

DATA SYNCHRONIZATION METHOD, APPARATUS, AND SYSTEM

TECHNICAL FIELD

[0001] The present application relates to the field of database technologies, and in particular, to a data synchronization method, apparatus, and system.

BACKGROUND

[0002] In the big data era, data owners usually build more than two data centers for the purpose of disaster recovery. In a conventional cold standby disaster recovery solution, an active data center is used as a primary service, and other data centers are used to back up data in the active data center. A problem with this solution is that the active data center and a standby center cannot be directly reused, and the standby data center can function only in a disaster, resulting in low resource utilization. In addition, it is time-consuming and complicated for the standby data center to take over the active data center as the primary service, and it usually severely affects processing of a normal service.

[0003] A distributed multi-active data center (multi-active for short) technology is proposed for the problem with the cold standby technology. An implementation idea is as follows: there are no active and standby services in a plurality of data centers, the data centers cooperate with each other in normal mode, and provide a service for service access in parallel. This prevents standby data centers from being idle and doubles the service capability of the system. When one data center fails, other data centers can quickly take over all services.

[0004] In the multi-active solution, because data centers each have relatively equal footing, a requirement on speed of data synchronization between data centers is relatively high. Especially in an application scenario where data centers are deployed at different locations, improving the speed of data synchronization between two locations at a relatively long communication distance is a problem in current multi-active data center construction.

SUMMARY

[0005] To resolve the previous technical problem, the present application provides a data synchronization method, apparatus, and system. The technical solutions are as follows:

5 **[0006]** According to a first aspect of the present application, a data synchronization method is provided. The method includes the following: after determining that data of a source end database is modified, generating a real-time notification for this data modification by a data synchronization source end and sending the real-time notification to a data synchronization target end, where the
10 real-time notification includes information related to this data modification; and after receiving the real-time notification, obtaining the data modification related information by parsing the real-time notification by the data synchronization target end and updating a cache in a target end database based on a parsing result.

[0007] According to a second aspect of the present application, a data
15 synchronization method is provided. The method is applied to a data synchronization source end and includes the following: generating a real-time notification for this data modification after determining that data of a source end database is modified, where the real-time notification includes information related to this data modification; and sending the real-time notification to a data synchronization target end. Thus, after
20 receiving the real-time notification, the data synchronization target end obtains the data modification related information by parsing the real-time notification and updates a cache in a target end database based on a parsing result.

[0008] According to a third aspect of the present application, a data
25 synchronization method is provided. The method is applied to a data synchronization target end and includes the following: receiving a real-time notification sent by a data synchronization source end; obtaining data modification related information by parsing the real-time notification; and updating a cache in a target end database based on a parsing result. After the data synchronization source end determines that data of a source end database is modified, the real-time notification is generated and sent for
30 this data modification, and the real-time notification includes information related to this data modification.

[0009] According to a fourth aspect of the present application, a data synchronization source end apparatus is provided. The apparatus includes a real-time

notification generation module and a real-time notification sending module, where the real-time notification generation module is configured to generate a real-time notification for this data modification after it is determined that data of a source end database is modified, where the real-time notification includes information related to this data modification. Moreover, the real-time notification sending module is configured to send the real-time notification to a data synchronization target end. Thus, after receiving the real-time notification, the data synchronization target end obtains the data modification related information by parsing the real-time notification and updates a cache in a target end database based on a parsing result.

10 **[0010]** According to a fifth aspect of the present application, a data synchronization target end apparatus is provided. The apparatus includes a real-time notification receiving module and a cache data update module, where the real-time notification receiving module is configured to receive a real-time notification sent by a data synchronization source end. The cache data update module is configured to obtain data modification related information by parsing the real-time notification and update a cache in a target end database based on a parsing result. After the data synchronization source end determines that data of a source end database is modified, the real-time notification is generated and sent for this data modification, and the real-time notification includes information related to this data modification.

20 **[0011]** According to a sixth aspect of the present application, a data synchronization system is provided. The system includes a data synchronization source end apparatus and a data synchronization target end apparatus, where the data synchronization source end apparatus includes a real-time notification generation module and a real-time notification sending module. The real-time notification generation module is configured to generate a real-time notification for this data modification after it is determined that data of a source end database is modified, where the real-time notification includes information related to this data modification. The real-time notification sending module is configured to send the real-time notification to a data synchronization target end. The data synchronization target end apparatus includes a real-time notification receiving module and a cache data update module, where the real-time notification receiving module is configured to receive the real-time notification. The cache data update module is configured to obtain data modification related information by parsing the real-time notification and update a cache in a target end database based on a parsing result.

[0012] Based on the technical solutions provided in the present application, after modifying data, the data synchronization source end sends a real-time notification to the data synchronization target end. If the target end can directly update a cache in a local database based on information included in the real-time notification, the cache
5 can be updated to reduce a cache synchronization update delay without waiting for completion of synchronization of an underlying database. If the target end cannot directly update a cache in a local database based on information included in the real-time notification, it can also be learned that the local database is to be synchronized and updated to reduce a cache synchronization update delay, so
10 monitoring the local database can be further initiated to update cache data immediately after synchronization of local data is completed.

[0013] It should be understood that the previous general description and the following detailed description are merely examples and explanations and cannot limit the present application.

BRIEF DESCRIPTION OF DRAWINGS

[0014] To describe the technical solutions in the implementations of the present application or in the existing technology more clearly, the following briefly describes the accompanying drawings for describing the implementations or the existing technology. Apparently, the accompanying drawings in the following description
20 merely show some implementations of the present application, and a person of ordinary skill in the art can still derive other drawings from these accompanying drawings.

[0015] FIG. 1 is a first schematic structural diagram illustrating an active-active system in the existing technology;

[0016] FIG. 2 is a second schematic structural diagram illustrating an active-active system in the existing technology;

[0017] FIG. 3 is a first schematic structural diagram illustrating an active-active system according to the present application;

[0018] FIG. 4 is a second schematic structural diagram illustrating an active-active system according to the present application;

[0019] FIG. 5 is a first schematic flowchart illustrating a data synchronization method according to the present application;

[0020] FIG. 6 is a second schematic flowchart illustrating a data synchronization method according to the present application;

[0021] FIG. 7 is a schematic structural diagram illustrating a data synchronization source end apparatus according to the present application;

5 [0022] FIG. 8 is a schematic structural diagram illustrating a data synchronization target end apparatus according to the present application;

[0023] FIG. 9 is a schematic structural diagram illustrating a data synchronization system according to the present application; and

[0024] FIG. 10 is a schematic structural diagram illustrating a device for
10 configuring an apparatus according to the present application.

DESCRIPTION OF IMPLEMENTATIONS

[0025] To enable a person skilled in the art to better understand the technical solutions in the present application, the following describes in detail the technical solutions in the implementations of the present application with reference to the
15 accompanying drawings in the implementations of the present application. Apparently, the described implementations are merely some rather than all of the implementations of the present application. Other implementations obtained by a person of ordinary skill in the art based on the implementations of the present application shall fall within the protection scope of the present application.

20 [0026] First, a system architecture and working principle of a multi-active data center solution are briefly described. FIG. 1 is a schematic architectural diagram illustrating an active-active system. Two data centers are respectively installed in equipment room A and equipment room B. Database A and database B are respectively located in equipment room A and equipment room B. In each equipment
25 room, an application server directly interacts with a database. The application server performs a read/write operation on a local database based on instructions sent by a user or other systems. Based on requirements of a multi-active system, if a database of an equipment room is modified, the modification needs to be synchronized to another equipment room as soon as possible. In a synchronization process, a database that is
30 first modified is referred to as a source end database, and a database that is subsequently synchronized is referred to as a target end database.

[0027] The active-active system shown in FIG. 1 is used as an example. Based on

a primitive data synchronization solution of an underlying database, if an application server A performs a write operation on database A, write events occurring on database A are sequentially packed. Then the packed events are input to database B in batches to perform data modification. After the modification is completed, an application server B can read data consistent with that in database A from database B.

[0028] Caching is a commonly used technology in database operations. Compared with a basic underlying database storage area, a cache has better data read/write performance and can avoid direct interaction between an application and an underlying database. For example, a relatively apparent feature of state-type data is that the data is read frequently and written infrequently. User information data is used as an example. After filling in personal information in a registration phase, a user is unlikely to frequently modify personal information. Based on such a feature, most systems use a layer of caches at an upper layer of the database to buffer a large number of data query requests, thereby reducing query requests actually sent to the databases and improving overall query performance of the systems.

[0029] A caching technology can also be applied to a multi-active system. As shown in FIG. 2, a layer of caches are disposed on an upper layer of a database, and commonly used data is put into a cache in advance. When an application server needs to use data, the application server needs to preferentially read data from the cache, thereby improving data read efficiency, avoiding direct interaction between the application server and an underlying database, and reducing a read/write burden on the underlying database.

[0030] After a caching mechanism is added, a cache becomes a primary source of data read by the application server. Therefore, when data synchronization is performed between data centers, a synchronization update problem of a cache in a target end equipment room needs to be considered so as to ensure accuracy of data query at a target end. The active-active system shown in FIG. 2 is used as an example. Based on an existing synchronization solution, if application server A performs a write operation on database A, both cache A and cache B need to be updated. Cache A is locally updated, and the existing technical solution can be directly used. No further description is provided. During the update of cache B, a data synchronization operation between database A and database B is first completed. Then application server B updates the local cache B based on the synchronized database B. In addition, in the active-active system, two databases can be deployed in equipment room A. One

database is used as a write database, and the other database is used as a read database. A read database is deployed in another equipment room such as equipment room B. As such, data write operations are performed by using the write database in equipment room A. After data is written to the write database, the newly written data can be
5 synchronized to the read database in the local equipment room and the read database in the other equipment room. As such, during a caching operation, data written into the write database needs to pass through read databases of equipment rooms before a corresponding cache is updated.

[0031] In an actual application, such a cache data synchronization method has at
10 least two problems:

[0032] First, in the caching mechanism, the application server at the target end preferentially reads data from a cache so as to finally perform a synchronization update on cache data at the target end. Efficiency of serial data synchronization between underlying databases is relatively low, especially when data centers are
15 deployed at different locations. As such, a delay is relatively high, and after cache update steps are added, the delay is further increased.

[0033] Second, the application server at the target end cannot actually sense an update of local underlying data and cannot update cache data immediately. Consequently, a delay is higher.

[0034] Data centers at different locations are used as an example. Two data
20 centers are respectively deployed in two cities at a distance of 1000 km from each other. Tests show that a data synchronization delay between a write database and a remote read database is about 3s. Therefore, if data at the two cities is synchronized by using a primitive data synchronization duplication solution of a database, the data
25 can be synchronized within seconds. When a user writes data at a data center of one city, it is possible that a data center in the other city cannot immediately read the written data, and a user needs to retry several seconds later. Consequently, user experience is affected.

[0035] To resolve the previous problem, the present application provides a data
30 synchronization method so as to reduce a synchronization update delay between database caches of data centers. The method includes the following steps:

[0036] After determining that data of a source end database is modified, a data synchronization source end generates a real-time notification for this data modification and sends the real-time notification to a data synchronization target end,

where the real-time notification includes information related to this data modification. Thus, after receiving the real-time notification, the data synchronization target end obtains the data modification related information by parsing the real-time notification and updates a cache in a target end database based on a parsing result.

5 **[0037]** The method can be performed by synchronization apparatuses separately located at the data synchronization source end and the data synchronization target end. As shown in FIG. 3, synchronization apparatus A located at the data synchronization source end is communicatively connected to application server A. After it is determined that application server A performs a write operation on database A,
10 synchronization apparatus A generates a real-time notification and sends the real-time notification to synchronization apparatus B located at the data synchronization target end; and synchronization apparatus B directly updates cache B based on the real-time notification.

[0038] It can be understood that, in an actual application design, all related
15 functional modules of a synchronization apparatus can be located in a physical entity (for example, a dedicated server) that is relatively independent of an application server, as shown in FIG. 3. Alternatively, some or all of the functional modules can be located in an application server. When all the functional modules of each synchronization apparatus are located in an application server, in essence, the
20 functions of the synchronization apparatus on each side of the data synchronization source end and the data synchronization target end are integrated into the application server on the corresponding side, a corresponding system architecture can be simplified as shown in FIG. 4. Interaction between a synchronization apparatus and an application server is simplified as internal interaction in the application server.

25 **[0039]** In the technical solution provided in the present application, a real-time notification is introduced to reduce an update delay of a cache in a data synchronization target end database without too much adjustment on the primitive data synchronization solution of the underlying database. In addition, an update operation of a local cache at the data synchronization source end can be directly
30 implemented by using the existing technical solution, and it is basically irrelevant to the update delay of the cache in the data synchronization target end database. Therefore, no further description is provided in the present application.

[0040] It should be noted that, based on the systems shown in FIG. 3 and FIG. 4, a typical configuration solution is that database A is a write database and database B is a

read database. Certainly, there is no necessary connection between the solution in the present application and a read/write separation mechanism of a database. For example, database A and database B are both write databases; or a write database and a read database are deployed in one equipment room, and a read database is deployed in another equipment room. It does not affect implementation of the solution in the present application.

[0041] The following further describes in detail the data synchronization method provided in the present application with reference to a specific implementation.

[0042] FIG. 5 shows a data synchronization procedure of applying a solution in the present application to an active-active system. For ease of description, the following implementation describes the whole procedure with reference to the simplified system architecture shown in FIG. 4.

[0043] S101. A data synchronization source end determines that data of a source end database is modified.

[0044] The data modification can be determined after application server A sends a data write request to database A, or after it is determined that data has been successfully written to database A. The former method can further reduce a delay of the whole procedure theoretically, and the latter method can avoid a situation that data fails to be written to a data synchronization source end database. Consequently, an update of a cache at a data synchronization target end is inconsistent with that in the database. It can be flexibly set by a person skilled in the art based on factors such as system stability and an application scenario.

[0045] S102. The data synchronization source end generates a real-time notification for this data modification and sends the real-time notification to a data synchronization target end.

[0046] In this implementation, application server A writes data modification details corresponding to each data write operation performed on database A to the real-time notification. In essence, the modification details are written to a message body corresponding to the real-time notification. The modification details here should include at least modified value information, for example, {Zhang San, male, 20}. As agreed based on an actual data storage format of a database and data transmission methods of data on two sides, the details can further include information such as a data table identifier, a row identifier, and a column identifier related to the modification, for example, {name="Zhang San", age="20"}. The present application

does not limit specific content of the "modification details". After being generated, the real-time notification is immediately sent to the data synchronization target end.

[0047] It can be understood that in a process of generating and sending the real-time notification, compression, encryption, or another coding scheme can be further used to improve transmission efficiency, encrypted transmission, etc. This implementation of the present application does not limit a specific generation and transmission method of the real-time notification.

[0048] In addition, as a contrast, after application server A writes data successfully to database A in S103, database A starts to initiate a synchronization operation on the database for this modification. It should be noted that, for the data synchronization source end, steps S102 and S103 are two relatively independent operations. Therefore, the sequence of performing the two steps is not limited. For the data synchronization target end, step 104 is performed after the real-time notification is received.

[0049] S104. The data synchronization target end obtains data modification details by parsing the real-time notification, and directly updates a cache in a target end database based on a parsing result.

[0050] At the data synchronization target end, application server B performs real-time detection on a real-time notification receiving port. After the real-time notification is received, application server B parses it immediately. Based on the processing of the real-time notification performed by application server A, application server B may need to perform corresponding operations such as decompression and decryption during parsing. Details are not described here.

[0051] In this implementation, after obtaining the data modification details of database A by parsing the real-time notification, application server B directly updates cache B in the local database based on the parsing result. Because data is not transmitted in real time to a remote location during synchronization of data of an underlying database, but is packed and transmitted in batches after the number of modified data blocks reaches a certain value, an average transmission speed of the real-time notification is higher than a synchronization speed of the underlying database. As shown in S105 in FIG. 4, usually, the data synchronization target end can directly update the local cache based on the real-time notification before synchronization of the underlying database is completed. Tests show that in a remote active-active system with a distance of about 1000 km, a synchronization average

delay of an underlying database is about 3s. A synchronization update delay of cache data can be reduced to about 30 ms or even lower through a real-time notification, and the speed of database cache synchronization can be improved significantly.

[0052] It should be noted that application server B sends an update message to a cache server immediately after the real-time notification is received to perform an update operation. However, after receiving the update message, the cache server can further determine whether cache data needs to be updated based on an actual situation. For example, the determining can be performed based on a timestamp for writing the data, or some other information can be included in a notification message to determine whether to update a cache based on the information. For example, version number information can be included so that a version number can be viewed to determine whether a cache needs to be updated. In essence, in some situations, although application server B performs an update operation, whether cache data is finally successfully updated needs to be determined by the cache server. This update method does not affect implementation of the solution in the present application.

[0053] In the previous implementation, the data synchronization source end adds the modification details of the local database to the real-time notification and uses a low delay characteristic of the real-time notification, so the data synchronization target end can complete a synchronization update of the cache in the local database before synchronization of the underlying database is completed. However, in an actual application, an amount of information that can be transmitted in the real-time notification can be limited by a plurality of objective factors such as transmission bandwidth and a storage capacity of a notification message center. If an amount of data modified by the data synchronization source end at a time is too large, it is difficult to transmit these modification details to the data synchronization target end through a real-time notification. The present application provides a corresponding solution for this situation. The previous real-time notification can be transferred by using a message system. For example, an existing message system can be used or a dedicated message system can be established. A structure and a form of the message system are not limited.

[0054] FIG. 6 shows another data synchronization procedure of applying a solution in the present application to an active-active system. For ease of description, the following implementation describes the whole procedure with reference to the system architecture shown in FIG. 3.

[0055] S201. A data synchronization source end determines that data of a source end database is modified.

[0056] The present step is similar to S101. Details are not described in this implementation again.

5 **[0057]** S202. The data synchronization source end generates a real-time notification for this data modification and sends the real-time notification to a data synchronization target end.

[0058] When application server A performs a data write operation on database A, identifier information is generated for each write operation. For example, the
10 identifier information can be a sequence number or an identifier character string generated by using a specific algorithm. When a write request is initiated to database A, application server A submits details (such as a value, a data table identifier, a row identifier, and a column identifier) of data that needs to be modified together with the identifier information to database A. In this implementation, because of a limited
15 bearer capability of a real-time notification message body, application server A writes the identifier information to the real-time notification. After being generated, the real-time notification is immediately sent to the data synchronization target end.

[0059] S203. After application server A writes data successfully to database A, database A starts to initiate a synchronization operation on the database for this
20 modification. In a synchronization process, identifier information of the previous modification is also transmitted to database B together with specific data modification content.

[0060] Similar to the previous implementation, for the data synchronization source end, steps S202 and S203 are two relatively independent operations. Therefore,
25 the sequence of performing the two steps is not limited. For the data synchronization target end, step 204 is performed after the real-time notification is received.

[0061] S204. The data synchronization target end obtains a data modification operation identifier by parsing the real-time notification and monitors a target end database based on a parsing result.

30 **[0062]** At the data synchronization target end, application server B performs real-time detection on a real-time notification receiving port. After the real-time notification is received, application server B parses it immediately. In this implementation, after obtaining, by parsing the real-time notification, an operation identifier of modification performed by the source end on database A. Application

server B starts to monitor database B for the identifier.

[0063] S205. Synchronization from the data synchronization source end to an underlying database of the data synchronization target end is complete.

5 **[0064]** S206. The data synchronization source end updates a cache in the target end database based on synchronized content of the target end database.

[0065] After detecting, through monitoring, that a modification operation corresponding to the identifier has been synchronized from database A to database B, the application server reads corresponding data from database B and updates cache B in the local database based on the reading result.

10 **[0066]** Because a real-time notification transmission speed is higher than a synchronization speed of the underlying database, after receiving the real-time notification, application server B can usually update cache B after a period of time. In contrast to the previous implementation, in this implementation, the data synchronization target end can update the local cache only after synchronization of
15 the underlying database is completed. However, the real-time notification is used so that the data synchronization target end can learn in advance that the underlying database is to be modified. Through monitoring, a cache can be updated by obtaining data immediately after the synchronization is completed. Therefore, a cache synchronization update delay can still be reduced to some extent compared with the
20 existing technical solution.

[0067] The two data synchronization methods described in the previous implementations can reduce a synchronization update delay of a cache at the data synchronization target end at different levels. When a capacity of a real-time notification is not limited, the first solution can be selected to deliver a relatively low
25 delay. In an actual application, an amount of data in a database to be modified by the application server is usually uncertain. A customer information file (CIF) system is used as an example. A data volume of modifying personal user information is relatively small. For a large account associated with a large number of sub-accounts, a modification operation can involve change of data in a large number of sub-accounts.
30 The present application further provides an adaptive data synchronization solution for this situation:

[0068] After determining that the data of the source end database is modified, the data synchronization source end first determines whether a data volume of this modification exceeds a predetermined threshold. The threshold is determined based

on a bearer capability of the real-time notification, and the bearer capability of the real-time notification depends on factors such as transmission bandwidth and a storage capacity of a notification message center. Certainly, the threshold can be further determined based on factors such as compression and encoding performed during generation of the real-time notification. In conclusion, if it is determined that the real-time notification can include modification details related to this modification operation, the modification details including at least specific value information related to this modification are written to the real-time notification. If it is determined that the real-time notification cannot include the modification details related to this modification operation, only an operation identifier of this modification is written to the real-time notification.

[0069] After receiving the real-time notification, the data synchronization target end first parses the real-time notification. If the parsing result includes the specific value information related to the modification, the cache in the target end database is directly updated based on the parsing result; otherwise, the target end database is monitored based on the modification operation identifier obtained by parsing. When it is detected that a modification operation corresponding to the identifier has been synchronized from the source end database to the target end database, the cache in the target end database is updated by obtaining corresponding data from the target end database.

[0070] In the previous solution, the data synchronization source end adaptively generates a real-time notification that includes different content based on a data volume related to the modification. As such, for small-scale data modification, the data synchronization target end can update data in the local cache without waiting for completion of synchronization of the underlying database. For large-scale data modification, data in the local cache can be updated immediately after synchronization is completed.

[0071] It can be understood that the synchronization update process of the cache for only one data modification is shown in the previous implementations. For continuous data modifications, the previous procedure only needs to be repeated. In addition, only the active-active system is used as an example for description in the previous implementations. For a multi-active system that includes at least three data centers, a cache update procedure of a data synchronization source end and data synchronization target ends is similar. The implementation details of the present

application are not repeated here.

[0072] The present application further provides a data synchronization source end apparatus. As shown in FIG. 7, the apparatus can include a real-time notification generation module 110 and a real-time notification sending module 120.

5 [0073] The real-time notification generation module 110 is configured to generate a real-time notification for this data modification after it is determined that data of a source end database is modified, where the real-time notification includes information related to this data modification.

[0074] The real-time notification sending module 120 is configured to send the
10 real-time notification to a data synchronization target end. Thus, after receiving the real-time notification, the data synchronization target end obtains the data modification related information by parsing the real-time notification and updates a cache in a target end database based on a parsing result.

[0075] In a specific implementation of the present application, the real-time
15 notification generation module 110 can be specifically configured to write specific value information related to this modification to the real-time notification after it is determined that the data of the source end database is modified.

[0076] In a specific implementation of the present application, the real-time
20 notification generation module 110 can be specifically configured to write an operation identifier of this modification to the real-time notification after it is determined that the data of the source end database is modified.

[0077] In a specific implementation of the present application, the real-time
notification generation module 110 can be specifically configured to determine
25 whether a data volume of this modification exceeds a predetermined threshold after it is determined that the data of the source end database is modified; and if no, write specific value information related to this modification to the real-time notification; or if yes, write an operation identifier of this modification to the real-time notification.

[0078] In a specific implementation of the present application, the data
synchronization source end apparatus can be disposed in a data synchronization
30 source end application server.

[0079] The present application further provides a data synchronization target end apparatus. As shown in FIG. 8, the apparatus can include a real-time notification receiving module 210 and a cache data update module 220.

[0080] The real-time notification receiving module 210 is configured to receive a

real-time notification sent by a data synchronization source end.

[0081] The cache data update module 220 is configured to obtain data modification related information by parsing the real-time notification and update a cache in a target end database based on a parsing result.

5 **[0082]** After the data synchronization source end determines that data of a source end database is modified, the real-time notification is generated and sent for this data modification, and the real-time notification includes information related to this data modification.

10 **[0083]** In a specific implementation of the present application, the real-time notification can include specific value information related to this modification.

[0084] Correspondingly, the cache data update module 220 can be specifically configured to directly update the cache in the target end database based on the modification related specific value information obtained by parsing.

15 **[0085]** In a specific implementation of the present application, the real-time notification can include an operation identifier of this modification.

[0086] Correspondingly, the cache data update module 220 can be specifically configured to monitor the target end database based on the modification operation identifier obtained by parsing, and when it is detected that a modification operation corresponding to the identifier has been synchronized from the source end database to
20 the target end database, update the cache in the target end database based on synchronized content of the target end database.

[0087] In a specific implementation of the present application, the real-time notification can include specific value information related to this modification or an operation identifier of this modification.

25 **[0088]** Correspondingly, the cache data update module 220 can be specifically configured to directly update the cache in the target end database based on the parsing result if the parsing result is modification related specific value information; or if the parsing result is a modification operation identifier, monitor the target end database based on the identifier, and when it is detected that a modification operation
30 corresponding to the identifier has been synchronized from the source end database to the target end database, update the cache in the target end database based on synchronized content of the target end database.

[0089] In a specific implementation of the present application, the data synchronization target end apparatus can be disposed in a data synchronization target

end application server.

[0090] The present application further provides a data synchronization system. As shown in FIG. 9, the system can include a data synchronization source end apparatus 100 and a data synchronization target end apparatus 200.

5 [0091] The data synchronization source end apparatus 100 can include a real-time notification generation module 110 and a real-time notification sending module 120.

[0092] The real-time notification generation module 110 is configured to generate a real-time notification for this data modification after it is determined that data of a source end database is modified, where the real-time notification includes information
10 related to this data modification.

[0093] The real-time notification sending module 120 is configured to send the real-time notification to a data synchronization target end.

[0094] The data synchronization target end apparatus 200 can include a real-time notification receiving module 210 and a cache data update module 220.

15 [0095] The real-time notification receiving module 210 is configured to receive the real-time notification.

[0096] The cache data update module 220 is configured to obtain data modification related information by parsing the real-time notification and update a cache in a target end database based on a parsing result.

20 [0097] In a specific implementation of the present application, the real-time notification generation module 120 can be specifically configured to write specific value information related to this modification to the real-time notification after it is determined that the data of the source end database is modified.

[0098] Correspondingly, the cache data update module 220 can be specifically
25 configured to directly update the cache in the target end database based on the modification related specific value information obtained by parsing.

[0099] In another specific implementation of the present application, the real-time notification generation module 120 can be specifically configured to write an operation identifier of this modification to the real-time notification after it is
30 determined that the data of the source end database is modified.

[0100] Correspondingly, the cache data update module 220 can be specifically configured to monitor the target end database based on the modification operation identifier obtained by parsing, and when it is detected that a modification operation corresponding to the identifier has been synchronized from the source end database to

the target end database, update the cache in the target end database based on synchronized content of the target end database.

[0101] In another specific implementation of the present application, the real-time notification generation module 120 can be specifically configured to determine whether a data volume of this modification exceeds a predetermined threshold after it is determined that the data of the source end database is modified; and if no, write specific value information related to this modification to the real-time notification; or if yes, write an operation identifier of this modification to the real-time notification.

[0102] Correspondingly, the cache data update module 220 can be specifically configured to directly update the cache in the target end database based on the parsing result if the parsing result is modification related specific value information; or if the parsing result is a modification operation identifier, monitor the target end database based on the identifier, and when it is detected that a modification operation corresponding to the identifier has been synchronized from the source end database to the target end database, update the cache in the target end database based on synchronized content of the target end database.

[0103] For an implementation process of functions and roles of each module in the apparatus, refer to an implementation process of a corresponding step in the previous method. Details are not described here again.

[0104] In an actual application design, all related functional modules of the data synchronization source end apparatus and the data synchronization target end apparatus can be separately located in physical entities (for example, dedicated servers) that are relatively independent of the application servers, as shown in FIG. 3. Alternatively, some or all of the functional modules can be located in the application servers. When all the functional modules of the synchronization apparatuses are located in the application servers, in essence, the functions of the synchronization apparatuses on both sides of the data synchronization source end and the data synchronization target end are integrated into the application servers on the both sides, a corresponding system architecture can be simplified as shown in FIG. 4, and interaction between the synchronization apparatuses and the application servers is simplified as internal interaction in the application servers.

[0105] The data synchronization source end apparatus and the data synchronization target end apparatus provided in the present application can be separately disposed on a hardware device such as a server. FIG. 10 is a schematic

diagram illustrating a hardware structure of a device for configuring the previous apparatus according to the present application. The device can include a processor 1010, a memory 1020, an input/output interface 1030, a communications interface 1040, and a bus 1050. The processor 1010, the memory 1020, the input/output interface 1030, and the communications interface 1040 implement mutual communication connections in the device by using the bus 1050.

[0106] The processor 1010 can be a general-purpose central processing unit (CPU), a microprocessor, an application specific integrated circuit (ASIC), or one or more integrated circuits, configured to execute a related program to implement the technical solution provided in the present application.

[0107] The memory 1020 can be implemented in a form of a read only memory (ROM), a random access memory (RAM), a static storage device, a dynamic storage device, etc. The memory 1020 can store an operating system and another application program. When the technical solutions provided in the present application are implemented by using software or firmware, related program code is stored in the memory 1020 and is invoked and executed by the processor 1010.

[0108] The input/output interface 1030 is configured to connect to an input/output module to implement information input and output. The input/output module can be configured as a component (not shown in the figure) in the device, or it can be connected to the device to provide a corresponding function. An input device can include a keyboard, a mouse, a touchscreen, a microphone, a sensor, etc. An output device can include a display, a loudspeaker, a vibrator, an indicator, etc.

[0109] The communications interface 1040 is configured to connect to a communications module (not shown in the figure) so as to implement communication and interaction between the present device and another device. The communications module can implement communication in a wired manner (such as a USB or a network cable), or it can implement communication in a wireless manner (such as a mobile network, Wi-Fi, or Bluetooth).

[0110] The bus 1050 includes a channel for transmitting information between components (such as the processor 1010, the memory 1020, the input/output interface 1030, and the communications interface 1040) of the device.

[0111] It should be noted that although only the processor 1010, the memory 1020, the input/output interface 1030, the communications interface 1040, and the bus 1050 of the previous device are shown, in a specific implementation process, the device can

further include another component essential for normal running. In addition, a person skilled in the art can understand that the device can include only components necessary for implementing the solutions in the present application, and it does not necessarily include all components shown in the figure.

5 **[0112]** It can be learned from descriptions of the implementations that a person skilled in the art can clearly understand that the present application can be implemented by using software in addition to a necessary universal hardware platform. Based on such an understanding, the technical solutions in the present application essentially or the part contributing to the existing technology can be implemented in a
10 form of a software product. The software product can be stored in a storage medium, such as a ROM/RAM, a magnetic disk, or an optical disc, and includes several instructions for instructing a computer device (a personal computer, a server, or a network device) to perform the methods described in the implementations or in some parts of the implementations of the present application.

15 **[0113]** The implementations in the present specification are described in a progressive manner. For same or similar parts in the implementations, refer to each other. Each implementation focuses on a difference from other implementations. Particularly, an apparatus or system implementation is similar to a method implementation, and therefore is described briefly. For related parts, refer to related
20 descriptions in the method implementation. The described apparatus or system implementations are merely examples. The modules described as separate parts can be physically separate or not. When the solutions in the present application are implemented, the functions of each module can be implemented in one or more pieces of software and/or hardware. Some or all the modules can be selected based on actual
25 needs to achieve the objectives of the solutions of the implementations. A person of ordinary skill in the art can understand and implement the implementations of the present application without creative efforts.

[0114] The previous descriptions are merely specific implementations of the present application. It should be noted that a person of ordinary skill in the art can
30 make several improvements or polishing without departing from the principle of the present application and the improvements or polishing shall fall within the protection scope of the present application.

CLAIMS

1. A data synchronization method, wherein the method is applied to a data synchronization source end, the method comprising the steps of:

determining that data of a source end database has been modified;

generating a real-time notification for the data modification, wherein the real-time notification comprises an operation identifier of the data modification, and information related to the data modification; and

sending the real-time notification to a data synchronization target end, so that after receiving the real-time notification, the data synchronization target end obtains the data modification related information and the operation identifier by parsing the real-time notification; monitors based on the operation identifier, detects that a modification operation corresponding to the operation identifier has been synchronized from the source end database to the target end database, and updates the cache in the target end database based on synchronized content of the target end database.

2. The method according to claim 1, wherein generating a real-time notification for the data modification comprises writing the operation identifier of the data modification to the real-time notification.

3. The method according to claim 1, wherein generating a real-time notification for the data modification comprises:

determining whether a data volume of the data modification exceeds a predetermined threshold;

in response to determining that the data volume of the data modification does exceed the predetermined threshold, writing the operation identifier of the data modification to the real-time notification.

4. The method of claim 1, further comprising initiating a synchronization operation on the source end database.

5. A data synchronization method, wherein the method is applied to a data synchronization target end, the method comprising the steps of:

receiving, by a data synchronization source end, a real-time notification sent by a data synchronization source end; and

obtaining data modification related information, comprising:

parsing the real-time notification; and

updating a cache in a target end database based on a parsing result,

wherein, after the data synchronization source end determines that data of a source end database is modified (201), the real-time notification is generated and sent for the data modification (202), and the real-time notification comprises an operation identifier of the data modification, and information related to the data modification; and

wherein updating a cache in a target end database based on a parsing result comprises:

monitoring the target end database based on the operation identifier obtained by parsing, and

in response to detecting that a modification operation corresponding to the operation identifier has been synchronized from the source end database to the target end database, updating the cache in the target end database based on synchronized content of the target end database.

6. The method according to claim 5, wherein a data volume of the information related to the data modification in the real-time notification does not exceed a predetermined threshold.

7. The method of any one of claims 1 to 6, wherein determining, by the data synchronization source end, that data of a source end database is modified comprises determining that data of the source end database is modified after an application server at the source end sends a data write request to the database at the source end, or after it is determined that data has been successfully written to the database at the source end.

8. A data synchronization source end apparatus configured to perform the method of any

one of claims 1 to 4.

9. A data synchronization target end apparatus configured to perform the method of any one of claims 5 to 6.