

QUERY OPTIMIZATION METHOD AND APPARATUS

TECHNICAL FIELD

[0001] The present specification relates to the field of Internet technologies, and in particular, to a query optimization method and apparatus.

5

BACKGROUND

[0002] In an online query scenario, there is often a query requirement of high concurrency, high throughput, and low delay. For example, in an online transaction processing (OLTP) query scenario, there may be tens of thousands of batch query requests per second. Each batch query request may include dozens or
10 even hundreds of query tasks, and a query response time is required to be within 10 ms. In such a scenario of a large number of query tasks, the query tasks need to be optimized to improve query efficiency.

[0003] In the related query optimization technology, a query logic of a single query task can be optimized only in a range of the query task. However, many
15 repeated query logics still exist in different query tasks. Therefore, query optimization in the range of the single query task still has low query efficiency.

SUMMARY

[0004] At least one embodiment of the present specification provides a query optimization method to improve query efficiency of a query task.

20 [0005] According to a first aspect, a query optimization method is provided, the method including: obtaining a plurality of to-be-optimized query tasks, each query task including at least one subtask, and when a same query task includes at least two subtasks, a query dependence relationship existing between the at least two subtasks; combining the plurality of query tasks based on the query
25 dependence relationship to obtain one combined task; and merging at least two subtasks in different query tasks based on the combined task to obtain a first merged subtask, the first merged subtask being executed to obtain query results of the at least two subtasks in the different query tasks.

[0006] According to a second aspect, a query optimization apparatus is provided, the apparatus including: an acquisition module, configured to obtain a plurality of to-be-optimized query tasks, each query task including at least one subtask, and when a same query task includes at least two subtasks, a query dependence relationship existing between the at least two subtasks; a combination module, configured to combine the plurality of query tasks based on the query dependence relationship to obtain one combined task; and a merging module, configured to merge at least two subtasks in different query tasks based on the combined task to obtain a first merged subtask, the first merged subtask being executed to obtain query results of the at least two subtasks in the different query tasks.

[0007] According to a third aspect, a computer device is provided, including a memory, a processor, and a computer program that is stored in the memory and that can run on the processor, the processor implementing the query optimization method in any embodiment of the present specification when executing the program.

[0008] According to a fourth aspect, a computer readable storage medium is provided, the computer readable storage medium storing a computer program, and steps of the query optimization method in any embodiment of the present specification being implemented when the program is executed by a processor.

[0009] It can be learned from the above technical solution that, in the at least one embodiment of the present specification, a plurality of query tasks are combined to obtain a combined task, and subtasks in different query tasks are merged into one task based on the complete combined task. Because subtasks optimized based on the combined task belong to different query tasks, in this solution, query optimization in ranges of a plurality of different query tasks is implemented. Compared with an optimization method of a single query task, repeated query logics in different query tasks can be merged and optimized, thereby improving query efficiency. Further, because tasks can be merged in the ranges of the plurality of different query task, a high throughput of hundreds of thousands or even hundreds of thousands of batch queries per second is possible.

[0010] It should be understood that the general descriptions above and the detailed descriptions below are merely examples for description, and cannot limit the present specification.

BRIEF DESCRIPTION OF DRAWINGS

[0011] FIG. 1 is a flowchart illustrating a query optimization method according to an example embodiment;

5 [0012] FIG. 2 is a schematic diagram illustrating a combined task according to an example embodiment;

[0013] FIG. 3 is a schematic diagram illustrating a query optimization apparatus according to an example embodiment;

[0014] FIG. 4 is a schematic diagram illustrating another query optimization apparatus according to an example embodiment;

10 [0015] FIG. 5 is a schematic diagram illustrating another query optimization apparatus according to an example embodiment;

[0016] FIG. 6 is a schematic diagram illustrating another query optimization apparatus according to an example embodiment; and

15 [0017] FIG. 7 is a schematic diagram illustrating another query optimization apparatus according to an example embodiment.

DETAILED DESCRIPTION

[0018] Embodiments are described in detail herein, and examples of which are presented in the accompanying drawings. When the following description refers to the accompanying drawings, unless specified otherwise, same numbers in
20 different accompanying drawings represent same or similar elements. Implementations described in the following example embodiments do not represent all implementations consistent with the present specification. On the contrary, the implementations are merely examples of apparatuses and methods that are described in the appended claims in details and consistent with some
25 aspects of the present specification.

[0019] The terms used in the present specification are merely for illustrating specific embodiments, and are not intended to limit the present specification. The terms "a" and "the" of singular forms used in the present specification and the appended claims are also intended to include plural forms, unless otherwise
30 specified in the context clearly. It should be further understood that the term

"and/or" used in the present specification indicates and includes any or all possible combinations of one or more associated listed items.

[0020] It should be understood that although terms "first", "second", "third", etc. may be used in the present specification to describe various types of information, the information is not limited to the terms. These terms are merely used to differentiate between information of the same type. For example, without departing from the scope of the present specification, first information can also be referred to as second information, and similarly, the second information can be referred to as the first information. Depending on the context, for example, the word "if" used herein can be explained as "while", "when", or "in response to determining".

[0021] In an online query scenario, there is often a query requirement of high concurrency, high throughput, and low delay. For example, in an online transaction processing (OLTP) query scenario during promotion of online retailers, there may be tens of thousands of batch query requests per second. Each batch query request may include dozens or even hundreds of query tasks, and a query response time is required to be within 10 ms. In such a scenario of a large number of query tasks, the query tasks need to be optimized to improve query efficiency.

[0022] In the related query optimization technology, a query logic of a single query task can be optimized only in a range of the query task. For example, two same query logics in a same query task are merged into one query logic, thereby avoiding repeated execution of a same query logic. However, many repeated query logics still exist in different query tasks. When different query tasks are executed, many same query logics are repeatedly executed. It can be learned that, in a method for optimizing a single query task, query efficiency is still low and even query resources are wasted.

[0023] Therefore, the present specification discloses a query optimization method to merge a plurality of query tasks into one combined task. In ranges of different query tasks included in the combined task, query logics that belong to different query tasks can also be optimized.

[0024] It should be noted that the query optimization method provided in the present specification can be applied to various online queries that can include but

are not limited to an OLTP query or an online analytical processing (OLAP) query.

5 [0025] To make the query optimization method provided in the present specification clearer, the following describes the solution execution process provided in the present specification in detail with reference to the accompanying drawings and the embodiments.

[0026] FIG. 1 is a flowchart illustrating a query optimization method according to an embodiment of the present specification. As shown in FIG. 1, the process includes the following steps.

10 [0027] Step 101: Obtain a plurality of to-be-optimized query tasks, each query task including at least one subtask, and when a same query task includes at least two subtasks, a query dependence relationship existing between the at least two subtasks.

15 [0028] In this embodiment, the query task is a process of obtaining a final query result from a data source based on a query condition. For example, when the query condition is "male name", a "name" that meets the condition can be found from a corresponding data source based on the query condition, and used as a query result. This is a complete query task.

20 [0029] Each query task includes different query logics, and a query result of the query task can be finally obtained by executing the query logics in the query task. In this embodiment, different query logics are represented by using subtasks in a query task, and each query task can include one or more subtasks.

25 [0030] For example, when the query condition is "male name", the query task can include only one subtask: a name that meets the condition is found from a data source based on the query condition "male". That is, a query result of the query task can be obtained by querying the subtask.

30 [0031] For example, when the query condition is "name of a male taller than 175 cm", the query task can include two subtasks: subtask 1: name information that meets a height condition needs to be found from a data source based on the query condition "taller than 175 cm", and is used as a query result of subtask 1; and subtask 2: a name that meets a gender condition needs to be found from the query result of subtask 1 based on the query condition "male".

[0032] That is, a query result of the query task needs to be obtained after the two subtasks are both queried. Based on a same principle, it can be understood

that when a query condition is more complex, a query task may include three or more subtasks.

[0033] When a same query task includes two or more subtasks, a query dependence relationship exists between the plurality of subtasks. For example, if execution of one subtask in a query task needs to depend on an execution result of another subtask in the query task, a query dependence relationship exists between the two subtasks.

[0034] FIG. 2 illustrates a plurality of query tasks in a form of a directed acyclic graph (DAG): query task 1, query task 2, and query task 3. Query task 1 is used as an example for description. Node "Input1" represents a query entry in which a query condition of a query task including the node can be input. Node "A" represents a query data source corresponding to the query task including the node, such as an HBASE database or a MySQL database. Node "Join1" represents a query result obtained through query based on the query condition in "Input1" and the query data source in "A".

[0035] Referring to FIG. 2, query task 1 includes only subtask a, and subtask a is obtained by connecting the three nodes "Input1", "A", and "Join1". When subtask a is executed, query can be performed based on the query condition in Input1 and the query data source in A, to obtain a query result of subtask a. Because query task 1 includes only subtask a, the query result of subtask a is a query result of query task 1.

[0036] Referring to FIG. 2, query task 2 includes two subtasks: subtask b and subtask c.

[0037] Subtask b is obtained by connecting three nodes: "Input2", "B", and "Join2". Executing subtask b can obtain a query result of subtask b. A process of obtaining the query result of subtask b is the same as the process of obtaining the query result of subtask a, and details are omitted for simplicity.

[0038] Subtask c is obtained by connecting three nodes: "Join2", "C", and "Join3". "Join2" further exists as a query condition of subtask c while representing the query result of subtask b. When subtask c is executed, query can be performed based on the query result "Join2" of subtask b and a query data source in C, to obtain a query result of subtask c. Because no other subtask needs to be executed after subtask c, the obtained query result of subtask c is a query

result of the query task 2. Based on a same principle as query task 2, query task 3 includes subtask d and subtask e, and subtask e depends on subtask d.

[0039] For query task 2 shown in FIG. 2, because execution of subtask c depends on the query result of subtask b as the query condition, a query dependence relationship exists between subtask b and subtask c. In addition, the query dependence relationship is specifically that subtask c depends on subtask b.

[0040] Step 102: Combine the plurality of query tasks based on the query dependence relationship to obtain one combined task.

[0041] In the present step, a plurality of different query tasks can be combined based on a query dependence relationship between subtasks in each query task, to obtain a complete combined task.

[0042] A to-be-connected subtask can be determined based on the query dependence relationship between the subtasks in each query task. FIG. 2 is used as an example. Because query task 1 includes only subtask a, subtask a can be used as a to-be-connected subtask. Based on the query dependence relationship between subtask b and subtask c in query task 2, subtask c can be used as a to-be-connected subtask because subtask c is the last subtask when query task 2 is executed. Based on the query dependence relationship between subtask d and subtask e in query task 3, subtask e can be used as a to-be-connected subtask.

[0043] The to-be-connected subtasks are connected to connect the query tasks together to obtain a complete combined task. For example, query task 1, query task 2, and query task 3 can be connected by using two nodes "JOIN1" and "JOIN2", to obtain complete combined task 1. Nodes "JOIN1" and "JOIN2" do not affect final query results of the connected different query tasks, that is, a query result of each query task obtained based on combined task 1 is the same as a query result obtained by separately executing each query task.

[0044] In an example, each subtask includes a query data source, and query data sources can be data sources of different physical architectures. FIG. 2 is used as an example. Query data source "A" of subtask a can be an HBASE database. Query data source "B" of subtask b can be a MySQL database. Query data source "C" of subtask c can be an HBASE database. Between different subtasks in a same query task, query data sources can be heterogeneous data sources. Alternatively, between subtasks in different query tasks, query data sources can be heterogeneous data sources. In this embodiment, a physical

structure of a data source in a subtask is not limited, and a query data source in any subtask can be a heterogeneous data source.

[0045] In an example, before the combining the plurality of query tasks to obtain the one combined task, the method further includes: determining a first query logic from a query task based on a predetermined determining condition, the first query logic including at least one subtask in the same query task; and storing a query result of the first query logic. After the combining the plurality of query tasks to obtain the one combined task, the method further includes: if determining that a query logic that is the same as the first query logic exists in the combined task, using the query result of the first query logic as a query result of the query logic that is the same as the first query logic.

[0046] In the above example, before a subtask of a query task in the combined task is optimized, a common query logic can be determined in the same query task as the first query logic based on the predetermined determining condition. The predetermined determining condition can be a number of execution times, an execution frequency, etc. For example, if a number of execution times of a certain query logic is greater than a predetermined number of times, it can be determined that the query logic is a common query logic and is used as the first query logic. The above query logic can be one subtask or a plurality of subtasks in a same query task that have a query dependence relationship.

[0047] When a subtask in the combined task is executed, if determining that a query logic that is the same as the first query logic exists, the query result of the first query logic can be directly obtained without repeatedly executing the same query logic.

[0048] Combined task 1 shown in FIG. 2 is used as an example. Before a subtask in combined task 1 is executed, the first query logic can be first determined. For example, subtask b can be used as the first query logic, or query task 2 including subtask b and subtask c can be used as the first query logic. If the determining condition is that the number of execution times is greater than the predetermined number of times, subtask b is used as the first query logic when a number of execution times of subtask b is greater than the predetermined number of times to execute subtask b and store the query result.

[0049] When a subtask in combined task 1 is executed, if determining that a query logic "subtask b" that is the same as the first query logic exists in

combined task 1, the stored query result of the first query logic can be used as the query result of subtask b without repeatedly executing a query process of subtask b.

5 [0050] Step 103: Merge at least two subtasks in different query tasks based on the combined task to obtain a first merged subtask, the first merged subtask being executed to obtain query results of the at least two subtasks in the different query tasks.

10 [0051] In the related query optimization technology, only a single query task can be optimized. FIG. 2 is used as an example. In the related technology, query optimization can be performed only on query task 1, query optimization can be performed only on query task 2, or query optimization can be performed only on query task 3.

15 [0052] In the present step, at least two subtasks that belong to different query tasks can be merged based on the combined task obtained by combining different query tasks, to obtain the first merged subtask. As such, query results of a plurality of subtasks can be obtained by performing the first merged subtask once. A method for merging subtasks can be merging a plurality of same subtasks into one task, to remove a repeated subtask. Alternatively, the plurality of subtasks can be merged into one task based on a same part of the plurality of subtasks. For example, subtasks whose storage paths have a same part can be merged into one executable task.

20 [0053] Combined task 1 shown in FIG. 2 is used as an example. In the present step, subtasks that belong to different query tasks can be merged based on combined task 1. For example, if subtask a in query task 1 is the same as subtask b in query task 2, subtask a and subtask b can be merged into a first merged subtask. As such, the query results of both subtask a and subtask b can be obtained by executing the first merged subtask only once. Specifically, for example, subtask a can be executed only once, or subtask b can be executed only once. A same determining condition of the two subtasks can be: query conditions and query data sources that are included in the two subtasks are the same. In this method, repeatedly executing a query process of a subtask is removed, and an effect of query optimization is implemented. In addition, in the "de-duplication optimization" process, because subtask a and subtask b belong to different query tasks, this optimization method implements cross-query task optimization.

[0054] It should be noted that in the combined task, in addition to the above "de-duplication optimization" method, there are many specific methods for merging subtasks in different query tasks. No limitation is imposed in this embodiment. For example, if subtask a is the same as subtask c, subtask a and
5 subtask c can be merged to obtain a first merged subtask. The query results of both subtask a and subtask c are obtained by executing the first merged subtask.

[0055] In the query optimization method in this embodiment, subtasks in the combined task for optimization belong to different query tasks, cross-query task optimization is implemented based on the combined task in this method.
10 Compared with optimization of a single query task, query efficiency is improved. Further, because cross-query task optimization is implemented in the method, repeated queries existing in different query tasks can be greatly reduced, and query resources can be saved.

[0056] In an example, the merging the at least two subtasks in different query
15 tasks to obtain the first merged subtask includes: determining a non-dependent subtask in each query task based on the query dependence relationship to form a first task set of the combined task; and merging at least two subtasks in the first task set to obtain the first merged subtask.

[0057] In the above embodiment, a subtask that can be executed without
20 depending on another subtask needs to be determined from each query task based on a query dependence relationship between subtasks in the query task, and subtasks determined from the query tasks form the first task set. Combined task 1 shown in FIG. 2 is used as an example. Subtask a in query task 1 can be executed without depending on another subtask. Subtask b in query task 2 does not depend
25 on another subtask. Subtask d in query task 3 does not depend on another subtask. Therefore, subtask a, subtask b, and subtask d can form a set to obtain the first task set of combined task 1.

[0058] After the first task set of the combined task is determined, optimization
30 can be performed based on a feature of the subtask in the first task set because all the subtasks in the first task set are non-dependent subtasks. In this embodiment, the at least two subtasks in the first task set can be merged to obtain the first merged subtask. As such, query results of a plurality of subtasks used for combination can be obtained by executing the first merged subtask once.

[0059] In this embodiment, because the subtasks in the first task set belong to different query tasks, the subtasks of the first task set are optimized, thereby implementing an effect of cross-query task optimization. Therefore, an optimization effect and query efficiency are improved. In addition, cross-query task optimization can reduce repeated query logics in different query tasks and save query resources.

[0060] In an example, after the forming the first task set of the combined task, the method further includes: concurrently executing the subtasks in the first task set; and placing, into a cache, a query result of a subtask whose query duration meets a predetermined condition.

[0061] In the above embodiment, a plurality of subtasks in the first task set can be executed concurrently. Therefore, in a process of concurrently executing the plurality of subtasks, query duration used by the subtasks is also different because specific query processes of the subtasks are different. Query duration of one or more subtasks may be longer (longer than query duration of another subtask), and query duration of concurrent execution may be affected by the one or more query tasks. In the above embodiment, the query result of the subtask whose query duration meets the predetermined condition can be placed into the cache to accelerate query. The predetermined condition can include but is not limited to a query duration threshold. For example, the predetermined condition can be: "query duration is longer than 10 ms".

[0062] In an example, each subtask includes a query condition and a query data source. The merging the at least two subtasks in the first task set to obtain the first merged subtask includes: merging at least two subtasks in the first task set that have a same query condition and query data source, to obtain the first merged subtask.

[0063] In the above embodiment, each subtask includes a query condition and a query data source.

[0064] As shown in FIG. 2, subtask a in combined task 1 includes the query condition input in node "Input1" and the query data source in node "A". Subtask b includes the query condition input in node "Input2" and the query data source in node "B". Subtask c can use the query result "Join2" of subtask b as the query condition, and further includes the query data source in node "C".

[0065] In this embodiment, subtasks that have a same query condition and query data source can be determined from the subtasks in the first task set for de-duplication.

[0066] If query conditions and query data sources of two subtasks are the same, it can be considered that the two subtasks are the same, and a same query result is obtained by executing the two subtasks. Therefore, a plurality of subtasks in the first task set that have a same query condition and query data source can be merged to obtain the first merged subtask. Query results of the plurality of subtasks can be obtained by executing the first merged subtask once. The first merged subtask can be one of a plurality of subtasks used for merging. Therefore, only one of the subtasks can be executed after the merging, to avoid repeatedly executing a same query logic.

[0067] FIG. 2 is used as an example. In the first task set, de-duplication can be performed if the query condition input in "Input1" of subtask a is the same as the query condition input in "Input2" of subtask b and query data source "A" of subtask a is the same as query data source "B" of subtask b. That is, subtask a and subtask b can be merged to obtain the first merged subtask, for example, subtask a is used as the first merged subtask. As such, the query results of both subtask a and subtask b are obtained by executing subtask a once.

[0068] In this embodiment, when the first task set is determined from the combined task, de-duplication optimization can be performed on the subtasks in the first task set. Because the subtasks in the first task set belong to different query tasks, in this de-optimization method, cross-query task optimization is implemented, repeated query logics in different query tasks are reduced, query efficiency is improved, and query resources are saved.

[0069] In an example, each subtask includes a query data source. The merging the at least two subtasks in the first task set to obtain the first merged subtask includes: determining a plurality of subtasks with a same query data source from the first task set; and merging at least two subtasks whose query paths have a same part in the plurality of subtasks with the same query data source, to obtain the first merged subtask.

[0070] In the above embodiment, when the first task set is determined from the combined task, subtasks whose storage paths have a same part can be merged.

[0071] FIG. 2 is used as an example. In the first task set, assume that the query data source of subtask a is an HBASE database, and the query data source of subtask b is also an HBASE database. In this embodiment, it can be determined from the first task set that subtask a and subtask b are subtasks with a same query data source.

[0072] For example, the HBASE database is used as the query data source. The following illustrates five query tasks:

[0073] Query1: T1, KEY1, f, col1;

[0074] Query2: T1, KEY1, f, col1;

10 [0075] Query3: T1, KEY1, f, col2;

[0076] Query4: T1, KEY2, f, col3;

[0077] Query5: T2, KEY3, f, col4;

[0078] Each query task includes a table, a row, a column cluster, and a column. For example, when a query process of query task Query1 is executed, a query path is successively as follows: table-T1, row-KEY1, column cluster-f, and column-col1. Finally, a query result of query task Query1 is obtained.

[0079] In this embodiment, subtasks whose query paths have a same part in subtasks with a same query data source can be merged into the first merged subtask.

20 [0080] The HBASE database is still used as an example. Assume that the first task set includes a total of five subtasks: query tasks Query1 to 5. Because query paths of two subtasks Query1 and Query2 are totally the same, only subtask Query1 can be executed, and a query result of Query2 can be obtained at the same time. Query1 can be referred to as a first merged subtask obtained by merging Query1 and Query2.

[0081] Because query paths of two subtasks Query1 and Query3 are different only in terms of "column", the two query tasks can be merged into a "Get task". The "Get task" can be referred to as a first merged subtask obtained by merging Query1 and Query3.

30 [0082] Because query paths of Query1 and the Query4 are different in terms of "row", Query4 and the "Get task" can be further merged into a "MultiGet task". The "MultiGet task" can be referred to as a first merged subtask obtained by merging Query1, Query3, and Query4.

[0083] Because query paths of Query5 and other subtasks are different in terms of "table", Query5 can only serve as a "new Get task". Based on the above merging processes, a total of five subtasks Query1 to 5 can be merged into two first merged subtasks: a "MultiGet task" and a "new Get task".

5 [0084] It should be noted that a specific implementation process of merging the subtasks in the first task set based on the HBASE database as a query data source is described above. For different query data sources such as the MySQL database, the subtasks in the first task set can be merged based on a query feature of the database. It can be understood that in the above merging process, merging is
10 performed based on a same query path. In another feasible embodiment, another form of task merging can be performed based on a query feature of a query data source.

[0085] Query results of a plurality of subtasks used for merging can be obtained after the first merged subtask is executed. With reference to the above
15 example of the HBASE database, query results of a total of four subtasks Query1 to 4 can be obtained after the "MultiGet task" is executed.

[0086] In this embodiment, after a plurality of subtasks with a same data source are determined from the first task set, the subtasks can be merged based on the query path. In this method, a number of query times is reduced, a number
20 of times of accessing a query data source is reduced, and a query time is reduced. In addition, in this merging method, cross-query task optimization is also implemented, query efficiency is improved, and query resources are saved.

[0087] In an example, after the forming the first task set of the combined task, the method further includes: determining, from the combined task based on the
25 query dependence relationship, subtasks that depend on each subtask in the first task set, to form a second task set of the combined task; executing the subtasks in the first task set, and storing query results; and if the second task set and the first task set have same subtasks with a same query condition and query data source, using a query result of the subtask in the first task set as a query result of the
30 same subtask in the second task set.

[0088] In the above embodiment, the subtasks in the second task set can be determined based on the subtasks in the first task set and a query dependence relationship between subtasks in each query task. The subtask in the second task set depends on the subtask in the first task set.

[0089] FIG. 2 is used as an example. Subtask c can be used as a subtask in the second task set because subtask c depends on subtask b in the first task set. Subtask e can be used as a subtask in the second task set because subtask e depends on subtask d in the first task set.

5 [0090] The subtasks in the first task set are executed to obtain and store query results. For example, a plurality of subtasks in the first task set are concurrently executed to obtain query results of the plurality of subtasks, and then the query results are correspondingly stored. For example, in FIG. 2, the subtasks in the first task set can be concurrently executed, and the query results are
10 correspondingly stored.

[0091] Because the subtask in the second task set depends on the subtask in the first task set, a query result obtained after the subtask in the first task set is executed can be used as a query condition of the subtask in the second task set. In this case, the subtask in the second task set can be compared with a subtask that
15 has been executed in the first task set. If the second task set includes a same subtask that has a same query condition and query data source as the subtask in the first task set, a query result of the corresponding same subtask can be directly obtained from the stored query results, and a query logic of the same subtask in the second task set does not need to be executed again.

20 [0092] Combined task 1 shown in FIG. 2 is used as an example. Combined task 1 includes the first task set and the second task set. A plurality of subtasks in the first task set are executed in advance, and corresponding query results are stored. Assume that query data source "E" of subtask e in the second task set is the same as query data source "B" of subtask b in the first task set and both are the HBASE
25 database, and the query result of subtask d in the first task set is the same as the query condition input in "Input" of subtask b. In this case, it can be determined that subtask e in the second task set and subtask b in the first task set are same subtasks with a same query condition and query data source. Because subtask b has been already executed and the query result is stored, the query result of
30 subtask b can be directly obtained as the query result of subtask e without executing subtask e.

[0093] In this embodiment, query results of executed subtasks can be stored. When a subtask is executed subsequently, a corresponding query result is directly obtained if a same subtask exists. In a process of optimizing the subtask in the

second task set by using the subtask in the first task set, cross-query task optimization is still implemented, repeated queries in different query tasks are reduced, query efficiency is improved, and query resources are saved.

5 [0094] As shown in FIG. 3, the present specification provides a query optimization apparatus, and the apparatus can execute the query optimization method in any embodiment of the present specification. The apparatus can include an acquisition module 201, a combination module 202, and a merging module 203.

10 [0095] The acquisition module 201 is configured to obtain a plurality of to-be-optimized query tasks, each query task including at least one subtask, and when a same query task includes at least two subtasks, a query dependence relationship existing between the at least two subtasks.

15 [0096] The combination module 202 is configured to combine the plurality of query tasks based on the query dependence relationship to obtain one combined task.

[0097] The merging module 203 is configured to merge at least two subtasks in different query tasks based on the combined task to obtain a first merged subtask, the first merged subtask being executed to obtain query results of the at least two subtasks in the different query tasks.

20 [0098] Optionally, as shown in FIG. 4, the merging module 203 includes: a first set submodule 301, configured to determine a non-dependent subtask in each query task based on the query dependence relationship to form a first task set of the combined task; and a merging submodule 302, configured to merge at least two subtasks in the first task set to obtain the first merged subtask.

25 [0099] Optionally, each subtask includes a query condition and a query data source, and that the merging submodule 302 is configured to merge the at least two subtasks in the first task set to obtain the first merged subtask includes: merging at least two subtasks in the first task set that have a same query condition and query data source, to obtain the first merged subtask.

30 [0100] Optionally, each subtask includes a query data source, and that the merging submodule 302 is configured to merge the at least two subtasks in the first task set to obtain the first merged subtask includes: determining a plurality of subtasks with a query data source from the first task set; and merging at least

two subtasks whose query paths have a same part in the plurality of subtasks with the same query data source, to obtain the first merged subtask.

[0101] Optionally, as shown in FIG. 5, the merging module 203 further includes: a second set submodule 401, configured to determine, from the combined task based on the query dependence relationship, subtasks that depend on each subtask in the first task set, to form a second task set of the combined task; a first execution submodule 402, configured to execute the subtasks in the first task set, and store query results; and a result acquisition submodule 403, configured to: if the second task set and the first task set have same subtasks with a same query condition and query data source, obtain a query result of the subtask in the first task set as a query result of the same subtask in the second task set.

[0102] Optionally, each subtask includes a query data source, and the query data source includes at least two data sources of different physical architectures.

[0103] Optionally, as shown in FIG. 6, the merging module 203 further includes: a concurrence execution submodule 501, configured to concurrently execute the subtasks in the first task set; and a caching submodule 502, configured to place, into a cache, a query result of a subtask whose query duration meets a predetermined condition.

[0104] Optionally, as shown in FIG. 7, the apparatus further includes: a query logic determining module 601, configured to determine first query logic from a query task based on a predetermined determining condition, the first query logic including at least one subtask in the same query task; a result storage module 602, configured to store a query result of the first query logic; and a result acquisition module 603, configured to: if determining that query logic that is the same as the first query logic exists in the combined task, obtain the query result of the first query logic as a query result of the query logic that is the same as the first query logic.

[0105] For an implementation process of functions and roles of each module in the apparatus, references can be made to an implementation process of a corresponding step in the above method. Details are omitted herein for simplicity.

[0106] Because an apparatus embodiment corresponds to a method embodiment, for related parts, references can be made to related descriptions in

the method embodiment. The apparatus embodiment described above is merely an example. The units described as separate parts can or cannot be physically separate, and parts displayed as units can or cannot be physical units, can be located in one position, or can be distributed on a plurality of network units.

5 Some or all of the modules can be selected based on actual needs to achieve the objectives of the solutions in at least one embodiment of the present specification. A person of ordinary skill in the art can understand and implement the embodiments of the present application without creative efforts.

[0107] The present specification further provides a computer device, including
10 a memory, a processor, and a computer program that is stored in the memory and that can run on the processor. The processor can implement the query optimization method in any implementation of the present specification when executing the program.

[0108] The present specification further provides a computer readable storage
15 medium, and the computer readable storage medium stores a computer program. Steps of the query optimization method in any implementation of the present specification are implemented when the program is executed by a processor.

[0109] The non-transitory computer-readable storage medium can be a
20 read-only memory (ROM), a random access memory (RAM), a CD-ROM, a magnetic tape, a floppy disk, an optical data storage device, etc. No limitation is imposed in the present specification.

[0110] A person skilled in the art can easily figure out another implementation
of the present specification after thinking over the specification and practicing the present specification herein. The present specification is intended to cover
25 any variations, uses, or adaptations of the present specification, and these variations, uses, or adaptations follow the general principles of the present specification and include common knowledge or conventional techniques that are not disclosed in the technical field of the present specification. The present specification and the embodiments are merely considered as examples, and the
30 actual scope and the spirit of the present specification are pointed out by the following claims.

[0111] It should be understood that the present specification is not limited to the precise structures that have been described above and illustrated in the accompanying drawings, and various modifications and changes can be made

without departing from the scope of the present specification. The scope of the present specification is limited by the appended claims only.

- [0112]** The previous descriptions are merely preferred embodiments of the present specification, but are not intended to limit the present specification. Any
- 5 modification, equivalent replacement, or improvement made without departing from the spirit and principle of the present specification shall fall within the protection scope of the present specification.

CLAIMS

What is claimed is:

1. A query optimization method, the method comprising:

5 obtaining a plurality of to-be-optimized query tasks, each query task including at least one subtask, and when a same query task includes at least two subtasks, a query dependence relationship existing between the at least two subtasks;

combining the plurality of query tasks based on the query dependence relationship to obtain one combined task; and

10 merging at least two subtasks in different query tasks based on the combined task to obtain a first merged subtask, the first merged subtask being executed to obtain query results of the at least two subtasks in the different query tasks.

2. The method according to claim 1, wherein the merging the at least two subtasks in the different query tasks to obtain the first merged subtask includes:

15 determining a non-dependent subtask in each query task based on the query dependence relationship to form a first task set of the combined task; and

merging at least two subtasks in the first task set to obtain the first merged subtask.

20 3. The method according to claim 2, wherein each subtask includes a query condition and a query data source; and

the merging the at least two subtasks in the first task set to obtain the first merged subtask includes:

merging at least two subtasks in the first task set that have a same query condition and query data source, to obtain the first merged subtask.

25 4. The method according to claim 2, wherein each subtask includes a query data source; and

the merging the at least two subtasks in the first task set to obtain the first merged subtask includes:

30 determining a plurality of subtasks with a same query data source from the first task set; and

merging at least two subtasks whose query paths have a same part in the plurality of subtasks with the same query data source, to obtain the first merged subtask.

5. The method according to claim 2, after the forming the first task set of the combined task, further comprising:

determining, from the combined task based on the query dependence relationship, subtasks that depend on each subtask in the first task set, to form a

5 second task set of the combined task;

executing the subtasks in the first task set, and storing query results; and

if the second task set and the first task set have same subtasks with a same query condition and query data source, using a query result of the subtask in the first task set as a query result of the same subtask in the second task set.

10 6. The method according to claim 1 or 2, wherein each subtask includes a query data source, and the query data source includes at least two data sources of different physical architectures.

7. The method according to claim 2, after the forming the first task set of the combined task, further comprising:

15 concurrently executing the subtasks in the first task set; and

placing, into a cache, a query result of a subtask whose query duration meets a predetermined condition.

8. The method according to claim 1, before the combining the plurality of query tasks to obtain the one combined task, further comprising:

20 determining a first query logic from a query task based on a predetermined determining condition, the first query logic including at least one subtask in the same query task; and

storing a query result of the first query logic; and

25 after the combining the plurality of query tasks to obtain the one combined task, further comprising:

if determining that a query logic that is the same as the first query logic exists in the combined task, using the query result of the first query logic as a query result of the query logic that is the same as the first query logic.

9. A query optimization apparatus, the apparatus comprising:

30 an acquisition module, configured to obtain a plurality of to-be-optimized query tasks, each query task including at least one subtask, and when a same query task includes at least two subtasks, a query dependence relationship existing between the at least two subtasks;

a combination module, configured to combine the plurality of query tasks based on the query dependence relationship to obtain one combined task; and

a merging module, configured to merge at least two subtasks in different query tasks based on the combined task to obtain a first merged subtask, the first merged subtask being executed to obtain query results of the at least two subtasks in the different query tasks.

10. The apparatus according to claim 9, wherein the merging module includes:

a first set submodule, configured to determine a non-dependent subtask in each query task based on the query dependence relationship to form a first task set of the combined task; and

a merging submodule, configured to merge at least two subtasks in the first task set to obtain the first merged subtask.

11. The apparatus according to claim 10, wherein each subtask includes a query condition and a query data source; and

that the merging submodule is configured to merge the at least two subtasks in the first task set to obtain the first merged subtask includes:

merging at least two subtasks in the first task set that have a same query condition and query data source, to obtain the first merged subtask.

12. The apparatus according to claim 10, wherein each subtask includes a query data source; and

that the merging submodule is configured to merge the at least two subtasks in the first task set to obtain the first merged subtask includes:

determining a plurality of subtasks with a same query data source from the first task set; and

merging at least two subtasks whose query paths have a same part in the plurality of subtasks with the same query data source, to obtain the first merged subtask.

13. The apparatus according to claim 10, wherein the merging module further includes:

a second set submodule, configured to determine, from the combined task based on the query dependence relationship, subtasks that depend on each subtask in the first task set, to form a second task set of the combined task;

a first execution submodule, configured to execute the subtasks in the first task set, and store query results; and

5 a result acquisition submodule, configured to: if the second task set and the first task set have same subtasks with a same query condition and query data source, obtain a query result of the subtask in the first task set as a query result of the same subtask in the second task set.

14. The apparatus according to claim 9 or 10, wherein each subtask includes a query data source, and the query data source includes at least two data sources of different physical architectures.

10 15. The apparatus according to claim 10, wherein the merging module further includes:

a concurrence execution submodule, configured to concurrently execute the subtasks in the first task set; and

15 a caching submodule, configured to place, into a cache, a query result of a subtask whose query duration meets a predetermined condition.

16. The apparatus according to claim 9, wherein the apparatus further comprises:

20 a query logic determining module, configured to determine a first query logic from a query task based on a predetermined determining condition, the first query logic including at least one subtask in the same query task;

a result storage module, configured to store a query result of the first query logic; and

25 a result acquisition module, configured to: if determining that a query logic that is the same as the first query logic exists in the combined task, obtain the query result of the first query logic as a query result of the query logic that is the same as the first query logic.

30 17. A computer device, comprising a memory, a processor, and a computer program that is stored in the memory and that can run on the processor, the processor implementing the method according to any one of claims 1 to 8 when executing the program.

18. A computer readable storage medium, the computer readable storage medium storing a computer program, and steps of the method according to any one of claims 1 to 8 being implemented when the program is executed by a processor.