DATABASE DATA MODIFICATION REQUEST PROCESSING METHOD AND APPARATUS

TECHNICAL FIELD

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

BACKGROUND

[0002] A database is a set of software running on a server, and is used to store, query, and manage large volumes of data. A database log is a storage content used by the database, and can be used to record operations that occurred in the database, for example, data modification operations. When the database is restarted, the log can be used to recover a previous normal running state. If the database sends the log to another database, the another database can also use the received log to recover a copy identical to the database that sends the log. This is a method for primary/secondary database synchronization. The log includes data after modification and data before modification, and in some scenarios even includes other data related to modified data and such.

[0003] In the existing technology, a user can send a data modification request to a database. The database correspondingly modifies data and generates a log based on the data modification request, performs persistence processing on the generated log (for example, stores the generated log in a hard disk), and then returns a modification result that corresponds to the data modification request to the user (for example, notify the user that modification succeeds). It is worthwhile to note that the generated log is usually in a transient state, and performing persistence processing is to prevent accidental losses caused by power failure or restart. Further, if the database has a secondary database, the database also first synchronizes the generated log to the secondary database, and then returns the modification result that corresponds to the data modification request.

[0004] However, performing persistence processing on the generated log or

synchronizing the generated log to the secondary database both need to consume a lot of time. Consequently, it takes a relatively long time for a database data modification request processing mode in the existing technology.

SUMMARY

[0005] Implementations of the present application provide a database data modification request processing method and apparatus, to resolve a problem in the existing technology that a database data modification request processing mode takes a relatively long time.

[0006] The following technical solutions are used in the implementations of the present application:

[0007] An implementation of the present application provides a database data modification request processing method, including: receiving a data modification request for a database; modifying second data in the database to first data and generating a first log and a second log based on the data modification request, where the first log includes the first data, and the second log includes the second data; and returning a modification result that corresponds to the data modification request after performing persistence processing on the first log.

[0008] An implementation of the present application provides a database data modification request processing apparatus, including: a receiving module, configured to receive a data modification request for a database; a processing module, configured to modify second data in the database to first data and generate a first log and a second log based on the data modification request, where the first log includes the first data, and the second log includes the second data; and a return module, configured to return a modification result that corresponds to the data modification request after persistence processing is performed on the first log.

[0009] The at least one technical solution used in the implementations of the present application can achieve the following beneficial effects: Compared with the existing technology for generating a complete log that includes data after modification, data before modification, and other data related to modified data, in the solutions of the present application, two kinds of logs, namely the first log and the second log, that are not necessarily complete and whose contents are not identical, can be generated for any data modification request. The first data is the data after modification, and the

second data is the data before modification. The first log can include only the first data but not the second data and the other data related to the modified data. Because a database status can be recovered or primary/secondary synchronization can be implemented based on only the data after modification without the second data and the other data related to the modified data, the modification result can be returned to the user after persistence processing is performed on the first log. Also, persistence processing consumes less time as a data volume of the first log is less than a data volume of a complete log, so that a database data modification request processing time can be reduced. Therefore, the problem in the existing technology can be partially or comprehensively alleviated.

BRIEF DESCRIPTION OF DRAWINGS

[0010] The accompanying drawings described here are intended to provide a further understanding of the present application, and constitute a part of the present application. The illustrative implementations of the present application and descriptions thereof are intended to describe the present application, and do not constitute limitations on the present application. In the accompanying drawings:

[0011] FIG. 1 is a schematic flowchart illustrating a database data modification request processing method, according to an implementation of the present application;

[0012] FIG. 2 is a schematic diagram illustrating a data modification request processing procedure in an actual application scenario, according to the existing technology;

[0013] FIG. 3 is a schematic diagram illustrating a data modification request processing procedure in an actual application scenario, according to an implementation of the present application;

[0014] FIG. 4 is a schematic diagram illustrating a procedure for sending a log to an external system such as a data analysis system in an actual application scenario, according to an implementation of the present application;

[0015] FIG. 5 is a schematic diagram illustrating a procedure that a secondary database processes a received first log synchronized by a primary database in an actual application scenario, according to an implementation of the present application;

[0016] FIG. 6 is a schematic diagram illustrating interactions between terminals in the solutions of the present application in an actual application scenario (by using

scenarios in FIG. 2 to FIG. 4), according to an implementation of the present application; and

[0017] FIG. 7 is a schematic structural diagram illustrating a database data modification request processing apparatus, according to an implementation of the present application.

DESCRIPTION OF IMPLEMENTATIONS

[0018] To make the objectives, technical solutions, and advantages of the present application clearer, the following clearly and comprehensively describes the technical solutions of the present application with reference to the implementations of the present application and the corresponding accompanying drawings. Apparently, the described implementations are merely some but not all of the implementations of the present application. All other implementations obtained by a person of ordinary skill in the art based on the implementations of the present application without creative efforts shall fall within the protection scope of the present application.

[0019] FIG. 1 is a schematic flowchart illustrating a database data modification request processing method, according to an implementation of the present application.

[0020] A procedure in FIG. 1 can be performed by a database, or a function module that does not belong to the database but can be connected to the database. A device where the database and/or the function module are/is located is not limited in the present application. The device includes but is not limited to a personal computer, a medium and large computer, a computer cluster, a mobile phone, a tablet computer, a smart watch, an in-vehicle mobile station, etc. that serves as a server or a client.

[0021] The procedure in FIG. 1 can include the following steps.

[0022] S101. Receive a data modification request for a database.

[0023] In the present implementation of the present application, the data modification request is a request for requesting to modify data in the database. The request can be manually sent by a user of the database, or automatically sent by executing an automation script. The data modification request includes at least one of the following requests: a data update request, a data deletion request, and a data addition request.

[0024] A format of the data modification request is not limited in the present application, and can be a structured query language (SQL) statement format, or can be

other formats supported by the database.

[0025] S102. Modify second data in the database to first data and generate a first log and a second log based on the data modification request, where the first log includes the first data, and the second log includes the second data.

[0026] In the present implementation of the present application, the data modification request can indicate a request for performing a specific data modification operation. For example, the data modification request indicates a request for modifying the second data in the database to the first data. Certainly, in practice, the data modification request can be a request not that specific. For example, a data modification request can indicate a request for reducing a total volume of data in the database by half, etc. Data to be reduced can be determined by the database based on certain policies.

[0027] In the present implementation of the present application, the first data or the second data can be null. For example, for a data deletion request, the second data is not null, and the first data is null; for a data addition request, the first data is not null, and the second data is null; for a data update request, neither the first data nor the second data is null.

[0028] Contents of the first data and the second data are not limited in the present application, and can be a part or all of any type of data such as a database record, a data table, a data index, or a data graph.

[0029] In the present implementation of the present application, a log needs to include at least the first data if database recovery or primary/secondary synchronization needs to be performed based on the log. The second data and other data related to modified data are dispensable for database recovery or primary/secondary synchronization, and therefore can be excluded from the log. In the solutions of the present application, a problem in the existing technology can be alleviated based on such an idea. In the existing technology, there is only one kind of log (which is referred to as a complete log for ease of description), and both the first data and the second data are included in the complete log. The solutions of the present application provide two kinds of logs: the first log and the second log. The first log can be a simplified log specialized in database recovery or primary/secondary synchronization. Preferably, the first log can include only the first data (so that a data volume of the first log can be as small as possible). The second log can be a complete log or an incomplete log used to supplement the first log. For example, the second log

can include both the second data and the first data, and can even include the other data related to the modified data and such. Alternatively, the second log can include only the second data.

[0030] In the present implementation of the present application, alternatively, the second log or the second log in combination with the first log can be used for database recovery or primary/secondary synchronization, and can be further used for tasks in addition to database recovery or primary/secondary synchronization, for example, performing real-time analysis on a data change in the database.

[0031] In the present implementation of the present application, a sequence of performing the two sub-steps of "modifying second data to first data" and "generating a first log and a second log" in step S102 is not limited. Usually, a modification operation is first performed, and a corresponding log is then generated. However, in practice, a log can be first generated, and a corresponding modification operation can be then performed, to prevent a log loss caused by an exception such as power failure that occurs when the modification operation is performed. In this case, even if an exception such as power failure occurs when performing the modification operation, the log can be rolled back based on an actual status of data in the database that is back to normal. It is helpful to ensure that the rolled-back log does not lose a log that should be included, and does not include a log with an error or a redundant log.

[0032] S103. Return a modification result that corresponds to the data modification request after performing persistence processing on the first log.

[0033] In the present implementation of the present application, the first log and the second log generated in step S102 can be generated in a memory or a buffer, and are in a transient and non-stable state after being generated. Usually, if at this time the database is restarted or powered off, the generated first log and the generated second log would be lost, and subsequently database recovery or primary/secondary synchronization cannot be performed, or other functions that need to use the logs cannot be implemented. To prevent such situations, persistence processing can be first performed on the first log, and the corresponding modification result can be then returned to a sender of the data modification request, to ensure that at least the first log that persistence processing is performed on can be used to perform database recovery or primary/secondary synchronization.

[0034] In the present implementation of the present application, persistence processing can be performed on the second log for the similar reason. However, the

persistence processing performed on the second log can also not be placed on a key path of the data modification request processing procedure. The procedure in FIG. 1 does not include "performing persistence processing on the second log", and the execution of any step in the procedure in FIG. 1 does not need to be based on "performing persistence processing on the second log".

[0035] For example, a main thread or process can be used to perform the procedure in FIG. 1, and another thread (for example, a backend thread) or process can be used to perform persistence processing on the second log. The two actions may not interfere with each other.

[0036] In the present implementation of the present application, a content included in the returned modification result is not limited. The modification result can include only description of whether modification succeeds, or can include description of data after modification.

[0037] It is worthwhile to note that the steps in the procedure in FIG. 1 can be performed by the same device or different devices. For example, step S101 and step S102 can be performed by device 1, and step S103 can be performed by device 2. For another example, step S101 can be performed by device 1, and step S102 and step S103 can be performed by device 2.

[0038] According to the previous method, compared with the existing technology for generating a complete log that includes data after modification, data before modification, and other data related to modified data, two kinds of logs, namely the first log and the second log, that are not necessarily complete and whose contents are not identical can be generated for any data modification request in the solutions of the present application. The first data is the data after modification, and the second data is the data before modification. The first log can include only the first data, and does not include the second data and the other data related to the modified data. Because a database status can be recovered or primary/secondary synchronization can be implemented based on only the data after modification without the second data and the other data related to the modified data, the modification result can be returned to the user after persistence processing is performed on the first log. Also, persistence processing consumes less time as a data volume of the first log is less than a data volume of a complete log, so that a database data modification request processing time can be reduced. Therefore, the problem in the existing technology can be alleviated.

[0039] In addition, because the second log is also generated in the solutions of the present application, the second log can be a complete log or an incomplete log used to supplement the first log. Therefore, compared with a log in the existing technology, a total log content is not lacking, and the quantity of resources consumed to generate a log are almost the same. A function that can be implemented based on a log in the existing technology can still be implemented in the solutions of the present application.

[0040] Based on the previous method, an implementation of the present application further provides some implementation solutions and extended solutions of the previous method, which are described below.

[0041] In the present implementation of the present application, the persistence processing performed on the first log or the second log is mentioned above. The persistence processing is described below for ease of understanding. The persistence processing can include the following: storing a log in a nonvolatile storage device; and more specifically, can be recording the log in a file in a certain format, and then storing the file in the nonvolatile storage device. The nonvolatile storage device can be a storage device whose stored data is not lost when being powered off, for example, a hard disk, a read-only memory (ROM), an erasable programmable read-only memory (EPROM), an electrically erasable read-only memory (EEPROM), a flash memory, etc.

[0042] For example, a generated record can be recorded in a text, and the text can be stored in a hard disk.

[0043] In the present implementation of the present application, it can be seen from the previous description that, in addition to primary/secondary database synchronization and database recovery, the log can be used in other scenarios, such as data analysis. The second log can be used to implement scenarios, which are described in detail below.

[0044] In some application scenarios, the database is connected to an external system such as a data analysis system. The log of the database can be provided for the external system for other uses. The data analysis system is used as an example. The data analysis system can perform real-time or non-real-time analysis on a data change in the database based on the log sent by the database, and provide an analysis result for a related user, organization, etc.

[0045] For the accuracy and comprehensiveness of the data analysis result, in the

data analysis system, the log needs to include data after modification (namely, the first data) and data before modification (namely, the second data), and even needs to include other related data (for example, other data that belongs to the same database record as the modified data).

[0046] In the present implementation of the present application, the second log can satisfy the need of the data analysis system. After the second log is generated or persistence processing is performed on the second log, the second log can be sent to the data analysis system, so that the data analysis system performs data analysis based on the second log. An analysis content of the data analysis system is not limited in the present application.

[0047] In the present implementation of the present application, to further improve robustness of the solutions of the present application, the modification result can be temporarily withheld after persistence processing is performed on the first log, and returned after the first log is synchronized to the secondary database or another device. As such, it is convenient for the secondary database to synchronize the modification in time. In addition, it is equivalent to backing up the first log. The first log can still be retrieved from the secondary database or another device even if a storage device that is used for persistence processing on the first log is faulty.

[0048] For example, when the database is a primary database, before step S103 of returning a modification result that corresponds to the data modification request, the following step can be further performed: synchronizing the first log to a secondary database of the database, so that the secondary database synchronizes the modification based on the first log.

[0049] According to the previous analysis, in an actual application scenario, the first log is a simplified log that includes only the first data, and the second log is a complete log that includes at least the first data and the second data. Based on FIG. 2 and FIG. 3, a processing procedure in the existing technology and a processing procedure in the solutions of the present application are compared and analyzed.

[0050] FIG. 2 is a schematic diagram illustrating a data modification request processing procedure in an actual application scenario, according to the existing technology. If the procedure in FIG. 2 is performed by a primary database, the procedure can include the following steps:

[0051] S201: The primary database receives a data modification request sent by a user.

[0052] S202: The primary database performs a corresponding data modification operation based on the data modification request.

[0053] S203: The primary database generates a second log in a memory based on the data modification request or the performed data modification operation.

[0054] S204: The primary database performs persistence processing on the generated second log.

[0055] S205: The primary database synchronizes the second log that persistence processing is performed on to a secondary database.

[0056] S206: The primary database responds to the user that modification succeeds.

[0057] If an exception occurs in any one of the steps S201 to S205, step S206 may not be performed before the exception is removed.

[0058] FIG. 3 is a schematic diagram illustrating a data modification request processing procedure in an actual application scenario, according to an implementation of the present application. If the procedure in FIG. 3 is also performed by a primary database, the procedure can include the following steps:

[0059] S301: The primary database receives a data modification request sent by a user.

[0060] S302: The primary database performs a corresponding data modification operation based on the data modification request.

[0061] S303: The primary database generates a first log and a second log in a memory based on the data modification request or the performed data modification operation.

[0062] S304: The primary database performs persistence processing on the generated first log.

[0063] S305: The primary database synchronizes the first log that persistence processing is performed on to a secondary database.

[0064] S306: The primary database responds to the user that modification succeeds.

[0065] If an exception occurs in any one of the steps S301 to S305, step S306 may not be performed before the exception is removed.

[0066] It can be seen from the comparison between the procedures in FIG. 2 and FIG. 3 that, because a data volume of the first log is less than a data volume of the second log, compared with the existing technology, a data volume of a log that

persistence processing needs to be performed on, before the primary database responds to the user that modification succeeds, can be reduced in the solutions of the present application. As such, a data modification request processing time can be reduced.

[0067] In addition, the primary database synchronizes the log to the secondary database in both of the procedures in FIG. 2 and FIG. 3. The log synchronization also needs to occupy corresponding bandwidth and consume a corresponding time. Because the data volume of the first log is less than the data volume of the second log, compared with the existing technology, the bandwidth occupied by and/or the time consumed by the log synchronization can also be reduced in the solutions of the present application.

[0068] As shown in FIG. 4, an implementation of the present application further provides a schematic diagram illustrating a procedure for sending a log to an external system such as a data analysis system in an actual application scenario. If the procedure in FIG. 4 is also performed by a primary database, the procedure can include the following steps:

[0069] S401: The primary database detects a second log newly generated in a memory.

[0070] S402: The primary database obtains the newly generated second log.

[0071] S403: The primary database performs persistence processing on the obtained second log.

[0072] S404: The primary database sends the second log to a predetermined external system.

[0073] In the present implementation of the present application, after receiving a log synchronized by the primary database, a secondary database can also perform operations such as performing persistence processing on the log and/or sending the log to the external system. However, such operation process is different from the process in the existing technology. The following describes the operations related to the secondary database.

[0074] In the existing technology, a secondary database receives a complete log (namely, the second log in the previous example), and therefore can directly perform operations such as performing persistence processing on the received log and/or sending the received log to an external system.

[0075] In the solutions of the present application, the secondary database receives

a simplified log (namely, the first log in the previous example), and can only directly perform persistence processing on the received log, and it is not suitable to send the received log to the external system. In practice, the secondary database may also need to use a complete log. Therefore, the secondary database can obtain a corresponding complete log based on the simplified log, and then can perform operations such as performing persistence processing on the complete log and/or sending the complete log to the external system. As such, the consistency between logs in the primary and secondary databases can be improved. Based on such an idea, after the database synchronizes the first log to the secondary database of the database, the secondary database can perform the following step: The secondary database synchronizes the modification and generates the second log based on the first log. Further, the secondary database can perform the following steps: The secondary database performs persistence processing on the first log obtained through synchronization and/or the second log generated based on the first log; and the secondary database sends the second log to the predetermined external system.

[0076] As shown in FIG. 5, an implementation of the present application provides a schematic diagram illustrating a procedure that a secondary database processes a received first log synchronized by a primary database in an actual application scenario (by using the scenarios in FIG. 2 to FIG. 4). If the procedure in FIG. 5 is performed by a secondary database of a primary database, the procedure can include the following steps:

[0077] S501: The secondary database receives a first log synchronized by the primary database.

[0078] S502: The secondary database performs persistence processing on the received first log.

[0079] S503: The secondary database replays the first log.

[0080] S504: The secondary database synchronizes a data modification operation and generates a second log in a memory through the replay.

[0081] S505: The secondary database performs persistence processing on the second log.

[0082] In the present implementation of the present application, as described above, the two kinds of logs: the first log and the second log, can be generated for any data modification request. To increase a subsequent speed of reading the two kinds of logs, the first log and the second log can be separately stored in different nonvolatile

storage devices when performing persistence process. Therefore, the two kinds of logs can be separately read based on performance (for example, bandwidth or a reading speed) of respective storage devices, and can be read in parallel.

[0083] In the present implementation of the present application, as described above, the complete log or the second log can include other data related to modified data in addition to first data and second data. Usually, if the first data and the second data each belong to at least one database record, the "other data related to the modified data" that can be further included in the second log includes but is not limited to: data other than the first data in the database record that the first data belongs to, and/or data other than the second data in the database record that the second data belongs to.

[0084] For ease of understanding, Table 1 and Table 2 are used as an example for description.

Table 1

	column_1	**column_2**	**…**	**column_*n***
a	a_1	a_2	…	a_*n*
b	b_1	b_2	…	b_*n*
c	c_1	c_2	…	c_*n*
…	…	…	…	…

[0085] Table 1 is a partial content of a data table in a database, including three database records: a, b, and c. Each record includes *n* key value pairs, *n* keys are respectively column_1 to column_*n*, and values that correspond to the keys in each database record are shown in Table 1. For ease of description, a key value pair and a row number of a database record that the key value pair belongs to are indicated by using <row>:<key, value>, where "key" indicates a key, "value" indicates a value, and "row" indicates the row number of the database record that the key value pair belongs to.

[0086] Assume that the database receives a data modification request sent by a user. Based on the data modification request, <a>:<column_2, a_2> is modified to <a>:<column_2, a_2'>, and <b>:<column_2, b_2> is modified to <b>:<column_2, b_2'>. The modified Table 1 is shown in the following Table 2.

Table 2

	column_1	**column_2**	**…**	**column_*n***
a	a_1	a_2'	…	a_*n*
b	b_1	b_2'	…	b_*n*
c	c_1	c_2	…	c_*n*
…	…	…	…	…

[0087] For the previous example, the first data is <a>:<column_2, a_2'> and <b>:<column_2, b_2'>, and the second data is <a>:<column_2, a_2> and <b>:<column_2, b_2>.

[0088] The other data related to the modified data can include other data than the modified data in record a and record b, namely, <a>:<column_1, a_1>, …, <a>:<column_*n*, a_*n*>, and <b>:<column_1, b_1>, …, <b>:<column_*n*, b_*n*>.

[0089] The generated first log can include only the first data, and does not include the second data and the other data related to the modified data, as shown in the following:"<a>:<column_2, a_2'>, and<b>: <column_2, b_2'>".

[0090] The generated second log can include not only the first data but also the second data and the other data related to the modified data, as shown in the following:"<a>: second data and other data related to modified data: <column_1, a_1>, <column_2, a_2>, …, <column_*n*, a_*n*>; first data: <column_2, a_2'>; and<b>: second data and other data related to modified data: <column_1, a_1>, <column_2, a_2>, …, <column_*n*, a_*n*>; first data: <column_2, a_2'>".

[0091] It is worthwhile to note that a representation format of a content included in the log in the previous example is merely an example, and is not contended to limit the present application. In practice, the content can also be represented in other formats.

[0092] As shown in FIG. 6, an implementation of the present application further provides a schematic diagram illustrating interaction between terminals in the solutions of the present application in an actual application scenario (by using the scenarios in FIG. 2 to FIG. 4), to understand the solutions of the present application more intuitively.

[0093] Four terminals are mainly included in FIG. 6, and are respectively a user, a

primary database, a secondary database, and a data analysis system. Steps that correspond to numbers ① to ⑤ in FIG. 6 are main steps of a data modification request processing method provided in the present implementation of the present application, and steps without numbers are secondary steps. All the steps are separately described in detail above, and details are omitted here for simplicity.

[0094] The above is the database data modification request processing method provided in the implementations of the present application. As shown in FIG. 7, an implementation of the present application further provides a corresponding apparatus based on the same idea.

[0095] FIG. 7 is a schematic structural diagram illustrating a database data modification request processing apparatus, according to an implementation of the present application. The apparatus includes the following: a receiving module 701, configured to receive a data modification request for a database; a processing module 702, configured to modify second data in the database to first data, and generate a first log and a second log based on the data modification request, where the first log includes the first data, and the second log includes the second data; and a return module 703, configured to return a modification result that corresponds to the data modification request after persistence processing is performed on the first log.

[0096] Optionally, the second log further includes the first data.

[0097] Optionally, the processing module 702 is configured to generate the first log and the second log in a memory.

[0098] Optionally, the processing module 702 is further configured to perform persistence processing on the second log.

[0099] Optionally, the apparatus further includes the following: a sending apparatus 704, configured to send the second log to a data analysis system, so that the data analysis system performs data analysis based on the second log.

[0100] Optionally, when the database is a primary database, the apparatus further includes the following: a synchronization module 705, configured to synchronize the first log to a secondary database of the database before the return module 703 returns the modification result that corresponds to the data modification request, so that the secondary database synchronizes the modification based on the first log.

[0101] Optionally, the first data and the second data each belong to at least one database record.

[0102] The second log further includes data other than the first data in the database record that the first data belongs to, and/or data other than the second data in the database record that the second data belongs to.

[0103] Optionally, the persistence processing includes the following: storing a log in a nonvolatile storage device.

[0104] Optionally, the persistence processing includes the following: storing a log in a nonvolatile storage device.

[0105] The first log and the second log are separately stored in different nonvolatile storage devices by using the persistence processing.

[0106] Optionally, the data modification request is a data update request, a data deletion request, or a data addition request.

[0107] Neither the first data nor the second data is null when the data modification request is a data update request; or the first data is null and the second data is not null when the data modification request is a data deletion request; or the first data is not null and the second data is null when the data modification request is a data addition request.

[0108] The apparatus in FIG. 7 can be located in the database or on a function module that does not belong to the database but can be connected to the database.

[0109] The apparatuses provided in the present application are in a one-to-one correspondence with the methods provided in the present application. Therefore, the apparatuses and the methods have the similar beneficial technical effects. The beneficial technical effects of the methods have been described above in detail, and therefore the beneficial technical effects of the apparatuses are omitted here for simplicity.

[0110] A person skilled in the art should understand that an implementation of the present invention can be provided as a method, a system, or a computer program product. Therefore, the present invention can use a form of hardware only implementations, software only implementations, or implementations with a combination of software and hardware. Moreover, the present invention can use a form of a computer program product that is implemented on one or more computer-usable storage media (including but not limited to a disk memory, a CD-ROM, an optical memory, etc.) that include computer-usable program code.

[0111] The present invention is described with reference to the flowcharts and/or block diagrams of the method, the device (system), and the computer program product

according to the implementations of the present invention. It should be understood that computer program instructions can be used to implement each process and/or each block in the flowcharts and/or the block diagrams and a combination of a process and/or a block in the flowcharts and/or the block diagrams. The computer program instructions can be provided for a general-purpose computer, a special-purpose computer, a built-in processor, or a processor of another programmable data processing device to generate a machine, so that the instructions executed by the computer or the processor of the another programmable data processing device generate an apparatus for implementing a specified function in one or more flows in the flowcharts and/or in one or more blocks in the block diagrams.

[0112] The computer program instructions can be stored in a computer readable memory that can instruct the computer or the another programmable data processing device to work in a specific way, so that the instructions stored in the computer readable memory generate an artifact that includes an instruction apparatus. The instruction apparatus implements a specified function in one or more flows in the flowcharts and/or in one or more blocks in the block diagrams.

[0113] The computer program instructions can be loaded onto the computer or the another programmable data processing device, so that a series of operations and steps are performed on the computer or the another programmable device, thereby generating computer-implemented processing. Therefore, the instructions executed on the computer or the another programmable device provide steps for implementing a specified function in one or more flows in the flowcharts and/or in one or more blocks in the block diagrams.

[0114] In a typical configuration, a computing device includes one or more processors (CPU), an input/output interface, a network interface, and a memory.

[0115] The memory can include a non-persistent memory, a random access memory (RAM), a nonvolatile memory, and/or another form in a computer readable medium, for example, a read-only memory (ROM) or a flash memory (flash RAM). The memory is an example of the computer readable medium.

[0116] The computer readable medium includes persistent, non-persistent, movable, and unmovable media that can implement information storage by using any method or technology. Information can be a computer readable instruction, a data structure, a program module, or other data. An example of a computer storage medium includes but is not limited to a phase-change random access memory

(PRAM), a static random access memory (SRAM), a dynamic random access memory (DRAM), another type of random access memory (RAM), a read-only memory (ROM), an electrically erasable programmable read-only memory (EEPROM), a flash memory or another memory technology, a compact disc read-only memory (CD-ROM), a digital versatile disc (DVD) or another optical storage, a cassette magnetic tape, a tape and disk storage or another magnetic storage device or any other non-transmission media that can be configured to store information that a computing device can access. Based on the definition in the present specification, the computer readable medium does not include transitory computer-readable media (transitory media), for example, a modulated data signal and carrier.

[0117] It is worthwhile to further note that the terms "include", "include", or their any other variant is intended to cover a non-exclusive inclusion, so that a process, a method, an article, or a device that includes a list of elements not only includes those elements but also includes other elements which are not expressly listed, or further includes elements inherent to such process, method, article, or device. An element preceded by "includes a ..." does not, without more constraints, preclude the existence of additional identical elements in the process, method, merchandise, or device that includes the element.

[0118] The previous descriptions are merely implementations of the present application, and are not intended to limit the present application. For a person skilled in the art, the present application can have various modifications and changes. Any modifications, equivalent substitutions, improvements, etc. made within the spirit and principle of the present application shall fall in the protection scope of the claims in the present application.

CLAIMS

1. A computer-implemented method, comprising:

receiving, from a user and at a primary database, a data modification request for modifying data in the primary database, wherein the data modification request specifies one or more updated values of one or more respective first database records in first columns;

modifying, in response to the data modification request, one or more previous values of the one or more respective first database records in the first columns to have the one or more updated values;

generating, based on the data modification request, a first log and a second log, wherein the second log comprises a) the one or more updated values and the one or more previous values of the one or more respective first database records in the first columns that are being modified and b) previous values of second database records in second columns that are not being modified, and wherein the first log comprises only the one or more updated values of the one or more respective first database records in the first columns for use in database recovery or synchronization;

performing persistence processing including storing the first columns in the first log that are being modified in persistent storage before returning a modification result to the user and storing the second columns in the second log that are not being modified in the persistent storage only after returning the modification result to the user; and

returning the modification result to the user that is indicative of having successfully modified the data in the primary database in response to receiving the data modification request.

2. The computer-implemented method of claim 1, wherein the data modification request comprises at least one of a data update request, a data deletion request, and a data addition request.

3. The computer-implemented method of claim 1, wherein performing the persistence processing on the first log comprises:

recording the first log as a file in a certain format, and

storing the file in a nonvolatile storage device.

4. The computer-implemented method of claim 1, further comprising, prior to returning the modification result:

temporarily withholding the modification result in response to performing the persistence processing on the first log; and

synchronizing the first log to a secondary database so that the secondary database synchronizes the modification based on the first log.

5. The computer-implemented method of claim 4, wherein the secondary database synchronizing the modification based on the first log comprises:

receiving, by the secondary database, the first log synchronized by the primary database;

performing, by the secondary database, the persistence processing on the first log; and

performing a replay on the first log.

6. A non-transitory, computer-readable medium storing one or more instructions executable by a computer system to perform operations comprising:

receiving, from a user and at a primary database, a data modification request for modifying data in the primary database, wherein the data modification request specifies one or more updated values of one or more respective first database records in first columns;

modifying, in response to the data modification request, one or more previous values of the one or more respective first database records in the first columns to have the one or more updated values;

generating, based on the data modification request, a first log and a second log, wherein the second log comprises a) the one or more updated values and the one or more previous values of the one or more respective first database records in the first columns that are being modified and b) previous values of second database records in second columns that are not being modified, and wherein the first log comprises only the one or more updated values of the one or more respective first database records in the first columns for use in database recovery or synchronization;

performing persistence processing including storing the first columns in the first log that are being modified in persistent storage before returning a modification result to the user and

storing the second columns in the second log that are not being modified in the persistent storage only after returning the modification result to the user; and

returning the modification result to the user that is indicative of having successfully modified the data in the primary database in response to receiving the data modification request.

7. The non-transitory, computer-readable medium of claim 6, wherein the data modification request comprises at least one of a data update request, a data deletion request, and a data addition request.

8. The non-transitory, computer-readable medium of claim 6, wherein performing the persistence processing on the first log comprises:

recording the first log as a file in a certain format, and

storing the file in a nonvolatile storage device.

9. The non-transitory, computer-readable medium of claim 6, wherein the operations further comprise, prior to returning the modification result:

temporarily withholding the modification result in response to performing the persistence processing on the first log; and

synchronizing the first log to a secondary database so that the secondary database synchronizes the modification based on the first log.

10. The non-transitory, computer-readable medium of claim 9, wherein the secondary database synchronizing the modification based on the first log comprises:

receiving, by the secondary database, the first log synchronized by the primary database;

performing, by the secondary database, the persistence processing on the first log; and

performing a replay on the first log.

11. A computer-implemented system, comprising:

one or more computers; and

one or more computer memory devices interoperably coupled with the one or more computers and having tangible, non-transitory, machine-readable media storing one or more instructions that, when executed by the one or more computers, perform operations comprising:

receiving, from a user and at a primary database, a data modification request for modifying data in the primary database, wherein the data modification request specifies one or more updated values of one or more respective first database records in first columns;

modifying, in response to the data modification request, one or more previous values of the one or more respective first database records in the first columns to have the one or more updated values;

generating, based on the data modification request, a first log and a second log, wherein the second log comprises a) the one or more updated values and the one or more previous values of the one or more respective first database records in the first columns that are being modified and b) previous values of second database records in second columns that are not being modified, and wherein the first log comprises only the one or more updated values of the one or more respective first database records in the first columns for use in database recovery or synchronization;

performing persistence processing including storing the first columns in the first log that are being modified in persistent storage before returning a modification result to the user and storing the second columns in the second log that are not being modified in the persistent storage only after returning the modification result to the user; and

returning the modification result to the user that is indicative of having successfully modified the data in the primary database in response to receiving the data modification request.

12. The computer-implemented system of claim 11, wherein the data modification request comprises at least one of a data update request, a data deletion request, and a data addition request.

13. The computer-implemented system of claim 11, wherein performing the persistence processing on the first log comprises:

recording the first log as a file in a certain format, and

storing the file in a nonvolatile storage device.

14. The computer-implemented system of claim 11, wherein the operations further comprise, prior to returning the modification result:

temporarily withholding the modification result in response to performing the persistence processing on the first log; and

synchronizing the first log to a secondary database so that the secondary database synchronizes the modification based on the first log, wherein the secondary database synchronizing the modification based on the first log comprises:

receiving, by the secondary database, the first log synchronized by the primary database;

performing, by the secondary database, the persistence processing on the first log; and

performing a replay on the first log.