network), which is  
10 a distributed application architecture that distributes tasks and workloads between users. It is a form of networking or network formed by P2P computing models at an application layer.

**[0033]**       The blockchain system shown in FIG. 1 includes multiple blockchain nodes, including node 10a, node 10b, node 10c, ..., and node 10n. The blockchain nodes in the blockchain system are computer devices in any form, which are connected to the blockchain network. For  
15 example, the computer device is a terminal device connected to the blockchain network, or a server connected to the blockchain network. A specific form of the blockchain node is not limited herein.

**[0034]**       In an embodiment, the server connected to the blockchain network is an independent physical server, or a server cluster or distributed system composed of multiple physical servers, or a cloud server that provides cloud computing services. The terminal device connected to the  
20 blockchain network includes: a smart phone, a tablet computer, a notebook computer, a desktop computer, a smart speaker, a smart watch, a vehicle-mounted terminal, a smart TV, and other intelligent terminals. An application client run on the terminal device in the embodiments of the present disclosure is an independent client, or an embedded sub-client integrated into a certain client (such as a social client, an education client, and a multimedia client). A type of the client is  
25 not limited here.

**[0035]**       The blockchain network shown in FIG. 1 includes multiple participants, such as a blockchain node (for example, a consensus node or a full node) that needs to store full ledger data, and a blockchain node (for example, a light node) that stores partial ledger data. The light node is configured to store some blockchain databases, that is, storing block header data and transaction  
30 data associated with its own node, instead of storing complete transaction data. In order to cope with scenarios of massive storage and high performance, a classification/sub-database/sharding storage method is provided for the blockchain nodes that need to store the full ledger data according to embodiments of the present disclosure.

**[0036]** A database of any blockchain node in the blockchain network includes a main database and an archiving database. The main database refers to a database currently being used by the blockchain node. The main database includes a database deployed locally on the blockchain node (for example, a local database 110 of node 10a shown in FIG. 1) and a database for remote storage, which has a network connection relationship with the blockchain node (for example, a remote database 120 of node 10a shown in FIG. 1). The archiving database refers to an independent database configured to perform batch processing of blockchain data. The batch processing includes batch transfer, batch deletion, or the like.

**[0037]** In an embodiment, each blockchain node includes a blockchain ledger module, a database service component, and other blockchain modules. The blockchain ledger module is configured to provide functions of operations such as storage, query, modification, and deletion of account data, and send recorded data of the operations on the account data to another node in the block chain system. The another node stores, after verifying that the account data is valid, the recorded data to a temporary block in response to admitting that the account data is valid, and may further send an acknowledgement to a node initiating the operation. The database service component is configured to provide a database service for the remote database of the blockchain node, such as storing or accessing blockchain data in the remote database based on the database service. The database service component is also referred to as a Software Development Kit (SDK).

**[0038]** In an embodiment, other blockchain modules include processing modules such as an object management module, a basic service module, a smart contract module, and an operational detection module. It is understood that the object management module is configured to manage identity information of all blockchain participants, including maintenance of generation of public and private keys (account management), key management, maintenance of correspondence between real identities of objects and blockchain addresses (permission management), and the like. In addition, when authorized, the object management module monitors and audits transaction states of the real identities and provides risk control rule configuration (risk control audit). The basic service module is deployed on all blockchain node devices to verify the validity of service data requests and record valid requests to a storage after the requests are verified to be valid. For a new service data request, the basic service module first analyzes and authenticates the interface adaptation (interface adaptation), then encrypts service information through a consensus algorithm (consensus management), and completely and consistently transmit the information to a shared ledger (network communication) after encryption for recording and storage. The smart contract module is configured to perform registration and issuance of contracts, trigger the contracts, and execute the contracts. A development object defines a contract logic through a certain programming language,

posts the contract logic onto the blockchain (contract registration), calls a key or other events to trigger and execute the contracts according to a logic of terms and conditions of the contracts, and also provides functions of upgrading and logging out the contracts. The operational detection module is mainly configured to perform deployment in a product release process, configuration  
5 modification, contract setting, cloud adaptation, and visual outputting of real-time states during product running, for example: alarming and detecting network conditions, detecting health states of node devices, and the like.

**[0039]** In an embodiment, in order to meet a service storage requirement, the blockchain node in the embodiments of the present disclosure is deployed with one or more hard disks, and  
10 each hard disk is deployed with multiple database instances. For ease of understanding, further, reference is made to FIG. 2, which is a schematic diagram of storage extension performed on a single disk according to an embodiment of the present disclosure. As shown in FIG. 2, a blockchain node in the embodiments of the present disclosure is a blockchain node that is deployed with a single hard disk in a host (namely, a main body part excluding input and output devices). The  
15 blockchain node is any blockchain node in the blockchain network shown in FIG. 1. For example, the blockchain node is node 10a. The blockchain node can be deployed with one hard disk, such as a hard disk A shown in FIG. 2.

**[0040]** When a read-write speed of a single database instance cannot meet a service read-write requirements of the blockchain node, and the read-write speed of the database instance  
20 does not reach an upper limit of the read-write performance of the hard disk, multiple database instances are deployed in the single hard disk based on the storage performance of the hard disk A in the embodiment of the present disclosure, in order to fully utilize read-write resources of the hard disk. Three database instances are taken as an example here, including database instance D<sub>1</sub>, database instance D<sub>2</sub>, and database instance D<sub>3</sub>. The storage performance of the hard disk A is  
25 jointly determined by the storage capacity and read-write performance of the hard disk A.

**[0041]** Reference is made to FIG. 3, which is a schematic diagram of storage extension performed on multiple disks according to an embodiment of the present disclosure. As shown in FIG. 3, a blockchain node in the embodiments of the present disclosure is a blockchain node that is deployed with multiple hard disks in a host. The blockchain node is any blockchain node in the  
30 blockchain network shown in FIG. 1. For example, the blockchain node is node 10a. Due to a possible difference in the storage performance of different hard disks, a quantity of database instances deployed on different hard disks varies according to the storage performances of the hard disks.

[0042] As shown in FIG. 3, two hard disks are deployed on the blockchain node for example, including a hard disk A and a hard disk B. The storage performance of the hard disk B is superior to that of the hard disk A. Based on this, when the storage extension scheme shown in FIG. 2 above still cannot meet the service requirement of the blockchain node, more database instances are deployed and the database instances are distributed on different hard disks according to this embodiment of the present disclosure, thereby achieving a distribution effect and achieving multiplication in the storage capacity and read-write performance.

[0043] According to the embodiment of the present disclosure, multiple database instances are deployed in the hard disk A based on the storage performance of the hard disk A for example, including database instance D<sub>1</sub>, database instance D<sub>2</sub>, and database instance D<sub>3</sub>. In the same way, according to the embodiment of the present disclosure, multiple database instances are deployed in the hard disk B based on the storage performance of the hard disk B for example, including database instance D<sub>4</sub>, database instance D<sub>5</sub>, database instance D<sub>6</sub>, and database instance D<sub>7</sub>.

[0044] Reference is made to FIG. 4, which is a schematic diagram of distributed classification storage extension to different databases according to an embodiment of the present disclosure. As shown in FIG. 4, a blockchain node in the embodiments of the present disclosure is any blockchain node in the blockchain network shown in FIG. 1. For example, the blockchain node is node 10a. When the storage performance of a single machine of the blockchain node is difficult to meet a service requirement, distributed storage extension is performed in the embodiment of the present disclosure, that is, performing storage through a database agent. In an embodiment, the database agent and the blockchain node are deployed on the same computer device, or are deployed on different computer devices. The deployment of the database agent is not limited here.

[0045] In a remote database associated with the blockchain node, a corresponding host cluster is deployed for each data type of X data types of blockchain data received by the blockchain node according to the embodiment of the present disclosure. X is a positive integer. In the embodiments of the present disclosure, N database instances are deployed in the remote database associated with the blockchain node. The remote database includes host clusters corresponding to the X data types. One data type corresponds to one host cluster, and each host cluster includes multiple database host nodes.

[0046] As shown in FIG. 4, three data types are taken as an example in the embodiments of the present disclosure, including a first data type, a second data type, and a third data type. The first data type refers to a data type corresponding to a small amount of data frequently accessed. For example, the first data type includes a block header data type, a state data type, and the like. The

block header data type refers to a data type to which block header data belongs, and the state data type refers to a data type to which state data belongs. The state data includes a risk state of an enterprise object, a state of an electronic bill, an asset state of a service object, and the like. The host cluster corresponding to the first data type is a host cluster 1 shown in FIG. 4. The host cluster 1 includes multiple hosts, such as host J<sub>11</sub>, host J<sub>12</sub>, ..., and host J<sub>1N</sub>. Each host in the host cluster 1 is deployed with a multiple hard disks. For example, host J<sub>11</sub> is deployed with hard disks A and B, and each hard disk is deployed with multiple database instances.

**[0047]** The second data type refers to a data type of blockchain data that needs to be stored independently. For example, the second data type includes a transaction data type. The transaction data type refers to a data type to which transaction data belongs. The transaction data is equivalent to transaction list details. In some scenarios, the transaction list details do not need to be store for long time, so these details need to be stored in an independent database instance, which is conducive to data archiving or regular deletion subsequently. The host cluster corresponding to the second data type is a host cluster 2 shown in FIG. 4. The host cluster 2 includes multiple hosts, such as host J<sub>21</sub>, host J<sub>22</sub>, ..., and host J<sub>2N</sub>. Each host in the host cluster 2 is deployed with multiple hard disks. For example, host J<sub>21</sub> is deployed with hard disks C and D, and each hard disk is deployed with multiple database instances.

**[0048]** The third data type refers to another data type corresponding to blockchain data that needs to be stored independently. For example, the third data type includes a contract data type. The contract data type refers to a data type to which contract data belongs. The contract data refers to real service data. The contract data is independently stored in multiple database instances to reduce a data volume in a single database, thereby effectively improving the read-write performance of the database, and thus improving the storage performance of the blockchain node. The host cluster corresponding to the third data type is a host cluster 3 shown in FIG. 4. The host cluster 3 includes multiple hosts, such as host J<sub>31</sub>, host J<sub>32</sub>, ..., and host J<sub>3N</sub>. Each host in the host cluster 3 is deployed with multiple hard disks. For example, host J<sub>31</sub> is deployed with hard disks E and F, and each hard disk is deployed with multiple database instances.

**[0049]** For ease of understanding, further, reference is made to FIG. 5, which is a schematic diagram of a scenario of data storage according to an embodiment of the present disclosure. As shown in FIG. 5, a blockchain node in the embodiments of the present disclosure is a blockchain node that needs to store full ledger data in a blockchain network. For example, the blockchain node is any blockchain node in the blockchain network shown in FIG. 1. For example, the blockchain node is node 10a.

**[0050]** It is understood that according to the embodiments of the present disclosure, blockchain data that needs to be stored is classified into X data types, so as to facilitate storage of the blockchain data in different databases according to the data types, thereby improving the storage performance of the node. In which, X is a positive integer. For ease of elaboration, three data types  
5 are taken as an example in the embodiments of the present disclosure, including a first data type 21, a second data type 22, and a third data type 23.

**[0051]** When the blockchain node is started, the blockchain node obtains a storage instance mapping table based on database configuration information of the blockchain node. In a case that the blockchain node is started for the first time, the blockchain node generates the storage instance  
10 mapping table based on the database configuration information. In a case that the blockchain node is not started for the first time, the blockchain node directly loads the generated storage instance mapping table based on the database configuration information.

**[0052]** The storage instance mapping table includes data mapping relationships of N database instances, and one data mapping relationship indicates a data type of one database instance, and N is a positive integer greater than 1. In an embodiment, the N database instances are deployed  
15 according to a service requirement of the blockchain node. That is, all the N database instances are deployed in a local database of the blockchain node (for example, the deployment method shown in FIG. 2 above), or all the N database instances are deployed in a remote database of the blockchain node (for example, the deployment method shown in FIG. 4 above), or the N database instances are  
20 deployed in both the local database and the remote database. The deployment of the database instances is not limited here.

**[0053]** Reference is made to Table 1, which is a storage instance mapping table according to an embodiment of the present disclosure. The storage instance mapping table shown in Table 1 includes data mapping relationships of N database instances. It is assumed that N is equal to 6. The  
25 N database instances include database instance D<sub>1</sub>, database instance D<sub>2</sub>, database instance D<sub>3</sub>, database instance D<sub>4</sub>, database instance D<sub>5</sub>, and database instance D<sub>6</sub>. In the embodiments of the present disclosure, the database instances of different data types can be deployed on the same hard disk or different hard disks. For example, the database instances corresponding to the data type 21, the data type 22, and the data type 23 can be deployed on the same hard disk (for example, the hard  
30 disk A shown in FIG. 2). In an embodiment, the database instances corresponding to the data type 21 and the data type 22 can be deployed on the same hard disk (for example, the hard disk A shown in FIG. 3), while the database instances corresponding to the data type 23 can be deployed on another hard disk (for example, the hard disk B shown in FIG. 3).

Table 1

| Data type    | Database instance                |
|--------------|----------------------------------|
| Data type 21 | Database instance D <sub>1</sub> |
| Data type 22 | Database instance D <sub>2</sub> |
|              | Database instance D <sub>3</sub> |
| Data type 23 | Database instance D <sub>4</sub> |
|              | Database instance D <sub>5</sub> |
|              | Database instance D <sub>6</sub> |

**[0054]** The data mapping relationship of database instance D<sub>1</sub> indicates that the data type of database instance D<sub>1</sub> is the data type 21; the data mapping relationship of database instance D<sub>2</sub> indicates that the data type of database instance D<sub>2</sub> is the data type 22; the data mapping relationship of database instance D<sub>3</sub> indicates that the data type of database instance D<sub>3</sub> is the data type 22; the data mapping relationship of database instance D<sub>4</sub> indicates that the data type of database instance D<sub>4</sub> is the data type 23; the data mapping relationship of database instance D<sub>5</sub> indicates that the data type of database instance D<sub>5</sub> is the data type 23; and the data mapping relationship of database instance D<sub>6</sub> indicates that the data type of database instance D<sub>6</sub> is the data type 23.

**[0055]** In the embodiments of the present disclosure, since blockchain data of different data types have different data volumes and different access frequencies, storage capacities occupied by the blockchain data of different data types may be different, resulting in different quantities of database instances deployed for different data types.

**[0056]** It is understood that the blockchain node needs to determine, in a case that to-be-stored blockchain data is already obtained, M candidate database instances from the N database instances included in the storage instance mapping table shown in Table 1 above. Data types of the candidate database instances are matched with the blockchain data, and M is a positive integer less than or equal to N. As shown in FIG. 5, when the blockchain node obtains blockchain data Q that needs to be stored, the blockchain node can determine a data type of the blockchain data Q. If the data type of the blockchain data Q is the data type 22, the blockchain node determines, from the storage instance mapping table shown in Table 1, the database instances (for example, database instance D<sub>2</sub> and database instance D<sub>3</sub>) corresponding to the data type 22 to be database instances matched with the blockchain data Q.

**[0057]** Further, the blockchain node determines, according to a data sharding strategy in the database configuration information, the database instances (namely, the first database instance, for example, database instance  $D_3$ ) to which the blockchain data  $Q$  needs to be stored, and stores the blockchain data  $Q$  to database instance  $D_3$ .

5 **[0058]** It can be seen that one blockchain node can be deployed with  $N$  database instances according to a service requirement. This means that the storage capacity of the blockchain node can be improved. For blockchain data of any data type, the blockchain data can be stored to a corresponding database instance according to the implementation of storing the blockchain data  $Q$  to the first database instance. That is, blockchain data of different data types can be stored  
10 separately to different database instances, thereby greatly alleviating a storage pressure and achieving a distribution effect. This not only solves the problem of inferior data read-write performance of the blockchain node, but also solves the problem of a limited storage space of a single machine. Even through refined contract storage management according to the data types and the data sharding strategy, implementation of the blockchain in a storage scenario with a large data  
15 volume becomes possible.

**[0059]** For the implementation of classification, sub-database, and sharding storage of the to-be-stored blockchain data by the blockchain node in the blockchain network deployed with the  $N$  database instances, one may refer to embodiments in FIG. 6 to FIG. 11 below.

**[0060]** Reference is made to FIG. 6, which is a schematic flowchart of a data processing  
20 method based on a blockchain according to an embodiment of the present disclosure. As shown in FIG. 6, the method is performed by a computer device. The computer device is a blockchain node in a blockchain network. The blockchain node is a blockchain node that needs to store full ledger data in the blockchain network. For example, the blockchain node is any blockchain node in the blockchain network shown in FIG. 1. For example, the blockchain node is node 10a. The method  
25 may include at least step 601 to step 603 below.

**[0061]** In step 601, the blockchain node obtains a storage instance mapping table based on database configuration information of a blockchain node. The storage instance mapping table includes data mapping relationships of  $N$  database instances, the data mapping relationships indicate data types of the database instances, and  $N$  is a positive integer greater than 1.

30 **[0062]** The storage instance mapping table includes data mapping relationships of  $N$  database instances. One data mapping relationship indicates a data type of one database instance.  $N$  is a positive integer greater than 1. In an embodiment, a service object (for example, a user) performs a starting operation on the blockchain node. In response to the starting operation, the

blockchain node obtains the storage instance mapping table of the blockchain node based on the database configuration information of the blockchain node. The starting operation refers to a triggering operation used for starting the blockchain node. The triggering operation includes contact operations such as click and long press, or non-contact operations such as voice and gesture. A type of the triggering operation is not limited here.

**[0063]** In an embodiment, in a case that the starting operation indicates that blockchain node is started for the first time, the blockchain node needs to be initialized, that is, the database configuration information of the blockchain node is loaded, and the N database instances are established based on the database configuration information and a database type of a main database of the blockchain node. After the initialization succeeds, the blockchain node creates a mapping relationship between each database instance and a data type of a hard disk to which the database instance belongs, and generates the storage instance mapping table based on the created mapping relationship.

**[0064]** The database configuration information refers to information that is configured by a configuration object (for example, a configuration user of the blockchain node) for the blockchain node based on a service storage requirement and is associated with databases of the blockchain node. For example, the database configuration information includes a database storage path, an instance quantity configured for hard disks of the blockchain node, a data sharding strategy, and the like. The data storage path includes multiple storage paths configured by the configuration object for blockchain data of different data types. The instance quantity is determined based on the storage performance of the hard disks deployed on the blockchain node. The data sharding strategy refers to a computing method used by the blockchain node to store the blockchain data, for example, a consistent hashing algorithm.

**[0065]** For example, when the configuration object determines that the data type of the blockchain data includes X data types (X is a positive integer), the configuration object configures different database instances for each data type to store different types of blockchain data into different database instances, in order to improve the node performance of the blockchain node. Certainly, the same type of data may be stored into multiple database instances, and the hard disks corresponding to the database instances may be deployed in a local database of the blockchain node or in a remote database of the blockchain node. This is not limited here.

**[0066]** In an embodiment, both the local database and the remote database of the blockchain node can be deployed with one or more hard disks. The database storage path in the database

configuration information in the embodiments of the present disclosure indicates a hard disk set associated with the X data types, including storage paths corresponding to the various data types.

**[0067]** For example, the X data types include at least one of a first data type, a second data type, or a third data type. The first data type includes at least one of a block header data type or a state data type. The second data type includes a transaction data type. The third data type includes a contract data type. Due to a relatively high read-write frequency and relatively small data volume of blockchain data of the first data type, when configuring database instances for the first data type, the configuration object selects a machine (that is, hard disk) with a relatively small storage capacity and inferior storage performance, so as to significantly save the costs while meeting a service requirement. The storage performance of the hard disk of the first data type is inferior to that of a hard disk of the second data type and that of a hard disk of the third data type.

**[0068]** Reference is made to Table 2, which is a schematic pseudo-code table associated with the database configuration information according to an embodiment of the present disclosure. For ease of elaboration, four hard disks associated with the X data types are deployed on the blockchain node in the embodiments of the present disclosure for example, including hard disk A (such as data1), hard disk B (such as data2), and hard disk D (such as data4) deployed in the local database, as well as hard C (such as data3) deployed in the remote database.

**[0069]** The computer device corresponding to the remote database may be represented by a data format of endpoint: port, as shown in Table 2:

Table 2

```
"leveldb": [  
  {  
    "dataType": "blockHeader",  
    "rootPath": "/data1/blockHeader"  
  },  
  {  
    "dataType": "transaction",  
    "rootPath": "/data1/transaction",  
    "instance": 3,  
    "shardingStrategy": "hash"  
  },  
  {  
    "dataType": "transaction",
```

```

    "rootPath": "/data2/transaction",
    "instance": 3,
    "shardingStrategy": "hash"
  },
  {
    "dataType": "transaction",
    "rootPath": "xx.xx.xx.xx:xx@/data3/transaction",
    "instance": 3,
    "shardingStrategy": "hash"
  },
  {
    "dataType": "contract",
    "rootPath": "/data4/contract",
    "instance": 3,
    "shardingStrategy": "hash"
  }

```

**[0070]** As shown in Table 2, the storage performance of the hard disk A in the local database is inferior to that of other hard disks. Therefore, when the first data type is the block header data type, the storage path corresponding to the first data type in the database configuration information is directory "/data1/blockHeader". When the second data type is the transaction data type, the storage path corresponding to the second data type in the database configuration information includes directory "/data1/transaction, /data2/transaction" in the local database and directory "/data3/transaction" in the remote database. This means that the configuration object configures three storage positions for the transaction data type. When the third data type is the contract data type, the storage path corresponding to the third data type in the database configuration information includes directory "/data4/contract" in the local database. When multiple database instances are deployed in the same directory, root directories of the database instances each include a target prefix and a serial number of an instance. For example, if multiple database instances are deployed on the hard disk A, the storage path corresponding to the transaction data type is "/data1/transaction/1, /data1/transaction/2, /data1/transaction/3".

**[0071]** For ease of understanding, reference is made to FIG. 7, which is a schematic flowchart of obtaining a storage instance mapping table according to an embodiment of the present disclosure. As shown in FIG. 7, a blockchain node in the embodiments of the present disclosure is any blockchain node in the blockchain network shown in FIG. 1. For example, the blockchain node is node 10a.

**[0072]** In a case that a starting operation performed on the blockchain node indicates that the blockchain node is started for the first time, the blockchain node executes step 702 after step 701 of starting the blockchain node is executed, so as to load database configuration information. Further, the blockchain node can obtain a database type of a main database of the blockchain node,  
5 so as to determine whether to automatically execute step 703 to step 704 according to a smart contract on a blockchain to establish database instances or manually establish database instances according to a configuration object of the blockchain node.

**[0073]** The database type of the main database includes a first type (for example, a type that database instances can be automatically established according to the smart contract) and a second  
10 type (for example, a type that database instances need to be manually established).

**[0074]** In a case that the database type of the blockchain node is the first type, the blockchain node obtains the smart contract from the blockchain to which the blockchain node belongs. The smart contract is used for establishing database instances. The smart contract is a computerized protocol, which can be used for executing terms and conditions of a contract and is  
15 implemented by using codes that are deployed in a shared account book and that are executed when a condition is satisfied. The codes are used for completing, according to an actual service requirement, an automated transaction. Further, the blockchain node respectively establishes a database instance for each hard disk in the hard disk set based on the smart contract to obtain the N database instances. In an embodiment, the blockchain node determines, for each hard disk in the  
20 hard disk set, an instance quantity of each hard disk based on the database configuration information.

**[0075]** Further, the blockchain node executes step 703 to determine whether a to-be-established hard disk is deployed in the local database. In an embodiment, in a case that the hard disk is deployed in the local database in the main database, the blockchain node executes step  
25 704 to establish, based on the smart contract, the database instance corresponding to the instance quantity for the hard disk. In an embodiment, the blockchain node calls a database service component in a case that the hard disk is deployed in a remote database in the main database, and establishes, based on the smart contract, the database instance corresponding to the instance quantity for the hard disk. The database service component is configured to provide a remote  
30 database service. That is, the blockchain node jumps to execute step 707 to step 709, calls the database service component, and establishes, based on the remote database service provided by the database service component, the database instance corresponding to the instance quantity. When the

establishment of the database instance is completed, the database service component receives a response from the database instance.

**[0076]** In an embodiment, in a case that the database type of the blockchain node is the second type, the blockchain node displays prompt information associated with the database configuration information to enable the configuration object to perform a triggering operation based on the prompt information. The prompt information may be information corresponding to types such as images, videos, audios and texts. The information type of the prompt information is not limited here. The blockchain node respectively establishes a database instance for each hard disk in the hard disk set in response to a triggering operation performed on the prompt information, to obtain the N database instances.

**[0077]** When the establishment of both the local database and the remote database is completed, the blockchain node executes step 705 to create a mapping relationship between each database instance and a data type of a hard disk to which the database instance belongs, and generate the storage instance mapping table based on the created mapping relationship, so as to execute step 706 subsequently and provide data services (for example, a data access service) for the blockchain node.

**[0078]** For ease of understanding, reference is made to FIG. 8, which is a schematic structural diagram of sub-database and classification storage of data according to an embodiment of the present disclosure. As shown in FIG. 8, a blockchain node in the embodiments of the present disclosure is a blockchain node that is deployed with multiple hard disks in a host. The blockchain node is any blockchain node in the blockchain network shown in FIG. 1. For example, the blockchain node is node 10a.

**[0079]** The blockchain node shown in FIG. 8 loads database configuration information in response to a starting operation. Storage path information in the database configuration information indicates a hard disk set associated with three data types (for example, a block header data type, a transaction data type, and a contract data type). The hard disk set includes a hard disk A and a hard disk B deployed locally according to a service requirement.

**[0080]** The blockchain node can establish database instances for the main database according to a database type (for example, a first type) of a main database and the database configuration information. A storage path of the database configuration information indicates instance quantities of different data types in each hard disk. For example, an instance quantity (namely, a first instance quantity) of the block header data type in the hard disk A is 1; an instance quantity (namely, a second instance quantity) of the transaction data type in the hard disk A is 2;

and an instance quantity (namely, a third instance quantity) of the contract data type in the hard disk B is 3.

**[0081]** The hard disk A is taken as an example. The blockchain node determines, based on the database configuration information, that the instance quantity of the hard disk A is a sum (namely, 3) of the first instance quantity and the second instance quantity. In this case, the blockchain node establishes three database instances for the hard disk A based on a smart contract, including database instance D<sub>1</sub>, database instance D<sub>2</sub>, and database instance D<sub>3</sub>.

**[0082]** Similarly, the hard disk B is taken as an example. The blockchain node determines, based on the database configuration information, that the instance quantity of the hard disk B is the third instance quantity (namely, 3). In this case, the blockchain node establishes three database instances for the hard disk B based on a smart contract, including database instance D<sub>4</sub>, database instance D<sub>5</sub>, and database instance D<sub>6</sub>.

**[0083]** Further, the blockchain node creates a mapping relationship between each of the six database instances and a data type of a hard disk to which the database instance belongs, and generates the storage instance mapping table shown in Table 1 above based on the created mapping relationship. In an embodiment, a storage method for the storage instance mapping table is Key-Value storage, KV storage for short. That is, the blockchain node determines Key as a data type and determines Value as a database instance.

**[0084]** The storage instance mapping table in the embodiments of the present disclosure can be an independent mapping table including the X data types as shown in Table 1 above, or a mapping table composed of X storage instance mapping subtables. A type of the storage instance mapping table is not limited here. For example, when the blockchain node performs the storage extension by using the method shown in FIG. 4 above, the blockchain node establishes a storage instance mapping subtable for each data type, so as to quickly determine, through a database agent, a host cluster that needs to be accessed from the X host clusters, thereby improving the access efficiency.

**[0085]** In step 602, the blockchain node determines M candidate database instances from the N database instances in a case that to-be-stored blockchain data is already obtained. Data types of the candidate database instances are matched with the blockchain data, and M is a positive integer less than or equal to N.

**[0086]** In an embodiment, in a case that the to-be-obtained blockchain data is already obtained, the blockchain node determines data information of the blockchain data and then

determines a data type of the blockchain data based on the data information of the blockchain data. Further, the blockchain node searches the storage instance mapping table for a data type matching with the data type of the blockchain data, and determines database instances corresponding to the found data type to be the candidate database instances, thereby obtaining the M candidate database instances.

**[0087]** In an embodiment, the database configuration information includes a data volume threshold and an access frequency threshold which are configured according to the service requirement. In a case that the data information indicates that a data volume of the blockchain data does not reach the data volume threshold and a data access frequency of the blockchain data reaches the access frequency threshold, it is indicated that the blockchain data is a small amount of data frequently accessed. In this case, the blockchain node determines the first data type to be the data type of the blockchain data. In an embodiment, in a case that the data information indicates that the blockchain data is transaction data, the blockchain node determines the second data type to be the data type of the blockchain data. For example, the transaction data includes asset transfer transaction data, electronic bill transfer transactions, or the like. In an embodiment, the blockchain node determines the third data type to be the data type of the blockchain data in a case that the data information indicates that the blockchain data is contract data. Certainly, the X data types in the embodiments of the present disclosure may further include other data types, and the data types are not exemplified one by one here.

**[0088]** As shown in FIG. 5, when the blockchain node obtains blockchain data Q that needs to be stored, the blockchain node determines a data type of the blockchain data Q. If the data type of the blockchain data Q is the data type 22, the blockchain node determines, from the storage instance mapping table shown in Table 1, the database instances (for example, database instance D<sub>2</sub> and database instance D<sub>3</sub>) corresponding to the data type 22 to be two candidate database instances matching with the blockchain data Q.

**[0089]** In step 603, the blockchain node stores the blockchain database to a first database instance among the M candidate database instances according to a data sharding strategy in the database configuration information.

**[0090]** In an embodiment, the blockchain node generates, from a hash value space along a storage direction based on the data sharding strategy in the database configuration information, a hash ring associated with the data type of the blockchain data. Further, the blockchain node performs hash computation on first key information (such as an address, a directory, and other inputted information) of each candidate database instance among the M candidate database

instances to obtain first position information of each candidate database instance on the hash ring. Similarly, the blockchain node performs hash computation on second key information (such as a data hash value) of the blockchain data to obtain second position information of the blockchain data on the hash ring. Further, the blockchain node searches the hash ring along the storage direction for  
5 first position information having a closest position relationship with the second position information, and determines a database instance corresponding to the found first position information to be the first database instance, thereby storing the blockchain data to the first database instance.

**[0091]** The embodiments of the present disclosure involve X data types. When the blockchain data of the X data types are stored in separate databases, this means that the  
10 embodiments of the present disclosure involve X groups of database instances, and each group of database instances correspond to one hash ring, that is, the embodiments of the present disclosure involve X hash rings. When the data types include the block header data type, the transaction data type, and the contract data type, this embodiment of the present disclosure involves three hash rings, and the three hash rings respectively correspond to the database mapping relationships of the three  
15 data types.

**[0092]** For ease of understanding, reference is made to FIG. 9, which is a schematic diagram of a scenario of storing blockchain data according to an embodiment of the present disclosure. As shown in FIG. 9, the hash ring in the embodiments of the present disclosure is a hash ring corresponding to the data type of the to-be-stored blockchain data. The hash ring is generated from  
20 a hash value space along a storage direction (for example, a clockwise direction) on based on a data sharding strategy (for example, a consistent hashing algorithm) in database configuration information. The hash value space refers to a numerical space from 0 to  $(2^{32})-1$ .

**[0093]** In an embodiment, four candidate database instances are determined by the blockchain node from the storage instance mapping table, including database instance D<sub>1</sub>, database  
25 instance D<sub>2</sub>, database instance D<sub>3</sub>, and database instance D<sub>4</sub> shown in FIG. 9. The blockchain node performs hash computation on first key information of each of the four database instances to obtain first position information of each database instance on the hash ring.

**[0094]** The to-be-stored blockchain data obtained by the blockchain node includes blockchain data Q<sub>61</sub>, blockchain data Q<sub>62</sub>, blockchain data Q<sub>63</sub>, blockchain data Q<sub>64</sub>, blockchain data  
30 Q<sub>65</sub>, and blockchain data Q<sub>66</sub> shown in FIG. 9. For example, for database instance D<sub>1</sub>, the blockchain node obtains directory information, device information (for example, a device identifier), and the like of database instance D<sub>1</sub> and determines the obtained information to be first key information of database instance D<sub>1</sub>. Further, the blockchain node performs the hash computation

on the first key information of database instance  $D_1$  according to the consistent hashing algorithm to obtain a hash value of database instance  $D_1$ . The hash value of database instance  $D_1$  is mapped onto the hash ring to obtain first position information of database instance  $D_1$  on the hash ring.

**[0095]** The blockchain node performs hash computation on second key information of the blockchain data to obtain second position information of the blockchain data on the hash ring. For example, for blockchain data  $Q_{61}$ , the blockchain node needs to determine second key information of blockchain data  $Q_{61}$  based on a data type of blockchain data  $Q_{61}$ . Further, the blockchain node performs the hash computation on the second key information of blockchain data  $Q_{61}$  according to the consistent hashing algorithm to obtain a hash value of blockchain data  $Q_{61}$ . The hash value of blockchain data  $Q_{61}$  is mapped onto the hash ring to obtain second position information of blockchain data  $Q_{61}$  on the hash ring.

**[0096]** In a case that the data type of blockchain data  $Q_{61}$  is a block header data type, the blockchain node takes a block header hash value of blockchain data  $Q_{61}$  as the second key information of blockchain data  $Q_{61}$ . In an embodiment, in a case that the data type of blockchain data  $Q_{61}$  is a transaction data type, the blockchain node takes a transaction hash value of blockchain data  $Q_{61}$  as the second key information of blockchain data  $Q_{61}$ . In an embodiment, in a case that the data type of blockchain data  $Q_{61}$  is a contract data type, the blockchain node takes a contract hash value of blockchain data  $Q_{61}$  as the second key information of blockchain data  $Q_{61}$ .

**[0097]** The blockchain node searches, based on the second position information of the blockchain data, the hash ring along the storage direction for first position information having a closest position relationship with the second position information, and determines a database instance corresponding to the found first position information to be the first database instance, thereby storing the blockchain data to the first database instance.

**[0098]** For example, for blockchain data  $Q_{61}$ , the blockchain node searches, based on the second position information of blockchain data  $Q_{61}$ , the hash ring along a clockwise storage direction for first position information having a closest position relationship with the second position information, and determines a database instance (for example, database instance  $D_1$  shown in FIG. 9) corresponding to the found first position information to be the first database instance, thereby storing the blockchain data to the first database instance.

**[0099]** In a similar manner, with reference to the storage method of blockchain data  $Q_{61}$ , the blockchain node stores blockchain data  $Q_{62}$  to database instance  $D_1$ , stores blockchain data  $Q_{63}$  to database instance  $D_2$ , stores blockchain data  $Q_{64}$ , blockchain data  $Q_{65}$ , and blockchain data  $Q_{66}$  to database instance  $D_3$ , and stores blockchain data  $Q_{67}$  to database instance  $D_4$ .

[0100] Therefore, with this storage method provided in the embodiments of the present disclosure, multiple database instances can be deployed on one blockchain node according to a service requirement, thereby significantly increasing the storage capacity. Afterwards, according to the data type of the blockchain data, the blockchain data of different data types can be respectively stored to different database instances in a sharding and classification manner, thereby greatly relieving a storage pressure and achieving a distribution effect. It follows that, with the storage method of the embodiments of the present disclosure, the storage capacity and read-write performance of the blockchain node can be multiplied, thereby effectively improving the node storage capability of the blockchain node and meeting a storage requirement.

10 [0101] Further, reference is made to FIG. 10, which is a schematic flowchart of a data processing method based on a blockchain according to an embodiment of the present disclosure. As shown in FIG. 10, the method is performed by a computer device. The computer device is a blockchain node in a blockchain network. The blockchain node is a blockchain node that needs to store full ledger data in the blockchain network. For example, the blockchain node is any  
15 blockchain node in the blockchain network shown in FIG. 1. For example, the blockchain node is node 10a. The method may include at least step 1001 to step 1006 below.

[0102] In step 1001, the blockchain node obtains a storage instance mapping table based on database configuration information of the blockchain node.

[0103] The storage instance mapping table includes data mapping relationships of N database instances. One data mapping relationship indicates a data type of one database instance. N is a positive integer greater than 1. In an embodiment, a service object (for example, a user) performs a starting operation on the blockchain node. In response to the starting operation, the blockchain node obtains the storage instance mapping table of the blockchain node based on the database configuration information of the blockchain node. The starting operation refers to a  
20 triggering operation used for starting the blockchain node. The triggering operation includes contact operations such as click and long press, or non-contact operations such as voice and gesture. A type of the triggering operation is not limited here.

[0104] In step 1002, the blockchain node determines M candidate database instances from the N database instances in the storage instance mapping table in a case that to-be-stored blockchain data is already obtained. Data types of the candidate database instances are matched with the  
30 blockchain data.

[0105] M is a positive integer less than or equal to N. In an embodiment, in a case that the to-be-stored blockchain data is already obtained, the blockchain node determines data information

of the blockchain data and then determines a data type of the blockchain data based on the data information of the blockchain data. Further, the blockchain node searches the storage instance mapping table for a data type matching with the data type of the blockchain data, and determines database instances corresponding to the found data type to be the candidate database instances, thereby obtaining the M candidate database instances.

**[0106]** In step 1003, the blockchain node stores the blockchain database to a first database instance among the M candidate database instances according to a data sharding strategy in the database configuration information.

**[0107]** In an embodiment, the blockchain node generates, from a hash value space along a storage direction based on the data sharding strategy in the database configuration information, a hash ring associated with the data type of the blockchain data. Further, the blockchain node performs hash computation on first key information (such as an address, a directory, and other inputted information) of each candidate database instance among the M candidate database instances to obtain first position information of each database instance on the hash ring. Similarly, the blockchain node performs hash computation on second key information (such as a data hash value) of the blockchain data to obtain second position information of the blockchain data on the hash ring. Further, the blockchain node searches the hash ring along the storage direction for first position information having a closest position relationship with the second position information, determines a database instance corresponding to the found first position information to be the first database instance, and stores the blockchain data to the first database instance.

**[0108]** For specific implementations of step 1001 to step 1003, one may refer to the descriptions of step 601 to step 603 in the embodiment of FIG. 6 above. Details are not described here.

**[0109]** In an embodiment, the blockchain data includes to-be-processed data.

**[0110]** In step 1004, the blockchain node generates a data access instruction in response to a service data request for the to-be-processed data. The data access instruction is used for accessing the to-be-processed data.

**[0111]** The service data request is generated by the blockchain node based on a data type of the to-be-processed data, or may be transmitted by a computer device (for example, another blockchain node in the blockchain network or a terminal device corresponding to a service object) that has a network connection relationship with the blockchain node. A source of the service data request is not limited here.

**[0112]** If the service data request is transmitted by a computer device that has a network connection relationship with the blockchain node, the computer device obtains a node public key of the blockchain node and encrypts the data information such as the data type of the to-be-processed data to obtain encrypted data information in order to effectively improve the security of data transmission, so that the service data request transmitted to the blockchain node can be generated based on the encrypted data information. When the blockchain node receives the service data request, the blockchain node first verifies the validity of the computer device (for example, determining whether the computer device belongs to an invalid list of computer devices). When it is verified that the computer device is valid, the blockchain node decrypts the encrypted data information based on a node private key of the blockchain node to obtain the data type and other data information of the to-be-processed data, and generates, based on the data information, the data access instruction used for accessing the to-be-processed data.

**[0113]** In step 1005, the blockchain node determines a second database instance from the N database instances in the storage instance mapping table, and accesses the to-be-processed data from the second database instance based on the data access instruction and the data sharding strategy in the database configuration information. A data type of the second database instance is matched with the to-be-processed data.

**[0114]** In an embodiment, with reference to the implementation corresponding to step 602 above, the blockchain node determines the second database instance from the N database instances included in the storage instance mapping table, and then determines, from the second database instance by using the storage method shown in FIG. 9 above, a service database instance to which the to-be-processed data belongs based on the data sharding strategy in the database configuration information, so as to access the to-be-processed data from the service database instance.

**[0115]** In an embodiment, the N database instances are all deployed in a remote database associated with the blockchain node. The remote database includes a host cluster corresponding to X data types. X is a positive integer. One data type corresponds to one host cluster. When the blockchain node accesses the to-be-processed data, the blockchain node determines a target host cluster from the X host clusters. A data type of the target host cluster is matched with the to-be-processed data. In an embodiment, the service data request carries the data type of the to-be-processed data. The data type of the target host cluster being matched with the to-be-processed data includes that: the data type of the target host cluster is matched with a data type of the to-be-processed data. The blockchain node determines a database instance deployed in the target host cluster to be the second database instance. Further, the blockchain node determines,

from the second database instance based on the data sharding strategy in the database configuration information, the service database instance to which the to-be-processed data belongs, forwards the data access instruction to a database agent having a network connection relationship with the blockchain node, so that the database agent transmits the data access instruction to a database host node deployed by the service database instance. In this case, the blockchain node receives the to-be-processed data returned by the database host node through the database agent.

**[0116]** For example, if the blockchain node performs storage extension with the method shown in FIG. 4 above, the blockchain node can determine host cluster 1 from the three host clusters when accessing the to-be-processed data. A data type of host cluster 1 is matched with the data type of the to-be-processed data, and then a database instance deployed in host cluster 1 is determined to be the second database instance. Further, the blockchain node determines, from the second database instance by using the method shown in FIG. 9 above, the service database instance (for example, the hard disk A shown in FIG. 4) to which the to-be-processed data belongs based on the data sharding strategy in the database configuration information.

**[0117]** In this way, the blockchain node calls a database service component. The database service component may be deployed on the same device as the blockchain node or deployed separately on another device different from the blockchain node. The deployment of the database service component is not limited here. If the database service component is deployed separately on another device different from the blockchain node, this device is referred to as a database agent having a network connection relationship with the blockchain node in the embodiments of the present disclosure. The blockchain node forwards the data access instruction to the database agent shown in FIG. 4 to cause the database agent to transmit data access instruction to the database host node (for example, the database host node where host J<sub>11</sub> is located) deployed by the service database instance. In this case, the blockchain node can receive the to-be-processed data returned by the database host node through the database agent.

**[0118]** For ease of understanding, reference is made to FIG. 11, which is a schematic flowchart of accessing to-be-processed data according to an embodiment of the present disclosure. As shown in FIG. 11, a blockchain node in the embodiments of the present disclosure is a blockchain node that needs to store full ledger data in a blockchain network. For example, the blockchain node is any blockchain node in the blockchain network shown in FIG. 1. For example, the blockchain node is node 10a.

**[0119]** When the blockchain node obtains a service data request, the blockchain node executes step 1101 to obtain a storage instance mapping table, and then executes step 1102 to

determine a second database instance based on a data type of to-be-processed data carried by the service data request.

[0120] As shown in FIG. 8 above, in a case that the data type of the to-be-processed data obtained by the blockchain node is the contract data type, the blockchain node obtains the storage instance mapping table. Among the N database instances in the storage instance mapping table, the  
5 second database instance is determined. The data type of the second database instance is matched with the to-be-processed data. For example, the second database instance is database instance D<sub>4</sub>, database instance D<sub>5</sub>, and database instance D<sub>6</sub> shown in FIG. 8.

[0121] The blockchain node executes step 1103 to step 1104 to determine position  
10 information of the to-be-processed data on a hash ring based on a data sharding strategy, finds, along a storage direction of the hash ring, first position information having closest position relationship with the position information of the to-be-processed data, and determines a database instance corresponding to the found first position information to be a service database instance.

[0122] In this case, the blockchain node executes step 1105 to determine whether the service  
15 database instance is deployed in a local database. In a case that the service database instance is deployed in a local database, the blockchain node executes step 1106 to access the to-be-processed data in the local database, and executes step 1107 to end the access flow. For example, if the service database instance is database instance D<sub>3</sub> shown in FIG. 3 above, the blockchain node accesses the to-be-processed data in the hard disk A in the local database.

20 [0123] In an embodiment, in a case that the service database instance is not deployed in the local database but deployed in a remote database, the blockchain node jumps to execute step 1108 to step 1111 to: call a database service component to cause the database service component to establish a connection between the blockchain node and the remote database, then access the to-be-processed data in the remote database, and return the to-be-processed data to the database  
25 service component, so as to cause the database service component to return the to-be-processed data to the blockchain node. The blockchain node then executes step 1107 to end the access flow.

[0124] In step 1006, the blockchain node performs a data operation indicated by the service data request on the to-be-processed data.

[0125] In an embodiment, the service data request includes a data archiving request, a data  
30 change request, a data addition request, a data query request, a data deletion request, and the like. The data archiving request is used for archiving the to-be-processed data when the data type of the to-be-processed data indicates that the to-be-processed data is transaction data. In an embodiment,

when the service data request is a data archiving request, the blockchain node obtains data information of the to-be-processed data and a data archiving condition for the to-be-processed data. The data archiving condition includes an archiving parameter threshold. The archiving parameter threshold includes at least one of an archiving time range (for example, a time range N years from now) or an archiving frequency threshold (for example, 50 times). The blockchain node determines that the to-be-processed data satisfies the data archiving condition, in a case that the data information indicates that a transaction time stamp of the to-be-processed data belongs to the archiving time range or an access frequency of the to-be-processed data is less than the archiving frequency threshold,. In this case, the blockchain node transfers the to-be-processed data to an archiving database from a database to which the to-be-processed data belongs.

**[0126]** It can be seen that the embodiments of the present disclosure are applicable to scenarios that require a blockchain system to store massive data and have high requirements for chain performance, including a scenario of performance pressure caused by a high transaction frequency and a scenario of storage space pressure caused by a large transaction quantity. In combination with the data characteristics of a blockchain, different types of blockchain data are stored on different database instances respectively to solve the bottleneck problem of storage capacity. For example, according to the embodiments of the present disclosure, inactive data such as historical transactions is stored to an archiving database. The archiving database is a device with poor performance and low costs, which is conducive to processing data such as archiving or regularly deleting data, thereby saving a storage space and the cost. This means that the main database of the blockchain in the embodiments of the present disclosure only needs to retain the latest data, thereby reducing the data magnitude of the database and improving the data read-write rate, and thus improving the data access performance. This storage extension method not only solves the problem of inferior data read-write performance of the blockchain node, but also solves the problem of a limited storage space of a single machine. Even through refined contract storage management according to the data types and the data sharding strategy, implementation of the blockchain in a storage scenario with a large data volume becomes possible.

**[0127]** Reference is made to FIG. 12, which is a schematic structural diagram of a data processing apparatus based on a blockchain according to an embodiment of the present disclosure. As shown in FIG. 12, the data processing apparatus 1 based on the blockchain is a computer program (including program codes) run on a computer device. For example, the data processing apparatus 1 based on the blockchain is application software. The data processing apparatus based on the blockchain is configured to execute the corresponding steps in the method provided in the embodiments of the present disclosure. As shown in FIG. 12, the data processing apparatus based

on the blockchain is run on a blockchain node that needs to store full ledger data in a blockchain network. For example, the blockchain node is any blockchain node in the blockchain network shown in FIG. 1. For example, the blockchain node is node 10a. The data processing apparatus based on the blockchain includes: a mapping table obtaining module 121, a data type matching module 122, a data storage module 123, an access instruction generation module 124, a data access module 125, and an operation execution module 126.

**[0128]** The mapping table obtaining module 121 is configured to obtain a storage instance mapping table based on database configuration information of the blockchain node. The storage instance mapping table includes data mapping relationships of N database instances, and the data mapping relationships indicating data types of the database instances, and N is a positive integer greater than 1.

**[0129]** In an embodiment, the mapping table obtaining module 121 includes: a configuration information loading unit 1211, an instance obtaining unit 1212, and a mapping table generation unit 1213.

**[0130]** The configuration information loading unit 1211 is configured to: in response to a starting operation performed on the blockchain node, load the database configuration information in a case that the starting operation indicates that the blockchain node is started for the first time.

**[0131]** The instance obtaining unit 1212 is configured to establish the N database instances based on the database configuration information and a database type of a main database of the blockchain node.

**[0132]** In an embodiment, storage path information in the database configuration information indicates a hard disk set associated with X data types, and X is a positive integer.

**[0133]** The instance obtaining unit 1212 includes: a database type obtaining subunit 12121, a smart contract obtaining subunit 12122, a first establishment subunit 12123, a prompt information display subunit 12124, and a second establishment subunit 12125.

**[0134]** The database type obtaining subunit 12121 is configured to obtain the database type of the main database of the blockchain node.

**[0135]** The smart contract obtaining subunit 12122 is configured to obtain, in a case that the database type of the blockchain node is the first type, the smart contract from the blockchain to which the blockchain node belongs. The smart contract is used for establishing database instances.

**[0136]** The first establishment subunit 12123 is configured to establish a database instance for each hard disk in the hard disk set based on the smart contract to obtain the N database instances.

**[0137]** The first establishment subunit 12123 is further configured to:

5 **[0138]** determine, for each hard disk in the hard disk set, an instance quantity of the hard disk based on the database configuration information;

**[0139]** establish, based on the smart contract in a case that the hard disk is deployed in a local database in the main database, the database instance corresponding to the instance quantity for the hard disk; and

10 **[0140]** call a database service component in a case that the hard disk is deployed in a remote database in the main database, and establish, based on the smart contract, the database instance corresponding to the instance quantity for the hard disk, the database service component being configured to provide a remote database service.

15 **[0141]** The prompt information display subunit 12124 is configured to display, in a case that the database type is a second type, prompt information associated with the database configuration information.

**[0142]** The second establishment subunit 12125 is configured to establish a database instance for each hard disk in the hard disk set in response to a triggering operation performed on the prompt information, to obtain the N database instances.

20 **[0143]** For implementations of the database type obtaining subunit 12121, the smart contract obtaining subunit 12122, the first establishment subunit 12123, the prompt information display subunit 12124, and the second establishment subunit 12125, one may refer to the descriptions of establishing the database instances in the embodiment of FIG. 6 above. Details are not described here.

25 **[0144]** The X data types include at least one of a first data type, a second data type, or a third data type. The first data type includes a block header data type. The second data type includes a transaction data type. The third data type includes a contract data type. Storage performance of a hard disk of the first data type is inferior to storage performance of a hard disk of the second data type; and the storage performance of the hard disk of the first data type is inferior to storage  
30 performance of a hard disk of the third data type.

**[0145]** The mapping table generation unit 1213 is configured to: create a mapping relationship between each database instance and a data type of a hard disk to which the database

instance belongs, and generate the storage instance mapping table based on the mapping relationship.

[0146] For implementations of the configuration information loading unit 1211, the instance obtaining unit 1212, and the mapping table generation unit 1213, one may refer to the description of  
5 step 601 in the embodiment of FIG. 6 above. Details are not described here.

[0147] The data type matching module 122 is configured to determine M candidate database instances from the N database instances in a case that to-be-stored blockchain data is already obtained. Data types of the candidate database instances are matched with the blockchain data, and M is a positive integer less than or equal to N.

10 [0148] The data type matching module 122 includes: a data information determining unit 1221, a data type determining unit 1222, and a data type matching unit 1223.

[0149] The data information determining unit 1221 is configured to determine data information of the blockchain data in a case that the to-be-stored blockchain data is already obtained.

15 [0150] The data type determining unit 1222 is configured to determine a data type of the blockchain data based on the data information of the blockchain data.

[0151] The database configuration information includes a data volume threshold and an access frequency threshold.

[0152] The data type determining unit 1222 includes: a first determining subunit 12221, a  
20 second determining subunit 12222, and a third determining subunit 12223.

[0153] The first determining subunit 12221 is configured to determine the first data type to be the data type of the blockchain data in a case that the data information indicates that a data volume of the blockchain data does not reach the data volume threshold and a data access frequency of the blockchain data reaches the access frequency threshold.

25 [0154] The second determining subunit 12222 is configured to determine the second data type to be the data type of the blockchain data in a case that the data information indicates that the blockchain data is transaction data.

[0155] The third determining subunit 12223 is configured to determine the third data type to be the data type of the blockchain data in a case that the data information indicates that the  
30 blockchain data is contract data.

**[0156]** For implementations of the first determining subunit 1221, the second determining subunit 1222, and the third determining subunit 1223, one may refer to the description of the data type in the embodiment of FIG. 6 above. Details are not described here.

**[0157]** The data type matching unit 1223 is configured to: search the storage instance mapping table for a data type matching with the data type of the blockchain data, and determine database instances corresponding to the found data type to be the candidate database instances.

**[0158]** For implementations of the data information determining unit 1221, the data type determining unit 1222, and the data type matching unit 1223, one may refer to the description of step 601 in the embodiment corresponding to FIG. 6 above. Details are not described here.

**[0159]** The data storage module 123 is configured to store the blockchain database to a first database instance among the M candidate database instances according to a data sharding strategy in the database configuration information.

**[0160]** The data storage module 123 includes: a hash ring generation unit 1231, a first hash computation unit 1232, a second hash computation unit 1233, a target instance determining unit 1234, and a data storage unit 1235.

**[0161]** The hash ring generation unit 1231 is configured to generate, from a hash value space along a storage direction based on the data sharding strategy, a hash ring associated with the data type of the blockchain data.

**[0162]** The first hash computation unit 1232 is configured to perform hash computation on first key information of each of the M candidate database instances to obtain first position information of each candidate database instance on the hash ring.

**[0163]** The second hash computation unit 1233 is configured to perform hash computation on second key information of the blockchain data to obtain second position information of the blockchain data on the hash ring.

**[0164]** The target instance determining unit 1234 is configured to: search, based on the second position information, the hash ring along the storage direction for first position information having a closest position relationship with the second position information, and determine a database instance corresponding to the found first position information to be the first database instance.

**[0165]** The data storage unit 1235 is configured to store the blockchain data to the first database instance.

[0166] For implementations of the hash ring generation unit 1231, the first hash computation unit 1232, the second hash computation unit 1233, the target instance determining unit 1234, and the data storage unit 1235, one may refer to the description of step 603 in the embodiment of FIG. 6 above. Details are not described here.

5 [0167] The blockchain data includes to-be-processed data.

[0168] The access instruction generation module 124 is configured to generate a data access instruction in response to a service data request for the to-be-processed data. The data access instruction is used for accessing the to-be-processed data.

10 [0169] The data access module 125 is configured to: determine a second database instance from the N database instances, and access the to-be-processed data from the second database instance based on the data access instruction and the data sharding strategy. A data type of the second database instance is matched with the to-be-processed data.

15 [0170] The N database instances are deployed in a remote database associated with the blockchain node. The remote database includes a host cluster corresponding to X data types. X is a positive integer. One data type corresponds to one host cluster.

[0171] The data access module 125 includes: a to-be-searched instance determining unit 1251, a service instance determining unit 1252, an access instruction forwarding unit 1253, and a data receiving unit 1254.

20 [0172] The to-be-searched instance determining unit 1251 is configured to: determine a target host cluster from the X host clusters, and determine a database instance deployed in the target host cluster to be the second database instance. A data type of the target host cluster is matched with the to-be-processed data.

25 [0173] The service instance determining unit 1252 is configured to determine, from the second database instance based on the data sharding strategy, a service database instance to which the to-be-processed data belongs.

[0174] The access instruction forwarding unit 1253 is configured to forward the data access instruction to a database agent having a network connection relationship with the blockchain node to cause the database agent to transmit the data access instruction to a database host node deployed by the service database instance.

30 [0175] The data receiving unit 1254 is configured to receive the to-be-processed data returned by the database host node through the database agent.

**[0176]** For implementations of the to-be-searched instance determining unit 1251, the service instance determining unit 1252, the access instruction forwarding unit 1253, and the data receiving unit 1254, one may refer to the description of step 1005 in the embodiment corresponding to FIG. 10 above. Details are not described here.

5 **[0177]** The operation execution module 126 is configured to perform a data operation indicated by the service data request on the to-be-processed data.

**[0178]** The service data request includes a data archiving request. The data archiving request is used for archiving the to-be-processed data when the data type of the to-be-processed data indicates that the to-be-processed data is transaction data.

10 **[0179]** The operation execution module 126 includes: an archiving condition obtaining unit 1261 and a data transferring unit 1262.

**[0180]** The archiving condition obtaining unit 1261 is configured to obtain data information of the to-be-processed data and a data archiving condition for the to-be-processed data. The data archiving condition includes an archiving parameter threshold. The archiving parameter threshold  
15 includes at least one of an archiving time range or an archiving frequency threshold.

**[0181]** The data transferring unit 1262 is configured to: determine that the to-be-processed data satisfies the data archiving condition, in a case that the data information indicates that a transaction time stamp of the to-be-processed data belongs to the archiving time range or an accessing frequency of the to-be-processed data is less than the archiving frequency threshold, and  
20 transfer the to-be-processed data to an archiving database from a database to which the to-be-processed data belongs.

**[0182]** For implementations of the archiving condition obtaining unit 1261 and the data transferring unit 1262, one may refer to the description of step 1006 in the embodiment corresponding to FIG. 10 above. Details are not described here.

25 **[0183]** For implementations of the mapping table obtaining module 121, the data type matching module 122, the data storage module 123, the access instruction generation module 124, the data access module 125, and the operation execution module 126, one may refer to the descriptions of step 1001 to step 1006 in the embodiment of FIG. 10 above. Details are not described here. In addition, the beneficial effects of the same method are not described herein again.

30 **[0184]** Reference is made to FIG. 13, which is a schematic diagram of a computer device according to an embodiment of the present disclosure. As shown in FIG. 13, the computer device 1300 may be a blockchain node that needs to store full ledger data in a blockchain network. For

example, the blockchain node is any blockchain node in the blockchain network shown in FIG. 1. For example, the blockchain node is node 10a. The computer device 1300 may include: at least one processor 1301, for example, a central processing unit (CPU), at least one network interface 1304, a user interface 1303, a memory 1305, and at least one communications bus 1302. The communications bus 1302 is configured to implement connection and communication between these components. The user interface 1303 may include a display and a keyboard. The network interface 1304 may optionally include a standard wired interface and wireless interface (such as a WI-FI interface). The memory 1305 may be a high-speed RAM memory, or may be a non-volatile memory, for example, at least one magnetic disk memory. Optionally, the memory 1305 may be at least one storage apparatus located remotely from the foregoing processor 1301. As shown in FIG. 13, the memory 1305 serving as a computer storage medium may include an operating system, a network communications module, a user interface module, and a device-control application program.

**[0185]** In the computer device 1300 shown in FIG. 13, the network interface 1304 is mainly configured to perform network communication. The user interface 1303 is mainly configured to provide an input interface for a user. The processor 1301 may be configured to invoke a device control application program stored in the memory 1305 to perform the following operations:

**[0186]** obtaining a storage instance mapping table based on database configuration information of the blockchain node, the storage instance mapping table including data mapping relationships of N database instances, the data mapping relationships indicating data types of the database instances, and N being a positive integer greater than 1;

**[0187]** determining M candidate database instances from the N database instances in a case that to-be-stored blockchain data is already obtained, data types of the candidate database instances being matched with the blockchain data, and M being a positive integer less than or equal to N; and

**[0188]** storing the blockchain database to a first database instance among the M candidate database instances according to a data sharding strategy in the database configuration information.

**[0189]** It is understood that the computer device 1300 described in the embodiments of the present disclosure can execute the foregoing data processing method based on the blockchain in the embodiments corresponding FIG. 6 and FIG. 13, as well as operations of the foregoing data processing apparatus based on the blockchain in the embodiment corresponding to FIG. 12. Details are not described here. In addition, the beneficial effects of the same method are not described herein again.

[0190] A computer-readable storage medium is further provided according to an embodiment of the present disclosure. The computer-readable storage medium stores a computer program, and the computer program includes program instructions. The program instructions, when executed by a processor, implement the data processing method based on the blockchain described in FIG. 6 and FIG. 10. One may refer to the implementations provided by all the steps of FIG. 6 and FIG. 10. Details are not described here.

[0191] The computer-readable storage medium may be an internal storage unit of the data processing apparatus or computer device provided by any of the aforementioned embodiments, for example, a hard disk or an internal memory of the computer device. The computer-readable storage medium may be an external storage device of the computer device, such as a plug-in hard disk, smart media card (SMC), a secure digital (SD) card, and a flash card, arranged on the computer device. Further, the computer-readable storage medium may include both the internal storage unit of the computer device and the external storage device. The computer-readable storage medium may be configured to store the computer program and other programs and data required by the computer device. The computer-readable storage medium may be configured to temporarily store data that has been or will be outputted.

[0192] A computer program product is further provided according to an embodiment of the present disclosure. The computer program product includes a computer program. The computer program is stored in a computer-readable storage medium. A processor of a computer device reads the computer program from the computer-readable storage medium and executes the computer program, causing the computer device to perform the foregoing data processing method based on the blockchain in the various embodiments. Details are not described here. In addition, the beneficial effects of the same method are not described herein again.

[0193] Optional embodiments of the present disclosure are described above, and are not intended to limit the scope of the claims of the present disclosure. Therefore, equivalent variations made in accordance with the claims of the present disclosure shall fall within the scope of the present disclosure.

## CLAIMS

### What is claimed is:

1. A data processing method based on a blockchain, performed by a blockchain node, the method comprising:

5 obtaining a storage instance mapping table based on database configuration information of the blockchain node, the storage instance mapping table comprising data mapping relationships of N database instances, the data mapping relationships indicating data types of the database instances, and N being a positive integer greater than 1;

10 determining M candidate database instances from the N database instances in a case that to-be-stored blockchain data is already obtained, data types of the candidate database instances being matched with the blockchain data, and M being a positive integer less than or equal to N; and

storing the blockchain database to a first database instance among the M candidate database instances according to a data sharding strategy in the database configuration information.

15 2. The method according to claim 1, wherein the obtaining a storage instance mapping table based on database configuration information of the blockchain node comprises:

in response to a starting operation performed on the blockchain node, loading the database configuration information of the blockchain node in a case that the starting operation indicates that the blockchain node is started for the first time;

20 establishing the N database instances based on the database configuration information and a database type of a main database of the blockchain node;

creating a mapping relationship between each database instance and a data type of a hard disk to which the database instance belongs; and

generating the storage instance mapping table based on the mapping relationship.

25 3. The method according to claim 2, wherein the database configuration information comprises storage path information; the storage path information indicates a hard disk set associated with X data types; X is a positive integer;

wherein the establishing the N database instances based on the database configuration information and a database type of a main database of the blockchain node comprises:

obtaining the database type of the main database of the blockchain node;

30 obtaining, in a case that the database type is a first type, a smart contract from a blockchain to

which the blockchain node belongs, the smart contract being used for establishing database instances; and

establishing a database instance for each hard disk in the hard disk set based on the smart contract to obtain the N database instances.

5        4. The method according to claim 3, wherein the X data types comprise at least one of a first data type, a second data type, or a third data type; the first data type comprises a block header data type; the second data type comprises a transaction data type; the third data type comprises a contract data type; storage performance of a hard disk of the first data type is inferior to storage performance of a hard disk of the second data type; and the storage performance of the hard disk of the first data  
10       type is inferior to storage performance of a hard disk of the third data type.

5. The method according to claim 3, wherein the establishing a database instance for each hard disk in the hard disk set based on the smart contract comprises:

determining, for each hard disk in the hard disk set, an instance quantity of the hard disk based on the database configuration information;

15       establishing, based on the smart contract in a case that the hard disk is deployed in a local database in the main database, the database instance corresponding to the instance quantity for the hard disk; and

calling a database service component in a case that the hard disk is deployed in a remote database in the main database, and establishing, based on the smart contract, the database instance  
20       corresponding to the instance quantity for the hard disk, the database service component being configured to provide a remote database service.

6. The method according to claim 3, further comprising:

displaying, in a case that the database type is a second type, prompt information associated with the database configuration information; and

25       establishing a database instance for each hard disk in the hard disk set in response to a triggering operation performed on the prompt information, to obtain the N database instances.

7. The method according to claim 1, wherein the determining M candidate database instances from the N database instances in a case that to-be-stored blockchain data is already obtained comprises:

30       determining data information of the blockchain data in a case that the blockchain data is already obtained;

determining a data type of the blockchain data based on the data information of the blockchain data; and

searching the storage instance mapping table for a data type matching with the data type of the blockchain data, and determining database instances corresponding to the found data type to be the candidate database instances.

8. The method according to claim 7, wherein the database configuration information comprises a data volume threshold and an access frequency threshold;

wherein the determining a data type of the blockchain data based on the data information of the blockchain data comprises:

10 determining a first data type to be the data type of the blockchain data in a case that the data information indicates that a data volume of the blockchain data does not reach the data volume threshold and a data access frequency of the blockchain data reaches the access frequency threshold;

determining a second data type to be the data type of the blockchain data in a case that the data information indicates that the blockchain data is transaction data; and

determining a third data type to be the data type of the blockchain data in a case that the data information indicates that the blockchain data is contract data.

9. The method according to claim 1, wherein the storing the blockchain database to a first database instance among the M candidate database instances according to a data sharding strategy in the database configuration information comprises:

generating, from a hash value space along a storage direction according to the data sharding strategy, a hash ring associated with the data type of the blockchain data;

performing hash computation on first key information of each candidate database instance to obtain first position information of each candidate database instance on the hash ring;

25 performing hash computation on second key information of the blockchain data to obtain second position information of the blockchain data on the hash ring;

searching the hash ring along the storage direction for first position information having a closest position relationship with the second position information, and determining a database instance corresponding to the found first position information to be the first database instance; and

30 storing the blockchain data to the first database instance.

10. The method according to claim 1, wherein the blockchain data comprises to-be-processed data, and the method further comprises:

generating a data access instruction in response to a service data request for the to-be-processed data, the data access instruction being used for accessing the to-be-processed data;

5 determining a second database instance from the N database instances, and accessing the to-be-processed data from the second database instance based on the data access instruction and the data sharding strategy, a data type of the second database instance being matched with the to-be-processed data; and

performing a data operation indicated by the service data request on the to-be-processed data.

10 11. The method according to claim 10, wherein the N database instances are deployed in a remote database associated with the blockchain node; the remote database comprises host clusters corresponding to X data types; X is a positive integer; one data type corresponds to one of the host clusters;

15 wherein the determining a second database instance from the N database instances, and accessing the to-be-processed data from the second database instance based on the data access instruction and the data sharding strategy comprises:

determining a target host cluster from X host clusters, and determining a database instance deployed in the target host cluster to be the second database instance, a data type of the target host cluster being matched with the to-be-processed data;

20 determining, from the second database instance based on the data sharding strategy, a service database instance to which the to-be-processed data belongs;

forwarding the data access instruction to a database agent having a network connection relationship with the blockchain node to cause the database agent to transmit the data access instruction to a database host node deployed by the service database instance; and

25 receiving the to-be-processed data returned by the database host node through the database agent.

12. The method according to claim 10, wherein the service data request comprises a data archiving request; the data archiving request is used for archiving the to-be-processed data in a case that the to-be-processed data is transaction data;

30 wherein the performing a data operation indicated by the service data request on the to-be-processed data comprises:

obtaining data information of the to-be-processed data and a data archiving condition of the to-be-processed data, the data archiving condition comprising an archiving parameter threshold, and the archiving parameter threshold comprising at least one of an archiving time range or an archiving frequency threshold; and

5 determining, in a case that the data information indicates that a transaction time stamp of the to-be-processed data belongs to the archiving time range or an accessing frequency of the to-be-processed data is less than the archiving frequency threshold, that the to-be-processed data satisfies the data archiving condition, and transferring the to-be-processed data to an archiving database from a database to which the to-be-processed data belongs.

10 13. A data processing apparatus based on a blockchain, configured in a blockchain node. the apparatus comprising:

a mapping table obtaining module, configured to obtain a storage instance mapping table based on database configuration information of the blockchain node, the storage instance mapping table comprising data mapping relationships of N database instances, the data mapping relationships  
15 indicating data types of the database instances, and N being a positive integer greater than 1;

a data type matching module, configured to determine M candidate database instances from the N database instances in a case that to-be-stored blockchain data is already obtained, data types of the candidate database instances being matched with the blockchain data, and M being a positive integer less than or equal to N; and

20 a data storage module, configured to store the blockchain database to a first database instance among the M candidate database instances according to a data sharding strategy in the database configuration information.

14. A computer device, comprising: a processor, a memory, and a network interface;

the processor being connected to the memory and the network interface, the network interface  
25 being configured to provide a data communication function, the memory being configured to store a computer program, and the processor being configured to call the computer program to cause the computer device to perform the method according to any one of claims 1 to 12.

15. A computer-readable storage medium storing a computer program, the computer program being loaded and executed by a processor to cause a computer device with the processor to perform  
30 the method according to any one of claims 1 to 12.

16. A computer program product comprising a computer program, the computer program being stored in a computer-readable storage medium, and the computer program being loaded and

executed by a processor to cause a computer device with the processor to perform the method according to any one of claims 1 to 12.