DATA QUERY METHOD, STORAGE MEDIUM, AND COMPUTER SYSTEM

TECHNICAL FIELD

[0001] This specification relates to the field of computer technologies, and in particular, to a data query method, a storage medium, and a computer system.

BACKGROUND

[0002] Secure multi-party computation is a privacy-preserving computation method, and allows a plurality of participants to perform computing without exposing privacy data stored by the plurality of participants. During privacy-preserving computation, each participant first queries, from a stored data file, data on which privacy-preserving computation needs to be performed, and then all participants jointly perform privacy-preserving computation based on data obtained through query. For example, an institution A and an institution B jointly compute a common user. The institution A or the institution B can first query a user identifier from stored user data, and then the institution A and the institution B jointly perform privacy-preserving computation based on a user identifier obtained through query, to determine the common user. However, each participant stores data in a file storage manner, and no index mechanism exists, so that a speed is slow when each participant queries data from the data file.

[0003] Therefore, this specification provides a data query method.

SUMMARY

[0004] Embodiments of this specification provide a data query method, a storage medium, and a computer system, to partially resolve the above-mentioned problems in the conventional technology.

[0005] This specification provides a data query method. The method is applied to any participant that performs secure multi-party computation, and the method includes: determining a to-be-queried statement, and determining a conditional statement included in the to-be-queried

statement; determining a cache tree of the participant, where the cache tree includes several nodes, an edge between the nodes corresponds to a query condition, and each node corresponds to cache data pre-cached by the participant; determining, based on the query condition corresponding to the edge between the nodes, a node hit by the conditional statement, and using the node as a first node; and determining target data corresponding to the to-be-queried statement based on cache data corresponding to the first node.

[0006] This specification provides a computer system, including a storage, a processor, and a computer program stored in the storage and capable of running on the processor. When the processor executes the program, the following steps are implemented: determining a to-be-queried statement, and determining a conditional statement included in the to-be-queried statement; determining a cache tree of a participant, where the cache tree includes several nodes, an edge between the nodes corresponds to a query condition, and each node corresponds to cache data pre-cached by the participant; determining, based on the query condition corresponding to the edge between the nodes, a node hit by the conditional statement, and using the node as a first node; and determining target data corresponding to the to-be-queried statement based on cache data corresponding to the first node.

[0007] This specification provides a computer-readable nonvolatile storage medium. The storage medium stores a computer program, and when the computer program is executed by a processor, the data query method is implemented.

[0008] The above-mentioned at least one technical solution used in the embodiments of this specification can achieve at least one of the following beneficial effects: A to-be-queried statement is determined, and a conditional statement included in the to-be-queried statement is determined. Then, a cache tree of a participant is determined, and a node hit by the conditional statement is determined based on a query condition corresponding to an edge between nodes in the cache tree, and is used as a first node. Target data corresponding to the to-be-queried statement is determined based on cache data corresponding to the first node. A direct query of data corresponding to the conditional statement from a data file is avoided, to reduce a quantity of times of accessing the data file. In addition, a full query of pre-cached data is avoided, to improve a query speed.

BRIEF DESCRIPTION OF DRAWINGS

[0009] The accompanying drawings described here are used to provide a further understanding of this specification and constitute a part of this specification. The example embodiments of this specification and descriptions thereof are used to explain this specification, and do not constitute an improper limitation on this specification. In the accompanying drawings:

[0010] FIG. 1 is a schematic flowchart of a data query method according to this specification;

[0011] FIG. 2 is a schematic diagram of a cache tree according to this specification;

[0012] FIG. 3 is a schematic diagram of an updated cache tree according to this specification;

[0013] FIG. 4 is a schematic diagram of another updated cache tree according to this specification;

[0014] FIG. 5 is a schematic diagram of a data query process according to this specification; and

[0015] FIG. 6 is a schematic structural diagram of a computer system according to an embodiment of this specification.

DESCRIPTION OF EMBODIMENTS

[0016] To make the objectives, technical solutions, and advantages of this specification clearer, the following clearly and comprehensively describes the technical solutions of this specification with reference to specific embodiments and corresponding accompanying drawings of this specification. Clearly, the described embodiments are merely some but not all of embodiments of this specification. All other embodiments obtained by a person of ordinary skill in the art based on the embodiment of this specification without creative efforts shall fall within the protection scope of this specification.

[0017] The following describes in detail the technical solutions provided in the embodiments of this specification with reference to the accompanying drawings.

[0018] FIG. 1 is a schematic flowchart of a data query method in this specification. The method shown in FIG. 1 is applied to any participant that performs secure multi-party computation, and can specifically include the following steps.

[0019] S100: Determine a to-be-queried statement, and determine a conditional statement included in the to-be-queried statement.

[0020] In one or more embodiments of this specification, currently, during privacy-preserving computation, each participant first queries, from a stored data file, data on which privacy-preserving computation needs to be performed, and then all participants jointly perform privacy-preserving computation based on data obtained through query.

[0021] An example in which all participants jointly determine a common user is used for description. Each participant can first query a user identifier from stored service data, and then all participants jointly perform privacy-preserving computation based on a user identifier obtained through query, to determine the common user.

[0022] Based on this, the participant can determine the to-be-queried statement, and determine the conditional statement included in the to-be-queried statement. The participant can be a domestic platform, institution, or organization; or can be overseas platform, institution, or organization. This is not specifically limited in this specification. For ease of description, the following provides descriptions by using an example in which the participant is an execution body. In addition, the participant can provide a service for a user. Specifically, the participant can be a bank that provides a financial service for the user. The participant can alternatively be a transaction platform that provides a transaction service for the user. This is not specifically limited in this specification.

[0023] The to-be-queried statement can be a query statement entered by an operator, or can be a to-be-queried statement delivered by a central server. This is not specifically limited in this specification. The central server can be a third-party server trusted by each participant. An example in which all the participants jointly determine the common user is used for description. The to-be-queried statement can be querying the user identifier from the stored service data. The service data is service data historically generated when the user uses a service provided by the participant to execute a service. When the participant is the bank that provides the financial service, the service data includes data such as a user identifier, a name, a gender, an age, and a transfer amount. When the participant is the transaction platform that provides the transaction service, the service data includes data such as a user identifier, a name, a transaction quantity, and a transaction amount.

[0024] When determining the conditional statement included in the to-be-queried statement, the participant can extract the conditional statement included in the to-be-queried statement. The conditional statement includes a conditional field, a conditional field value, and a conditional relationship. The conditional relationship is a relationship between the conditional field and the

conditional field value.

[0025] S102: Determine a cache tree of the participant, where the cache tree includes several nodes, an edge between the nodes corresponds to a query condition, and each node corresponds to cache data pre-cached by the participant.

[0026] In one or more embodiments of this specification, the participant can determine the cache tree of the participant. The cache tree includes several nodes, the edge between the nodes corresponds to the query condition, and each node corresponds to the cache data pre-cached by the participant. The cache tree is constructed by the participant based on a historical query statement and data obtained through querying based on the query statement. In other words, the participant can pre-construct the cache tree based on the historical query statement and the data corresponding to the query statement. Data corresponding to each node in the cache tree is data that corresponds to the query statement and that is pre-cached by the participant. A specific storage that specifically stores the data corresponding to the query statement is not specifically limited in this specification. For ease of description, the following provides descriptions by using an example in which cache data corresponding to each node in the cache tree is stored in a cache.

[0027] In one or more embodiments of this specification, a root node in the cache tree usually does not correspond to any cache data, and the root node can be an empty node. The query condition corresponding to the edge between the nodes can include a query field, a query field value, and a query relationship. The query relationship is a relationship between the query field and the query field value, and the query relationship can be a first-type relationship, a second-type relationship, or a third-type relationship. The first-type relationship is a "greater" relationship. To be specific, the query field is greater than the query field value. The second-type relationship is a "less" relationship. To be specific, the query field is less than the query field value. The third-type relationship is an "equal" relationship. To be specific, the query field is equal to the query field value. Certainly, when the query field value is a plurality of values, the query relationship can further include a fourth-type relationship. The fourth-type relationship is a "belong" relationship. To be specific, the query field belongs to a set including the query field value.

[0028] For example, the participant historically queries data of a user whose age is greater than 20 in a data file, namely, data 1; further queries data of a user whose transaction quantity is less than 30 in the data file, namely, data 2; and further queries data in which a height is in a range of 160 to 165 in the data file, namely, data 3. The participant can construct the cache quantity based

on the conditional statements, that is, age>20, transaction quantity<30, and height is in [160, 165], the data 1, the data 2, and the data 3. Specifically, FIG. 2 is a schematic diagram of a cache tree according to this specification. Nodes 1 to 3 in FIG. 2 respectively correspond to the pre-cached data 1 to 3 of the participant. A node 0 is an empty node, and the node 0 is the root node of the cache tree. A query condition corresponding to an edge between the node 0 and the node 1 is age>20, the query relationship in the query condition is the first-type relationship, and the edge between the node 0 and the node 1 is a solid line edge pointing from the node 0 to the node 1. A query condition corresponding to an edge between the node 0 and a node 2 is transaction quantity<30, the query relationship in the query condition is the second-type relationship, and the edge between the node 0 and the node 2 is a solid line edge pointing from the node 2 to the node 0. A query condition corresponding to an edge between the node 0 and a node 3 is height is in [160, 165], the query relationship in the query condition is the fourth-type relationship, and the edge between the node 0 and the node 1 is a dashed line edge pointing from the node 0 to the node 3.

[0029] S104: Determine, based on the query condition corresponding to the edge between the nodes, a node hit by the conditional statement, and use the node as a first node.

[0030] In one or more embodiments of this specification, the participant can determine, based on the query condition corresponding to the edge between the nodes, the node hit by the conditional statement, and use the node as the first node. When a query condition corresponding to an edge between the node and a parent node of the node is the same as the conditional statement, it indicates that the node is a node hit by the conditional statement. When the query condition corresponding to the edge between the node and the parent node of the node is partially the same as the conditional statement, and a query range of the query condition is in a query range of the query condition corresponding to the edge between the nodes in the cache tree, and is closest to a query range of the conditional statement, it indicates that the node is a node hit by the conditional statement. That the query condition corresponding to the edge between the node and the parent node of the node is partially the same as the conditional statement means that the query range of the conditional statement is a proper subset of the query range of the query condition corresponding to the edge between the node and the parent node of the node. However, when the query condition corresponding to the edge between the node and the parent node of the node is completely different from the conditional statement, in other words, the query range of the conditional statement is not a subset of the query range of the query condition corresponding to the edge between the node and

the parent node of the node, it indicates that the node is not a node hit by the conditional statement.

[0031] In one or more embodiments of this specification, the conditional statement includes at least one subcondition, each subcondition is a sum relationship, and can be connected by using "and". When the query condition corresponding to the edge between the node and the parent node of the node is the same as all subconditions included in the conditional statement, in other words, a subcondition included in the query condition is the same as the subcondition included in the conditional statement, it indicates that the node is a node hit by the conditional statement. In addition, when the query condition corresponding to the edge between the node and the parent node of the node is the same as a part of subconditions included in the conditional statement, and in the query condition corresponding to the edge between the nodes in the cache tree, the query condition (namely, the query condition corresponding to the edge between the node and the parent node of the node) includes the most subconditions in the conditional statement, in other words, the query condition includes the subcondition in the conditional statement, and the query condition includes the most subconditions, it indicates that the node is a node hit by the conditional statement.

[0032] For example, with reference to FIG. 2, it is assumed that a conditional statement 1 is age>20 and birthday<4, and the conditional statement 1 includes two subconditions: age>20 and birthday<4. In FIG. 2, if a query condition that includes the most subconditions in the conditional statement is the query condition corresponding to the edge between the node 0 and the node 1, that is, age>20, the node 1 is a node hit by the conditional statement 1. Then, it is assumed that a conditional statement 2 is transaction quantity<30, and the conditional statement 2 includes one subcondition: the transaction quantity<30. The query condition corresponding to the edge between the node 0 and the node 2 in FIG. 2 is the same as the conditional statement 2. Therefore, the node 2 is a node hit by the conditional statement 2.

[0033] Specifically, the participant can sequentially determine each node of the cache tree as a query node starting from the root node of the cache tree, determine a child node whose parent node is the query node, and use the child node as a second node; determine a query condition of an edge between the second node and the query node, and use the query condition as a first condition; determine whether a query range corresponding to the conditional statement is a subset of a query range corresponding to the first condition; and if yes, use the second node as a new query node, and until the query range corresponding to the conditional statement is not a subset of the query range corresponding to the first condition, and the participant can use the query node as

the first node hit by the conditional statement.

[0034] When the query range corresponding to the conditional statement is a subset of the query range corresponding to the first condition, it indicates that the first condition includes at least one subcondition in the conditional statement. In other words, the first condition is at least the same as at least one subcondition in the conditional statement. The participant can continue to use the second node as a new query node, continue to determine the second node whose parent node is the query node, continue to determine the first condition, and determine whether the query range corresponding to the conditional statement is a subset of the query range corresponding to the first condition, and until the query range corresponding to the conditional statement is not a subset of the query range corresponding to the first condition, the participant can use the query node as the first node in the conditional statement.

[0035] In addition, if the query range corresponding to the conditional statement is not a subset of the query range corresponding to the first condition, it indicates that the first condition does not include any subcondition in the conditional statement, in other words, the first condition is different from all subconditions in the conditional statement, and the participant can directly use the query node as the first node hit by the conditional statement.

[0036] S106: Determine target data corresponding to the to-be-queried statement based on cache data corresponding to the first node.

[0037] In one or more embodiments of this specification, the participant can determine the target data corresponding to the to-be-queried statement based on the cache data corresponding to the first node. The cache data is data pre-cached by the participant, and the cache data is data corresponding to the first node. The cache data satisfies a query condition corresponding to an edge between the first node and a parent node of the first node. The target data is data that satisfies the conditional statement in the to-be-queried statement.

[0038] Specifically, the participant can determine the parent node of the first node in the cache tree, determine the query condition corresponding to the edge between the parent node and the first node, and use the query condition as a second condition; use the cache data corresponding to the first node as the target data corresponding to the to-be-queried statement when the second condition is the same as the conditional statement; and when a query range of the conditional statement is a proper subset of a query range of the second condition, filter the cache data corresponding to the first node based on the conditional statement, and use filtered data as the

target data corresponding to the to-be-queried statement.

[0039] When the second condition is the same as the conditional statement, it indicates that the second condition includes all subconditions in the conditional statement, in other words, the second condition is exactly the same as the conditional statement, and the participant can use the cache data corresponding to the first node as the target data corresponding to the to-be-queried statement. However, when the query range of the conditional statement is a proper subset of the query range of the second condition, it indicates that the second condition includes at least one subcondition in the conditional statement, in other words, the second condition is the same as at least one subcondition in the conditional statement. In this case, the participant can filter the cache data corresponding to the first node based on the conditional statement, to obtain through filtering data that completely satisfies the conditional statement, namely, the filtered data, and then use the filtered data as the target data corresponding to the to-be-queried statement.

[0040] The above-mentioned example continues to be used. With reference to FIG. 2, the conditional statement 1 is age>20 and birthday<4, the node 1 is a node hit by the conditional statement 1, and the query condition corresponding to the edge between the node 0 and the node 1 is the second condition, that is, age>20. The query range of the conditional statement 1 is a proper subset of the query range of the second condition, in other words, age>20 and birthday<4 is a proper subset of age>20. The participant can filter cache data corresponding to the node 1 based on age>20 and birthday<4, and use filtered data as the target data corresponding to the to-be-queried statement, that is, data of age>20 and birthday<4 in the cache data corresponding to the node 1.

[0041] It can be learned from the above-mentioned method that in this application, during data query, the participant can determine the to-be-queried statement, and determine the conditional statement included in the to-be-queried statement. Then, the cache tree of the participant is determined, and the node hit by the conditional statement is determined based on the query condition corresponding to the edge between nodes in the cache tree, and is used as the first node. The node hit by the conditional statement is determined by using the cache tree, to improve a hit rate of the cache data. Then, the target data corresponding to the to-be-queried statement is determined based on the cache data corresponding to the first node. During data query, the data corresponding to the conditional statement is first queried from pre-cached cache data, to avoid a direct query of data corresponding to the conditional statement from a data file, and reduce a

quantity of times of accessing the data file. In addition, when the data corresponding to the conditional statement is queried from the cache data, the participant can determine, based on the cache tree, the node hit by the conditional statement, and then determine the target data based on the cache data corresponding to the node, to avoid performing a full query on the pre-cached data and improve a query speed.

[0042] In one or more embodiments of this specification, the participant can determine a prestored data file when no node is hit by the conditional statement in the cache tree; and query data corresponding to the to-be-queried statement from the data file, and use the data as the target data. The data file stores all service data of the participant. When performing data query, the participant can determine, from the data file based on the to-be-queried statement, data that satisfies the to-be-queried statement, and use the data as the target data. Specifically, for how to determine, from the data file based on the to-be-queried statement, the data that satisfies the to-be-queried statement, the participant can use any existing data query manner. This is not specifically limited in this specification.

[0043] For example, with reference to FIG. 2, it is assumed that a conditional statement 3 in the to-be-queried statement is age>10, and the conditional statement 3 does not hit any node in the cache tree shown in FIG. 2. Therefore, the participant can determine the prestored data file, query data corresponding to the to-be-queried statement from the data file, that is, data of age>10 in the data file, and use the data as the target data.

[0044] In one or more embodiments of this specification, after the target data corresponding to the to-be-queried statement is obtained, the participant can update the cache tree based on the to-be-queried statement and the target data, so that when data query is subsequently reperformed based on the cache tree, data corresponding to the query statement can be determined from the cache data pre-cached by the participant. Based on this, the participant can sequentially determine each node in the cache tree as an update node starting from a root node of the cache tree, determine a child node whose parent node is the update node, determine a query condition corresponding to an edge between the update node and the child node, and use the query condition as a third condition; and determine whether at least one subcondition in the conditional statement is the same as the third condition; and if yes, the participant uses the child node as a new update node, and until the third condition is different from all subconditions in the conditional statement, the participant can construct a first child node whose parent node is the update node, determine cache

data corresponding to the first child node as the target data, and determine the query condition corresponding to the edge between the update node and the first child node as the conditional statement.

[0045] The conditional statement includes at least one subcondition. When at least one subcondition in the conditional statement is the same as the third condition, in other words, when the third condition is completely the same as at least one subcondition in the conditional statement, it indicates that cache data corresponding to the child node satisfies the at least one subcondition in the conditional statement. The participant can continue to use the child node as a new update node, continue to determine a child node whose parent node is the update node, determine the third condition, and determine whether at least one subcondition in the conditional statement is the same as the third condition, and until the third condition is different from all subconditions in the conditional statement, in other words, when the third condition includes a subcondition different from all the subconditions in the conditional statement, the participant can update the cache tree based on the conditional statement and the target data.

[0046] In addition, when the third condition is different from all the subconditions in the conditional statement, the participant can construct the first child node whose parent node is the update node, determine the cache data corresponding to the first child node as the target data, and determine the query condition corresponding to the edge between the update node and the first child node as the conditional statement. The update node can be a root node in the cache tree, or can be a node in the cache tree other than the root node.

[0047] In one or more embodiments of this specification, the cache tree may include a plurality of child nodes whose parent node is the update node. Therefore, the participant can sequentially determine each node in the cache tree as an update node starting from the root node of the cache tree, and determine a child node whose parent node is the update node. Then, for each child node, a query condition corresponding to an edge between the update node and the child node is determined, and is used as the third condition. The participant needs to determine whether the third condition includes a third condition the same as at least one subcondition in the conditional statement; and if yes, determine the third condition the same as at least one subcondition in the conditional statement, use the third condition as a fifth condition, determine a child node for which a query condition of an edge between the child node and the update node is the fifth condition, and use the determined child node as a new update node, and until all third conditions are different

from all the subconditions in the conditional statement, construct the first child node whose parent node is the update node, determine cache data corresponding to the first child node as the target data, and determine the query condition corresponding to the edge between the update node and the first child node as the conditional statement.

[0048] However, if all the third conditions include no third condition the same as at least one subcondition in the conditional statement, in other words, all the third conditions are different from all the subconditions in the conditional statement, the participant can directly use an update node in this case as a parent node to construct the first child node, and determine the cache data and the query condition that correspond to the first child node. A specific process is consistent with a process when all the third conditions are different from all the subconditions in the conditional statement. Details are omitted here for simplicity.

[0049] The above-mentioned example continues to be used. With reference to FIG. 2, the conditional statement 3 is age>10, and the conditional statement 3 includes only one subcondition: age>10. The participant can sequentially perform the following operations for all nodes (namely, the node 0 to the node 3) in the cache tree starting from the root node (namely, the node 0) of the cache tree: For example, for the node 0, the participant can determine the node 0 as an update node, and determine a child node whose parent node is the update node (namely, the node 0), namely, the node 1 to the node 3. Then, for each child node (namely, the node 1 to the node 3), for example, for the node 1, the query condition corresponding to the edge between the node 0 and the node 1 is determined, and is used as the third condition, that is, age>20. Based on a process of determining the third condition in this example, each third condition that can be determined by the participant is age>20, transaction quantity<30, and height is in [160, 165].

[0050] Then, the participant needs to determine whether all the third conditions include a third condition the same as at least one subcondition in the conditional statement 3. Clearly, all the third conditions include no third condition the same as at least one subcondition in the conditional statement 3. In this case, the participant can construct the first child node whose parent node is the node 0, namely, a node 4, determine cache data corresponding to the node 4 as the target data, and determine a query condition corresponding to an edge between the node 0 and the node 4 as the conditional statement 3. Specifically, FIG. 3 is a schematic diagram of an updated cache tree according to this specification. A query relationship corresponding to the query condition of the edge between the node 0 and the node 4 is the first-type relationship, and therefore, the edge

between the node 0 and the node 4 is a solid line edge pointing from the node 0 to the node 4.

[0051] Then, with reference to FIG. 2, it is assumed that a conditional statement 1 is age>20 and birthday<4, and the conditional statement 1 includes two subconditions: age>20 and birthday<4. The participant can sequentially perform the following operations for all nodes (namely, the node 0 to the node 3) in the cache tree starting from the root node (namely, the node 0) of the cache tree: For example, for the node 0, the participant can determine the node 0 as an update node, and determine a child node whose parent node is the update node (namely, the node 0), namely, the node 1 to the node 3. Then, for each child node (namely, the node 1 to the node 3), for example, for the node 1, the query condition corresponding to the edge between the node 0 and the node 1 is determined, and is used as the third condition, that is, age>20. Based on a process of determining the third condition in this example, each third condition that can be determined by the participant is age>20, transaction quantity<30, and height is in [160, 165].

[0052] Then, the participant needs to determine whether all the third conditions include a third condition the same as at least one subcondition in the conditional statement 1. Clearly, all the third conditions include a third condition the same as at least one subcondition in the conditional statement 1, that is, age>20. Therefore, the participant can determine the third condition the same as the at least one subcondition in the conditional statement 1, and use the third condition as the fifth condition, that is, age>20. Then, a child node for which a query condition of an edge between the child node and the node 0 is determined, that is, the node 1. Then, the determined child node is used as a new update node, and until all the third conditions are different from all the subconditions in the conditional statement, if the update node is the node 1 in this case, the participant can construct the first child node whose parent node is the node 1, namely, the node 5, determine cache data corresponding to the node 5 as the target data, and determine a query condition corresponding to an edge between the node 1 and the node 5 as the conditional statement 1. Specifically, FIG. 4 is a schematic diagram of another updated cache tree according to this specification. A query relationship corresponding to the query condition of the edge between the node 1 and the node 5 is the first-type relationship and the second-type relationship. To be specific, a query relationship corresponding to age>20 is the first-type relationship, and a query relationship corresponding to birthday<4 is the second-type relationship. A subcondition not included in the fifth condition is birthday<4. Therefore, the edge between the node 1 and the node 5 is a solid line edge pointing from the node 5 to the node 1.

[0053] In one or more embodiments of this specification, if a child node whose parent node is the update node does not exist in the cache tree, the participant can construct the first child node whose parent node is the update node, determine the cache data corresponding to the first child node as the target data, and determine the query condition corresponding to the edge between the update node and the first child node as the conditional statement.

[0054] In one or more embodiments of this specification, to avoid a waste of storage resources caused by repeated storage, for the node in the cache tree, when the node has a child node, that is, a child node whose parent node is the node exists, the participant needs to delete cache data corresponding to the child node from cache data corresponding to the node, to avoid repeatedly caching the same data in the node and the child node.

[0055] Based on this, after the cache tree is updated as described above, the participant can determine the cache data corresponding to the update node, and then delete the target data from the cache data, to update the update node. The updated cache data corresponding to the update node does not include the target data, to prevent the update node from repeatedly storing the target data.

[0056] The above-mentioned example continues to be used. With reference to FIG. 4, the update node is the node 1, the cache data corresponding to the node 1 is data 1, a condition satisfied by the data 1 is age>20, and a condition satisfied by the target data is the conditional statement 1, that is, age>20 and birthday<4. The participant can delete the target data included in the data 1, to update the node 1, so that the cache data corresponding to the node 1 is data that satisfies age>20 and birthday≥4.

[0057] Based on the above-mentioned content, when the target data corresponding to the to-be-queried statement is determined based on the cache data corresponding to the first node in step S106, or when the cache data corresponding to the first node is used as the target data corresponding to the to-be-queried statement in step S106, the participant can determine, from the cache tree, the child node whose parent node is the first node. Then, the cache data corresponding to the child node is determined, and is used as sub-data, and the cache data corresponding to the first node and the sub-data are used as the target data corresponding to the to-be-queried statement.

[0058] In one or more embodiments of this specification, after the cache tree is updated, the participant can further combine nodes at the same level, to improve a query speed. Parent nodes of the nodes at the same level are the same. Specifically, the participant can determine another

child node that is in the cache tree and whose parent node is the update node, determine a query condition corresponding to an edge between the update node and the another child node, and use the query condition as a fourth condition. The another child node is a node other than the first child node. Then, when a query range of the fourth condition is a proper subset of a query range of the conditional statement, the first child node and the another child node are combined, to obtain a combined child node. Then, the target data and cache data corresponding to the another child node are combined, and combined data is used as cache data corresponding to the combined child node. A query condition corresponding to an edge between the update node and the combined child node is determined as the conditional statement.

[0059] When the query range of the fourth condition is a proper subset of the query range of the conditional statement, it indicates that the query range of the fourth condition is a subset of the query range of the conditional statement, and the query range of the fourth condition is not equal to the query range of the conditional statement, the participant can combine the first child node and another child node, so that the first child node and the another child node are combined into the same node.

[0060] In addition, when the target data and the cache data corresponding to the another child node are combined, and the combined data is used as the cache data corresponding to the combined child node, the participant can determine intersection data of the target data and the cache data corresponding to the another child node, determine data in the target data other than the intersection data, and use the as newly added data. Then, the cache data corresponding to the another child node and the newly added data are combined, and combined data is used as cache data corresponding to the combined child node.

[0061] For example, it is assumed that the another child node is the node 1, the cache data corresponding to the node 1 is data 1, a condition satisfied by the data 1 is age>20, a condition satisfied by the target data is the conditional statement 3, and the conditional statement 3 is age>10. The participant can determine intersection data of the target data and the data 1, that is, data that satisfies age>20. Then, data in the target data other than the intersection data is determined, and is used as newly added data, that is, data that satisfies 10<age≤20. Then, the data 1 and the newly added data are combined, and combined data is used as cache data corresponding to the combined child node.

[0062] In one or more embodiments of this specification, as a quantity of times the participant

performs query increases, there are more nodes in the cache tree and more cache data corresponding to the cache tree. That is, more data is in the cache. To avoid an excessive data amount in the cache, the participant can periodically clear the cache data. In addition, to avoid a waste of storage resources caused by repeated storage, for the node in the cache tree, when the node has a child node, the participant needs to delete the cache data corresponding to the child node from the cache data corresponding to the node, to avoid repeatedly caching the same data in the node and the child node, so that cache data corresponding to some parent nodes is actually cache data corresponding to a child node of the parent nodes. Therefore, when the cache data is cleared, cache data corresponding to a leaf node in the cache tree is usually first cleared.

[0063] Based on this, the participant can delete cache data corresponding to a leaf node in the cache tree based on a specified period. The specified period is a preset period, for example, three months. In addition, when the cache data corresponding to the leaf node in the cache tree is deleted, the participant can delete cache data corresponding to a preset quantity of leaf nodes in the cache tree, or can delete cache data corresponding to all leaf nodes in the cache tree. This is not limited in this specification. Specifically, the participant can randomly determine, from all the leaf nodes in the cache tree, a specific leaf node or some specific leaf nodes whose cache data is to be deleted. The preset quantity is a preset value, for example, 4.

[0064] In one or more embodiments of this specification, the to-be-queried statement can include a plurality of conditional statements, and all the conditional statements can have an "or" relationship, and can be connected by using "or". Based on this, when the conditional statement included in the to-be-queried statement is determined in step S100, the participant can determine each conditional statement included in the to-be-queried statement, and subsequently determine, for each conditional statement based on the process of step S102 to S104, the first node hit by the conditional statement.

[0065] Then, in step S106, the participant can determine the first data corresponding to the conditional statement based on the cache data corresponding to the first node hit by the conditional statement, and then combine all first data, to obtain the target data corresponding to the to-be-queried statement. The above-mentioned process of determining the first data corresponding to the conditional statement based on the cache data corresponding to the first node hit by the conditional statement is similar to the above-mentioned process of determining the target data corresponding to the to-be-queried statement based on the cache data corresponding to the first node, and only

the target data is replaced with the first data.

[0066] In one or more embodiments of this specification, the plurality of conditional statements in the to-be-queried statement may have an "and" relationship. The participant needs to first convert the plurality of conditional statements in the to-be-queried statement, to convert a relationship between all the conditional statements into an "or" relationship. Therefore, when the conditional statement included in the to-be-queried statement is determined in step S100, the participant can determine all initial conditional statements included in the to-be-queried statement, and convert all the initial conditional statements, to obtain all conditional statements. A relationship between all the conditional statements is an "or" relationship. All the conditional statements can be represented in a form of (conditional statement) or (conditional statement), and each conditional statement can be represented in a form of (subcondition and subcondition).

[0067] For example, the to-be-queried statement is select * from table where (a>1 or b>2) and (c>2 or d in(1, 2)), all initial conditional statements included in the to-be-queried statement are (a>1 or b>2) and (c>2 or d in(1, 2)), and all the initial conditional statements are connected by using "and", that is, have an "and" relationship. The participant can convert all the initial conditional statements into all conditional statements whose relationship is an "or" relationship, and all the conditional statements can be represented as (a>1 and c>2) or (a>1 and d in(1, 2)) or (b>2 and c>2) or (a>1 and d in(1, 2)). (a>1 and c>2), (a>1 and d in(1, 2)), (b>2 and c>2) and (a>1 and d in(1, 2)) are all conditional statements.

[0068] In one or more embodiments of this specification, all the conditional statements included in the to-be-queried statement may have a repeated conditional statement, and the participant needs to remove a duplicate from all the conditional statements. Therefore, when the conditional statement included in the to-be-queried statement is determined in step S100, the participant can determine all initial conditional statements included in the to-be-queried statement, and convert all the initial conditional statements, to obtain all candidate statements. A relationship between all the candidate statements is an "or" relationship. Then, the same statement in all the candidate statements is determined, and is used as a repeated statement. A quantity of repeated statements is determined, repeated statements whose quantity is the quantity minus one are deleted from all the candidate statements, and all candidate statements obtained after deletion are used as all conditional statements. It should be noted that when at least two candidate statements are the same in all the candidate statements, the same candidate statement is a repeated statement.

[0069] In one or more embodiments of this specification, FIG. 5 is a schematic diagram of a data query process according to this specification. The participant can determine the to-be-queried statement, determine all the initial conditional statements included in the to-be-queried statement, convert all the initial conditional statements, to obtain all the candidate statements, and remove a duplicate from all the candidate statements, to obtain all the conditional statements. For each conditional statement, determine a first node hit by the conditional statement based on the process of step S102 to S104, and determine first data corresponding to the conditional statement based on cache data corresponding to the first node hit by the conditional statement. Then, whether the first data corresponding to each conditional statement is determined is determined. If yes, all first data is combined, to obtain the target data corresponding to the to-be-queried statement. If no, a first node of another conditional statement different from the conditional statement is determined, and first data of the another conditional statement is determined, until first data corresponding to each conditional statement is determined.

[0070] In one or more embodiments of this specification, an example in which all participants jointly determine the common user is used for description. Target data obtained by the participant through query is a user identifier, and the user identifier is a unique identifier representing a user identity. After the user identifier is queried, the participant can perform joint privacy-preserving computation through interaction with another participant, to determine the common user of all the participants. Specifically, the participant can determine the common user of all the participants based on the target data through interaction with another participant and private set intersection (PSI for short).

[0071] Then, the participant can push specified information to the common user, to provide a better service for the user. The specified information is information preset by the participant, and the specified information can be a text or an image. The specified information can be information such as a message and an activity related to a service provided by the participant.

[0072] Certainly, the participant can further perform risk control on the common user, to determine whether the common user has a risk. Specifically, the participant can determine service data of the common user, and input the service data into a pre-trained risk identification model, to determine a risk identification result of the common user. In addition, when the risk identification result is that there is a risk, risk control is performed on the common user.

[0073] In one or more embodiments of this specification, the participant can be deployed with

a data query engine, and the data query engine can perform data query based on the to-be-queried statement. Therefore, the data query method shown in FIG. 1 can also be performed by the data query engine deployed on the participant. A specific process is shown in FIG. 1. Details are omitted here for simplicity.

[0074] The specification also provides a computer-readable non-transitory storage medium, where the storage medium stores a computer program, which, when executed by a processor, can be used to perform one or more steps of one or more methods described or illustrated here or provides functionality described or illustrated here. Here, the computer-readable non-transitory storage medium or media may include one or more semiconductor-based or other integrated circuits (ICs) (such, as for example, field-programmable gate arrays (FPGAs) or application-specific ICs (ASICs)), hard disk drives (HDDs), hybrid hard drives (HHDs), optical discs, optical disc drives (ODDs), magneto-optical discs, magneto-optical drives, floppy diskettes, floppy disk drives (FDDs), magnetic tapes, solid-state drives (SSDs), RAM-drives, SECURE DIGITAL cards or drives, any other suitable computer-readable non-transitory storage media, or any suitable combination of two or more of these, where appropriate. A computer-readable non-transitory storage medium may be volatile, non-volatile, or a combination of volatile and non-volatile, where appropriate.

[0075] The specification also provides a computer system. FIG. 6 illustrates an example computer system 600. In particular embodiments, one or more computer systems 600 perform one or more steps of one or more methods described or illustrated here. In particular embodiments, one or more computer systems 600 provide functionality described or illustrated here. In particular embodiments, software running on one or more computer systems 600 performs one or more steps of one or more methods described or illustrated here or provides functionality described or illustrated here. Particular embodiments include one or more portions of one or more computer systems 600. Here, reference to a computer system may encompass a computing device, and vice versa, where appropriate. Moreover, reference to a computer system may encompass one or more computer systems, where appropriate.

[0076] This disclosure contemplates any suitable number of computer systems 600. This disclosure contemplates computer system 600 taking any suitable physical form. As example and not by way of limitation, computer system 600 may be an embedded computer system, a system-on-chip (SOC), a single-board computer system (SBC) (such as, for example, a computer-on-

module (COM) or system-on-module (SOM)), a desktop computer system, a laptop or notebook computer system, an interactive kiosk, a mainframe, a mesh of computer systems, a mobile telephone, a personal digital assistant (PDA), a server, a tablet computer system, or a combination of two or more of these. Where appropriate, computer system 600 may include one or more computer systems 600; be unitary or distributed; span multiple locations; span multiple machines; span multiple data centers; or reside in a cloud, which may include one or more cloud components in one or more networks. Where appropriate, one or more computer systems 600 may perform without substantial spatial or temporal limitation one or more steps of one or more methods described or illustrated here. As an example and not by way of limitation, one or more computer systems 600 may perform in real time or in batch mode one or more steps of one or more methods described or illustrated here. One or more computer systems 600 may perform at different times or at different locations one or more steps of one or more methods described or illustrated here, where appropriate.

[0077] In particular embodiments, computer system 600 includes a processor 602, memory 604, storage 606, an input/output (I/O) interface 608, a communication interface 610, and a bus 612. Although this disclosure describes and illustrates a particular computer system having a particular number of particular components in a particular arrangement, this disclosure contemplates any suitable computer system having any suitable number of any suitable components in any suitable arrangement.

[0078] In particular embodiments, processor 602 includes hardware for executing instructions, such as those making up a computer program. As an example and not by way of limitation, to execute instructions, processor 602 may retrieve (or fetch) the instructions from an internal register, an internal cache, memory 604, or storage 606; decode and execute them; and then write one or more results to an internal register, an internal cache, memory 604, or storage 606. In particular embodiments, processor 602 may include one or more internal caches for data, instructions, or addresses. This disclosure contemplates processor 602 including any suitable number of any suitable internal caches, where appropriate. As an example and not by way of limitation, processor 602 may include one or more instruction caches, one or more data caches, and one or more translation lookaside buffers (TLB s). Instructions in the instruction caches may be copies of instructions in memory 604 or storage 606, and the instruction caches may speed up retrieval of those instructions by processor 602. Data in the data caches may be copies of data in memory 604

or storage 606 for instructions executing at processor 602 to operate on; the results of previous instructions executed at processor 602 for access by subsequent instructions executing at processor 602 or for writing to memory 604 or storage 606; or other suitable data. The data caches may speed up read or write operations by processor 602. The TLBs may speed up virtual-address translation for processor 602. In particular embodiments, processor 602 may include one or more internal registers for data, instructions, or addresses. This disclosure contemplates processor 602 including any suitable number of any suitable internal registers, where appropriate. Where appropriate, processor 602 may include one or more arithmetic logic units (ALUs); be a multi-core processor; or include one or more processors 602. Although this disclosure describes and illustrates a particular processor, this disclosure contemplates any suitable processor.

[0079] In particular embodiments, memory 604 includes main memory for storing instructions for processor 602 to execute or data for processor 602 to operate on. As an example and not by way of limitation, computer system 600 may load instructions from storage 606 or another source (such as, for example, another computer system 600) to memory 604. Processor 602 may then load the instructions from memory 604 to an internal register or internal cache. To execute the instructions, processor 602 may retrieve the instructions from the internal register or internal cache and decode them. During or after execution of the instructions, processor 602 may write one or more results (which may be intermediate or final results) to the internal register or internal cache. Processor 602 may then write one or more of those results to memory 604. In particular embodiments, processor 602 executes only instructions in one or more internal registers or internal caches or in memory 604 (as opposed to storage 606 or elsewhere) and operates only on data in one or more internal registers or internal caches or in memory 604 (as opposed to storage 606 or elsewhere). One or more memory buses (which may each include an address bus and a data bus) may couple processor 602 to memory 604. Bus 612 may include one or more memory buses, as described below. In particular embodiments, one or more memory management units (MMUs) reside between processor 602 and memory 604 and facilitate accesses to memory 604 requested by processor 602. In particular embodiments, memory 604 includes random access memory (RAM). This RAM may be volatile memory, where appropriate. Where appropriate, this RAM may be dynamic RAM (DRAM) or static RAM (SRAM). Moreover, where appropriate, this RAM may be single-ported or multi-ported RAM. This disclosure contemplates any suitable RAM. Memory 604 may include one or more memories 604, where appropriate. Although this disclosure

describes and illustrates particular memory, this disclosure contemplates any suitable memory.

[0080] In particular embodiments, storage 606 includes mass storage for data or instructions. As an example and not by way of limitation, storage 606 may include a hard disk drive (HDD), a floppy disk drive, flash memory, an optical disc, a magneto-optical disc, magnetic tape, or a Universal Serial Bus (USB) drive or a combination of two or more of these. Storage 606 may include removable or non-removable (or fixed) media, where appropriate. Storage 606 may be internal or external to computer system 600, where appropriate. In particular embodiments, storage 606 is non-volatile, solid-state memory. In particular embodiments, storage 606 includes read-only memory (ROM). Where appropriate, this ROM may be mask-programmed ROM, programmable ROM (PROM), erasable PROM (EPROM), electrically erasable PROM (EEPROM), electrically alterable ROM (EAROM), or flash memory or a combination of two or more of these. This disclosure contemplates mass storage 606 taking any suitable physical form. Storage 606 may include one or more storage control units facilitating communication between processor 602 and storage 606, where appropriate. Where appropriate, storage 606 may include one or more storages 606. Although this disclosure describes and illustrates particular storage, this disclosure contemplates any suitable storage.

[0081] In particular embodiments, I/O interface 608 includes hardware, software, or both, providing one or more interfaces for communication between computer system 600 and one or more I/O devices. Computer system 600 may include one or more of these I/O devices, where appropriate. One or more of these I/O devices may enable communication between a person and computer system 600. As an example and not by way of limitation, an I/O device may include a keyboard, keypad, microphone, monitor, mouse, printer, scanner, speaker, still camera, stylus, tablet, touch screen, trackball, video camera, another suitable I/O device or a combination of two or more of these. An I/O device may include one or more sensors. This disclosure contemplates any suitable I/O devices and any suitable I/O interfaces 608 for them. Where appropriate, I/O interface 608 may include one or more device or software drivers enabling processor 602 to drive one or more of these I/O devices. I/O interface 608 may include one or more I/O interfaces 608, where appropriate. Although this disclosure describes and illustrates a particular I/O interface, this disclosure contemplates any suitable I/O interface.

[0082] In particular embodiments, communication interface 610 includes hardware, software, or both providing one or more interfaces for communication (such as, for example, packet-based

communication) between computer system 600 and one or more other computer systems 600 or one or more networks. As an example and not by way of limitation, communication interface 610 may include a network interface controller (NIC) or network adapter for communicating with an Ethernet or other wire-based network or a wireless NIC (WNIC) or wireless adapter for communicating with a wireless network, such as a WI-FI network. This disclosure contemplates any suitable network and any suitable communication interface 610 for it. As an example and not by way of limitation, computer system 600 may communicate with an ad hoc network, a personal area network (PAN), a local area network (LAN), a wide area network (WAN), a metropolitan area network (MAN), or one or more portions of the Internet or a combination of two or more of these. One or more portions of one or more of these networks may be wired or wireless. As an example, computer system 600 may communicate with a wireless PAN (WPAN) (such as, for example, a BLUETOOTH WPAN), a WI-FI network, a WI-MAX network, a cellular telephone network (such as, for example, a Global System for Mobile Communications (GSM) network), or other suitable wireless network or a combination of two or more of these. Computer system 600 may include any suitable communication interface 610 for any of these networks, where appropriate. Communication interface 610 may include one or more communication interfaces 610, where appropriate. Although this disclosure describes and illustrates a particular communication interface, this disclosure contemplates any suitable communication interface.

[0083] In particular embodiments, bus 612 includes hardware, software, or both coupling components of computer system 600 to each other. As an example and not by way of limitation, bus 612 may include an Accelerated Graphics Port (AGP) or other graphics bus, an Enhanced Industry Standard Architecture (EISA) bus, a front-side bus (FSB), a HYPERTRANSPORT (HT) interconnect, an Industry Standard Architecture (ISA) bus, an INFINIBAND interconnect, a low-pin-count (LPC) bus, a memory bus, a Micro Channel Architecture (MCA) bus, a Peripheral Component Interconnect (PCI) bus, a PCI-Express (PCIe) bus, a serial advanced technology attachment (SATA) bus, a Video Electronics Standards Association local (VLB) bus, or another suitable bus or a combination of two or more of these. Bus 612 may include one or more buses 612, where appropriate. Although this disclosure describes and illustrates a particular bus, this disclosure contemplates any suitable bus or interconnect.

[0084] The above-mentioned descriptions are merely embodiments of this specification, and are not intended to limit this specification. A person skilled in the art can make various changes

and changes to this specification. Any modification, equivalent replacement, improvement, etc. made without departing from the spirit and principle of this specification shall fall within the scope of the claims in this application.

CLAIMS

What is claimed is:

1. A data query method, wherein the method is applied to any participant that performs secure multi-party computation, and the method comprises:

determining a to-be-queried statement, and determining a conditional statement comprised in the to-be-queried statement;

determining a cache tree of the participant, wherein the cache tree comprises several nodes, an edge between the nodes corresponds to a query condition, and each node corresponds to cache data pre-cached by the participant;

determining, based on the query condition corresponding to the edge between the nodes, a node hit by the conditional statement, and using the node as a first node; and

determining target data corresponding to the to-be-queried statement based on cache data corresponding to the first node.

2. The method according to claim 1, wherein the determining, based on the query condition comprised by the edge between the nodes, a node hit by the conditional statement, and using the node as a first node specifically comprises:

sequentially determining each node of the cache tree as a query node starting from a root node of the cache tree, determining a child node whose parent node is the query node, and using the child node as a second node;

determining a query condition of an edge between the second node and the query node, and using the query condition as a first condition;

determining whether a query range corresponding to the conditional statement is a subset of a query range corresponding to the first condition; and

if yes, using the second node as a new query node, and until the query range corresponding to the conditional statement is not a subset of the query range corresponding to the first condition, using the query node as the first node hit by the conditional statement.

3. The method according to claim 1, wherein the determining target data corresponding to the to-be-queried statement based on cache data corresponding to the first node specifically comprises:

determining, from the cache tree, a child node whose parent node is the first node;

determining cache data corresponding to the child node as sub-data; and

using the cache data corresponding to the first node and the sub-data as the target data corresponding to the to-be-queried statement.

4. The method according to claim 1, wherein the method further comprises:

determining a prestored data file when no node is hit by the conditional statement in the cache tree; and

querying data corresponding to the to-be-queried statement from the data file, and using the data as the target data.

5. The method according to claim 1, wherein the determining target data corresponding to the to-be-queried statement based on cache data corresponding to the first node specifically comprises:

determining a parent node of the first node in the cache tree, determining a query condition corresponding to an edge between the parent node and the first node, and using the query condition as a second condition;

using the cache data corresponding to the first node as the target data corresponding to the to-be-queried statement when the second condition is the same as the conditional statement; and

when a query range of the conditional statement is a proper subset of a query range of the second condition, filtering the cache data corresponding to the first node based on the conditional statement, and using filtered data as the target data corresponding to the to-be-queried statement.

6. The method according to claim 1, wherein the conditional statement comprises at least one subcondition; and

the method further comprises:

sequentially determining each node in the cache tree as an update node starting from a root node of the cache tree, determining a child node whose parent node is the update node, determining a query condition corresponding to an edge between the update node and the child node, and using the query condition as a third condition;

determining whether at least one subcondition in the conditional statement is the same as the third condition; and

if yes, using the child node as a new update node, and until the third condition is different from all subconditions in the conditional statement, constructing a first child node whose parent node is the update node, determining cache data corresponding to the first child node as the target data, and determining the query condition corresponding to the edge between the update node and the first child node as the conditional statement.

7. The method according to claim 6, wherein the method further comprises:

determining cache data corresponding to the update node; and

deleting the target data from the cache data, to update the update node.

8. The method according to claim 6, wherein the method further comprises:

determining another child node that is in the cache tree and whose parent node is the update node, determining a query condition corresponding to an edge between the update node and the another child node, and using the query condition as a fourth condition, wherein the another child node is a node other than the first child node;

when a query range of the fourth condition is a proper subset of a query range of the conditional statement, combining the first child node and the another child node, to obtain a combined child node;

combining the target data and cache data corresponding to the another child node, and using combined data as cache data corresponding to the combined child node; and

determining a query condition corresponding to an edge between the update node and the combined child node as the conditional statement.

9. The method according to claim 1, wherein the method further comprises:

deleting cache data corresponding to a leaf node in the cache tree based on a specified period.

10. A computer system, comprising a storage, a processor, and a computer program that is stored in the storage and that is capable of running on the processor, wherein when the processor executes the program, the following steps are implemented:

determining a to-be-queried statement, and determining a conditional statement comprised in the to-be-queried statement;

determining a cache tree of a participant, wherein the cache tree comprises several nodes, an edge between the nodes corresponds to a query condition, and each node corresponds to cache data pre-cached by the participant;

determining, based on the query condition corresponding to the edge between the nodes, a node hit by the conditional statement, and using the node as a first node; and

determining target data corresponding to the to-be-queried statement based on cache data corresponding to the first node.

11. The computer system according to claim 10, wherein the determining, based on the query condition comprised by the edge between the nodes, a node hit by the conditional statement, and

using the node as a first node specifically comprises:

sequentially determining each node of the cache tree as a query node starting from a root node of the cache tree, determining a child node whose parent node is the query node, and using the child node as a second node;

determining a query condition of an edge between the second node and the query node, and using the query condition as a first condition;

determining whether a query range corresponding to the conditional statement is a subset of a query range corresponding to the first condition; and

if yes, using the second node as a new query node, and until the query range corresponding to the conditional statement is not a subset of the query range corresponding to the first condition, and using the query node as the first node hit by the conditional statement.

12. The computer system according to claim 10, wherein the determining target data corresponding to the to-be-queried statement based on cache data corresponding to the first node specifically comprises:

determining, from the cache tree, a child node whose parent node is the first node;

determining cache data corresponding to the child node as sub-data; and

using the cache data corresponding to the first node and the sub-data as the target data corresponding to the to-be-queried statement.

13. The computer system according to claim 10, wherein when the processor executes the program, the following steps are further implemented:

determining a prestored data file when no node is hit by the conditional statement in the cache tree; and

querying data corresponding to the to-be-queried statement from the data file, and using the data as the target data.

14. The computer system according to claim 10, wherein the determining target data corresponding to the to-be-queried statement based on cache data corresponding to the first node specifically comprises:

determining a parent node of the first node in the cache tree, determining a query condition corresponding to an edge between the parent node and the first node, and using the query condition as a second condition;

using the cache data corresponding to the first node as the target data corresponding to the to-

be-queried statement when the second condition is the same as the conditional statement; and

when a query range of the conditional statement is a proper subset of a query range of the second condition, filtering the cache data corresponding to the first node based on the conditional statement, and using filtered data as the target data corresponding to the to-be-queried statement.

15. The computer system according to claim 11, wherein the conditional statement comprises at least one subcondition; and

the system further comprises:

sequentially determining each node in the cache tree as an update node starting from a root node of the cache tree, determining a child node whose parent node is the update node, determining a query condition corresponding to an edge between the update node and the child node, and using the query condition as a third condition;

determining whether at least one subcondition in the conditional statement is the same as the third condition; and

if yes, using the child node as a new update node, and until the third condition is different from all subconditions in the conditional statement, constructing a first child node whose parent node is the update node, determining cache data corresponding to the first child node as the target data, and determining the query condition corresponding to the edge between the update node and the first child node as the conditional statement.

16. The computer system according to claim 15, wherein when the processor executes the program, the following steps are further implemented:

determining cache data corresponding to the update node; and

deleting the target data from the cache data, to update the update node.

17. The computer system according to claim 15, wherein when the processor executes the program, the following steps are further implemented:

determining another child node that is in the cache tree and whose parent node is the update node, determining a query condition corresponding to an edge between the update node and the another child node, and using the query condition as a fourth condition, wherein the another child node is a node other than the first child node;

when a query range of the fourth condition is a proper subset of a query range of the conditional statement, combining the first child node and the another child node, to obtain a combined child node;

combining the target data and cache data corresponding to the another child node, and using combined data as cache data corresponding to the combined child node; and

determining a query condition corresponding to an edge between the update node and the combined child node as the conditional statement.

18. The computer system according to claim 10, wherein when the processor executes the program, the following step is further implemented:

deleting cache data corresponding to a leaf node in the cache tree based on a specified period.

19. A computer-readable nonvolatile storage medium, wherein the storage medium stores a computer program, and when the computer program is executed by a processor, the method according to any one of claims 1 to 9 is implemented.