GRAPH DATABASE SYSTEM AND QUERY METHOD THEREFOR

[0001] This application claims priorities to Chinese Patent Application No. 202410263169.X, filed on March 7, 2024 and entitled "GRAPH DATA QUERY, APPARATUS, AND COMPUTING DEVICE CLUSTER", and to Chinese Patent Application No. 202410532715.5, filed on April 29, 2024 and entitled "GRAPH DATABASE SYSTEM AND QUERY METHOD THEREFOR", both of which are incorporated herein by reference in their entireties.

TECHNICAL FIELD

[0002] This application relates to the field of database technologies, and in particular, to a graph database system and a query method therefor.

BACKGROUND

[0003] With the advent of the big data era, the correlations between objects, between people, and between pieces of knowledge have become increasingly close. By representing people, objects, and knowledge as vertices in a graph (graph), and the relationships therebetween as edges in the graph, diverse types of data can be represented by graph data. For example, a social network between people, a communication network between machines, and a knowledge graph can all be represented by graph data. As dedicated databases developed for storing, managing, and querying graph-related data, graph databases have attracted growing attention.

[0004] Currently, most graphs suffer from the problem of missing information. For example, a graph may lack information of some vertices, edges, or information of edges. Under such circumstance, when users perform a graph-based query in a database, they usually query an existing graph in the database directly.

[0005] As a result, when information is missing from the graph being queried, the accuracy of the query against the graph database is compromised.

SUMMARY

[0006] This application provides a graph database system and a query method therefor. This application helps improve accuracy of a query result of the graph database system. Technical solutions provided in this application are as follows:

[0007] According to a first aspect, this application provides a graph database system. The

graph database system includes: an interaction module, configured to receive a data query request sent by a first user, where the data query request indicates to query the graph database system for a specified event; and a processing module, configured to obtain, based on the data query request, first graph data related to the specified event, where the first graph data is graph data obtained by supplementing missing information to second graph data, the second graph data is obtained by querying, based on the data query request, graph data stored in the graph database system, and the graph data indicates a plurality of entities and a relationship among different entities; and the interaction module is further configured to return a response to the data query request to the first user based on the first graph data, where the response carries data that matches the specified event.

[0008] Because the response to the data query request is obtained based on the graph data supplemented with the missing information, failure to retrieve all query results due to the information missing from the graph data can be avoided, which helps improve accuracy of the query results. In addition, the graph database system in this application automatically provides the information missing from the graph data imperceptibly to users, that is, users do not perceive the information missing from the graph data. This can improve user trust in the graph database and help improve usability and user stickiness to the graph database.

[0009] In a possible implementation, the response further carries a probability that the data matches the specified event. In this way, the graph database system provides a more detailed query result for the first user, and can provide more reference information for the first user, so that the first user can make a more appropriate decision based on the probability indicating a degree to which the entity matches the specified event.

[0010] In a possible implementation, the processing module is configured to: obtain, based on the data query request, the second graph data related to the specified event and target information missing from the second graph data, obtain a value of the target information based on the second graph data and an association relationship between the target information and the specified event, and add the value of the target information to the second graph data, to obtain the first graph data, where the target information indicates one or more of the following: an identifier of the entity, property information of the entity, the relationship among different entities, or property information of the relationship.

[0011] In a possible implementation, the graph database system further includes a storage module and a prediction module. The storage module is configured to store graph data; the processing module is configured to: obtain, based on the data query request, the second graph data related to the specified event from the storage module, determine the target information missing from the second graph data, and provide the second graph data and the target information to the prediction module; the prediction module is configured to: predict the value of the target

information based on the second graph data and the target information, add the value of the target information to the second graph data, to obtain the first graph data, and provide the first graph data to the processing module; and the processing module is configured to obtain the response to the data query request based on the first graph data.

[0012] Because the prediction module is a built-in module of the graph database system, the prediction module can be deeply integrated with an existing function of the current graph database system, so that the user can use the prediction module to perform a query in a manner that is transparent to the user in a process of using the graph database system, thereby implementing transparent access to the graph database system with a prediction function, and improving user experience. In addition, in comparison with a technology in which the graph database system can only invoke the prediction function externally, the graph database system provided herein obviates repeated data migration between the graph database system and the external prediction module in a process of implementing the prediction function, thereby reducing query overheads of the graph database system and lowering a usage entry barrier of the graph database system. This helps further improve usability of the graph database system.

[0013] In a possible implementation, the prediction module is implemented based on a pre-trained machine learning model.

[0014] In a possible implementation, the prediction module is configured to: obtain first computing data pre-computed by using the pre-trained machine learning model, and predict the value of the target information based on the first computing data, the second graph data, and the target information. When the first graph data is obtained based on the first computing data pre-computed, a computation process of obtaining the first computing data through computation based on the second graph data does not need to be performed in a process of obtaining the first graph data, thereby reducing a computation amount in the process of obtaining the first graph data, accelerating the process of obtaining the first graph data, and accelerating a process of querying a graph database system 60. In addition, because the machine learning model is obtained through pre-training, a relationship reflected by the first computing data is highly likely to match a relationship between various data in the second graph data and the target information. Therefore, even if the first computing data is not obtained through computation based on the second graph data and the target information, impact of the first computing data on accuracy of the obtained first graph data can be controlled within an acceptable range, thereby ensuring accuracy of the obtained first graph data.

[0015] In a possible implementation, the machine learning model includes a deep learning model, the first computing data is a computation result that is obtained by an embedding layer of a pre-trained deep learning model based on graph data used for training, and the first computing

data is stored in the storage module as property information of the graph data used for training. In this way, when the graph data is obtained, the first computing data obtained based on the graph data can be directly obtained.

[0016] In a possible implementation, the interaction module is further configured to receive configuration information set by a second user for the second graph data, where the configuration information indicates information that needs to be recorded in the second graph data; and the processing module is configured to determine, based on the configuration information, the target information missing from the second graph data. In this way, the second graph data can be completed in a customized manner.

[0017] In a possible implementation, the configuration information indicates a type of the information that needs to be recorded in the second graph data, and the type indicates one or more of the following: the identifier of the entity, the property information of the entity, the relationship among different entities, or the property information of the relationship.

[0018] According to a second aspect, this application provides a query method for a graph database system. The query method for a graph database system is applied to the graph database system. The graph database method includes: receiving a data query request sent by a first user, where the data query request indicates to query the graph database system for a specified event; obtaining, based on the data query request, first graph data related to the specified event, where the first graph data is graph data obtained by supplementing missing information to second graph data, the second graph data is obtained by querying, based on the data query request, graph data stored in the graph database system, and the graph data indicates a plurality of entities and a relationship among different entities; and returning a response to the data query request to the first user based on the first graph data, where the response carries data that matches the specified event.

[0019] In a possible implementation, the response further carries a probability that the data matches the specified event.

[0020] In a possible implementation, the obtaining, based on the data query request, the first graph data related to the specified event includes: obtaining, based on the data query request, the second graph data related to the specified event and target information missing from the second graph data, obtaining a value of the target information based on the second graph data and an association relationship between the target information and the specified event, and adding the value of the target information to the second graph data, to obtain the first graph data, where the target information indicates one or more of the following: an identifier of the entity, property information of the entity, the relationship among different entities, or property information of the relationship.

[0021] In a possible implementation, the obtaining the value of the target information based

on the second graph data and the association relationship between the target information and the specified event includes: predicting the value of the target information based on the second graph data and the target information by using a pre-trained machine learning model.

[0022] In a possible implementation, the predicting the value of the target information based on the second graph data and the target information by using the pre-trained machine learning model includes: obtaining first computing data pre-computed by using the pre-trained machine learning model, and predicting the value of the target information based on the first computing data, the second graph data, and the target information by using the pre-trained machine learning model.

[0023] In a possible implementation, the machine learning model includes a deep learning model, the first computing data is a computation result that is obtained by an embedding layer of a pre-trained deep learning model based on graph data used for training, and the first computing data is stored as property information of the graph data used for training.

[0024] In a possible implementation, the query method for a graph database system further includes: receiving configuration information set by a second user for the second graph data, where the configuration information indicates information that needs to be recorded in the second graph data. Correspondingly, the obtaining the target information missing from the second graph data includes: determining, based on the configuration information, the target information missing from the second graph data.

[0025] In a possible implementation, the configuration information indicates a type of the information that needs to be recorded in the second graph data, and the type indicates one or more of the following: the identifier of the entity, the property information of the entity, the relationship among different entities, or the property information of the relationship.

[0026] According to a third aspect, this application provides a computing device, including a memory and a processor. The memory stores program instructions. The processor runs the program instructions to implement the graph database system according to any one of the first aspect and the possible implementations of the first aspect in this application.

[0027] According to a fourth aspect, this application provides a computing device cluster, including a plurality of computing devices. The plurality of computing devices include a plurality of processors and a plurality of memories, the plurality of memories store program instructions, and the plurality of processors run the program instructions, to cause the computing device cluster to implement the graph database system according to any one of the first aspect and the possible implementations of the first aspect in this application.

[0028] According to a fifth aspect, this application provides a computer-readable storage medium. The computer-readable storage medium is a non-volatile computer-readable storage

medium. The computer-readable storage medium includes program instructions. When the program instructions are run on a computing device, the computing device is caused to implement the graph database system according to any one of the first aspect and the possible implementations of the first aspect in this application.

[0029] According to a sixth aspect, this application provides a computer program product including instructions. When the computer program product runs on a computer, the computer is caused to implement the graph database system according to any one of the first aspect and the possible implementations of the first aspect in this application.

BRIEF DESCRIPTION OF DRAWINGS

[0030] FIG. 1 is a diagram of a type of graph data according to an embodiment of this application;

[0031] FIG. 2 is a diagram of another type of graph data according to an embodiment of this application;

[0032] FIG. 3 is a diagram of an architecture of an implementation scenario related to a graph database system and a query method therefor according to an embodiment of this application;

[0033] FIG. 4 is a diagram of an architecture of another implementation scenario related to a graph database system and a query method therefor according to an embodiment of this application;

[0034] FIG. 5 is a diagram of a manner of deploying an infrastructure resource in a data center according to an embodiment of this application;

[0035] FIG. 6 is a diagram of an architecture of a graph database system according to an embodiment of this application;

[0036] FIG. 7 is a diagram of an architecture of another graph database system according to an embodiment of this application;

[0037] FIG. 8 is a diagram of an architecture of still another graph database system according to an embodiment of this application;

[0038] FIG. 9 is a diagram of an architecture of yet another graph database system according to an embodiment of this application;

[0039] FIG. 10 is a diagram of a data query process according to an embodiment of this application;

[0040] FIG. 11 is a diagram of a query plan tree according to an embodiment of this application;

[0041] FIG. 12 is a diagram of a training process according to an embodiment of this application;

[0042] FIG. 13 is a diagram of a query plan tree used for training according to an embodiment

of this application;

[0043] FIG. 14 is a flowchart of a query method for a graph database system according to an embodiment of this application;

[0044] FIG. 15 is a flowchart of another query method for a graph database system according to an embodiment of this application;

[0045] FIG. 16 is a diagram of a structure of a computing device according to an embodiment of this application;

[0046] FIG. 17 is a diagram of a structure of a computing device cluster according to an embodiment of this application; and

[0047] FIG. 18 is a diagram of a structure of another computing device cluster according to an embodiment of this application.

DESCRIPTION OF EMBODIMENTS

[0048] To make the objectives, technical solutions, and advantages of this application clearer, the following further describes the implementations of this application in detail with reference to the accompanying drawings.

[0049] With the advent of the big data era, the correlations between objects, between people, and between pieces of knowledge have become increasingly close. By representing information such as people, objects, and knowledge as vertices in a graph (graph), and the relationships therebetween as edges in the graph, diverse types of data can be represented by graphs. For example, a social network between people, a communication network between machines, and a knowledge graph can all be represented by graphs. Data represented by graphs is referred to as graph data. Currently, as dedicated databases developed for storing, managing, and querying graph-related data, graph databases have attracted growing attention.

[0050] A graph is a data structure that abstracts entities and a relationship between the entities in information into vertices

(vertices) and a relationship between the vertices, respectively. A graph includes at least two vertices and at least one edge. The vertex represents an entity. The entity refers to an objectively existing and distinguishable thing. In some cases, the vertex is also referred to as a point, a node, an endpoint, or the like. The edge (edge) between vertices represents the relationship between the entities. When there is an association relationship between the entities represented by two vertices, the two vertices are connected by an edge. When there is no association relationship between the entities represented by two vertices, the two vertices are not connected. In an implementation, an association between one entity and another entity includes one or more of the

following cases: One entity pro-actively associates with another entity, and/or the another entity pro-actively associates with the entity. Both the vertex and the edge carry information. Information of a vertex may include: a name (also referred to as an identifier or a label (label)) of an entity represented by the vertex, property (property) information of the entity, and the like. Information of an edge may include: property information indicating a relationship between two entities that are correspondingly represented by two vertices connected by the edge. Optionally, the graph may be a directed graph or an undirected graph. When the graph is a directed graph, an edge pointing from one vertex to another vertex is referred to as an outgoing edge of the vertex; correspondingly, the edge is referred to as an incoming edge of the another vertex.

[0051] In different application scenarios, vertices and edges in a graph have different practical meanings. For example, in a friend recommendation scenario, each vertex in the graph may represent one user. An identifier of the vertex indicates an identifier of the user, for example, a name. Property information of the vertex indicates a feature of the corresponding user, for example, an age, an occupation, or a hobby of the user. An edge in the graph represents a social relationship between two users represented by vertices connected by the edge, for example, a degree of familiarity between the two users. For another example, in a commodity recommendation scenario, some vertices in the graph represent users, and some vertices represent commodities. An identifier of the vertex representing the user in the graph indicates an identifier of the user, for example, a name. Property information of the vertex indicates a feature of the corresponding user, for example, an age, an occupation, or a commodity preference of the user. An identifier of the vertex representing the commodity in the graph indicates an identifier of the commodity, for example, a name or a number of the commodity. Property information of the vertex indicates a feature of the corresponding commodity, for example, a category and an application scope of the commodity. An edge between the vertex representing the user and the vertex representing the commodity indicates an interaction relationship between the user and the commodity, for example, a purchase relationship or a favorite relationship. For another example, in a scenario of employee information statistics collection, some vertices in the graph represent employees, some vertices represent locations, and some vertices represent companies. An identifier of the vertex representing the employee in the graph indicates an identifier of the employee, for example, a name. An identifier of the vertex representing the location in the graph indicates an identifier of the location, for example, a name of the location. In the graph, an identifier of the vertex representing the company indicates an identifier of the company, for example, a name of the company. An edge between the vertex representing the employee and the vertex representing the location indicates that the employee indicated by the vertex resides at the location indicated by the vertex. An edge between the vertex representing the employee and the vertex representing the company indicates that the

employee indicated by the vertex belongs to the company indicated by the vertex, and an edge between the vertex representing the company and the vertex representing the location indicates that the company indicated by the vertex is located at the location indicated by the vertex.

[0052] For example, FIG. 1 is a diagram of a graph indicating employee information according to an embodiment of this application. The graph shown in FIG. 1 includes four vertices and five edges. The four vertices indicate an employee A, an employee B, a company C, and a location D respectively. An edge between a vertex representing the employee A and a vertex representing the location D indicates that the employee A resides at the location D. An edge between a vertex representing the employee B and a vertex representing the location D indicates that the employee B resides at the location D. An edge between a vertex representing the employee A and a vertex representing the company C indicates that the employee A works in the company C. An edge between a vertex representing the employee B and a vertex representing the company C indicates that the employee B works in the company C. An edge between a vertex representing the company C and a vertex representing the location D indicates that the company C is located at the location D. FIG. 2 is a diagram of a graph indicating a social relationship according to an embodiment of this application. The graph shown in FIG. 2 includes five vertices and four edges, and each vertex has an identifier and property information. Identifiers of the five vertices respectively indicate Xiao Zhang, Xiao Li, Xiao Wang, Xiao N, and Xiao O. Property information of a vertex representing Xiao Zhang indicates that a probability that Xiao Zhang is a potential customer of a specified commodity and a probability that Xiao Zhang is a high-net-worth customer are unknown, that is, {buyer:null,rich:null} in FIG. 2. Property information of a vertex representing Xiao Li indicates that Xiao Li is most likely a potential customer of a specified commodity, but a probability that Xiao Li is a high-net-worth customer is unknown, that is, {buyer:true,rich:null} in FIG. 2. Property information of a vertex representing Xiao Wang indicates that a probability that Xiao Wang is a potential customer of a specified commodity and a probability that Xiao Wang is a high-net-worth customer are unknown, that is, {buyer:null,rich:null} in FIG. 2. Property information of a vertex representing Xiao O indicates that a probability that Xiao O is a potential customer of a specified commodity is unknown, but Xiao O is most likely a high-net-worth customer, that is, {buyer:null,rich:true} in FIG. 2. Property information of a vertex representing Xiao N indicates that a probability that Xiao N is a potential customer of a specified commodity and a probability that Xiao N is a high-net-worth customer are unknown, that is, {buyer:null,rich:null} in FIG. 2. An edge between two vertices represents a social relationship between people represented by the two vertices, and the social relationship is presented by property information of the edge. An edge between the vertex representing Xiao Zhang and the vertex representing Xiao Li indicates that Xiao Zhang and Xiao Li are good friends. An edge between the

vertex representing Xiao Li and the vertex representing Xiao O indicates that Xiao Li and Xiao O indicated by respective vertices are good friends. An edge between the vertex representing Xiao Li and the vertex representing Xiao Wang indicates that Xiao Li and Xiao Wang are good friends. An edge between the vertex representing Xiao Wang and the vertex representing Xiao N indicates that Xiao Wang and Xiao N are good friends.

[0053] A graph database (graph database) is a new non-relational database implemented based on graph theory. A data storage structure and a data management mode of the graph database are both based on graph theory. In other words, relationships between data in the graph database are all represented by graphs, and all features of the graph database are implemented by graphs. For example, operations such as create (create), read (read, also referred to as query), update (update), and delete (delete) performed on a graph, a transaction processing capability, high availability, and the like are all implemented based on the graph.

[0054] Currently, the following problem exists in use of graph databases: Most graphs have missing information. Information missing from a graph means that the graph lacks identifiers and property (property) information of some vertices, lacks property information of an edge, or lacks an edge between vertices. For example, in some graph databases, 93.8% of people in a graph do not have birthplace information, 78.5% of people do not have nationality information, 68% of people do not have any occupation information, while in some other graph databases, 50% of artists in a graph do not have birth date information. On this basis, when a user performs a graph-based query in a database, they usually query an existing graph in the database directly. As a result, if some information is missing in the graph, the graph database is unable to return all results expected by the user, thereby compromising accuracy of the query against the graph database. This problem is further exacerbated by a fact that most graphs in a real-world scenario have missing information.

[0055] Corresponding to the graph data shown in FIG. 1, an edge represented by a solid line in FIG. 1 is a present edge in the graph data, and an edge represented by a dashed line is an edge that should be recorded in the graph data but is actually missing. If a user wants to query, based on the graph data, for a person residing at the location D, a corresponding query statement is as follows:

[0056] Match (p: people) – [:reside] -> (c: country {name: Location D})

[0057] Return p.name

[0058] After the user inputs the query statement into the graph database, a result output by the graph database based on the graph data is: employee A. However, people who actually reside at the location D also include the employee B. Because the edge indicated by the dashed line is missing from the graph data, the graph database cannot output information about the people

residing at the location D and the employee B. Therefore, the accuracy of the query against the graph database is low.

[0059] Corresponding to the graph data shown in FIG. 2, property information represented as unknown in FIG. 2 is information missing from the graph data. If a user wants to query for potential high-net-worth customers and potential buyers of a specified commodity based on the graph data, so as to conduct proactive outreach to these potential high-net-worth customers and potential buyers in advance, a corresponding query statement is as follows:

[0060] Match (f:Person {rich:true})

[0061] Match (v:Person{buyer:true})

[0062] Return f.name,v.name

[0063] After the user inputs the query statement into the graph database, a result output by the graph database based on the graph data is: f.name: Xiao O, v.name: Xiao Li. The result indicates that Xiao O may be a high-net-worth customer and Xiao Li is likely to purchase the specified commodity. However, Xiao Zhang, Xiao Wang, and Xiao O are also likely to purchase the specified commodity, and Xiao Li is also likely a high-net-worth customer. Because property information of a vertex is missing from the graph data, the graph database cannot output these query results. Therefore, accuracy of the query against the graph database is low.

[0064] In view of this, embodiments of this application provide a graph database system and a query method therefor. The graph database system includes an interaction module and a processing module. The interaction module is configured to receive a data query request sent by a first user. The data query request indicates to query the graph database system for a specified event. The processing module is configured to obtain, based on the data query request, first graph data related to the specified event. The first graph data is graph data obtained by supplementing missing information to second graph data. The second graph data is obtained by querying, based on the data query request, graph data stored in the graph database system. The graph data indicates a plurality of entities and a relationship among different entities. The interaction module is further configured to return a response to the data query request to the first user based on the first graph data. The response carries data that matches the specified event.

[0065] It can be learned that, after receiving the data query request sent by the user, the graph database system in this application can automatically provide the information missing from the second graph data for which the data query request indicates to query, to obtain the first graph data, and then return a response to the data query request to the user based on the first graph data. Because the response to the data query request is obtained based on the graph data supplemented with the missing information, failure to retrieve all query results due to the information missing from the graph data can be avoided, which helps improve accuracy of the query results. In addition,

the graph database system in this application automatically provides the information missing from the graph data imperceptibly to users, that is, users do not perceive the information missing from the graph data. This can improve user trust in the graph database and help improve usability and user stickiness to the graph database.

[0066] This specification describes the technical solutions of this application in detail from a plurality of perspectives such as an implementation scenario, a method procedure, a hardware apparatus, and a software apparatus.

[0067] The following first describes implementation scenarios of embodiments of this application by using examples.

[0068] FIG. 3 is a diagram of an architecture of an implementation scenario related to a graph database system and a query method therefor according to an embodiment of this application. As shown in FIG. 3, the implementation scenario includes one or more computing devices 10 and a client 20. The one or more computing devices 10 are configured to implement the graph database provided in embodiments of this application. The client 20 can establish a communication connection to the one or more computing devices 10. For example, the client 20 establishes the communication connection to the one or more computing devices 10 through a network. Optionally, the network may be a local area network, the Internet, or another network. This is not limited in embodiments of this application.

[0069] The client 20 is configured for a user to interact with the one or more computing devices 20. In an implementation, the client 20 is configured to send a data query instruction to the computing device 10 according to an indication of the user. The one or more computing devices 20 are configured to query the graph database based on the data query instruction, and return a query result to the client 20.

[0070] In a possible implementation, the client 20 may be a desktop computer, a laptop computer, a mobile phone, a smartphone, a tablet computer, a multimedia player, a smart home appliance, an artificial intelligence device, a smart wearable device, an e-reader, a smart vehicle-mounted device, an internet of things device, or the like. The computing device 10 may be a server (for example, a cloud server). When the implementation scenario includes a plurality of computing devices 10, the plurality of computing devices 10 may be referred to as a computing device cluster. In this case, the computing device cluster is a server cluster including several servers, or is implemented by a cloud computing service center. A large quantity of infrastructure resources owned by a cloud service provider are deployed in the cloud computing service center. For example, a computing resource, a storage resource, and a network resource are deployed in the cloud computing service center. The cloud computing service center may implement, by using the large quantity of infrastructure resources, the graph database provided in embodiments of this

application.

[0071] When the computing device cluster is implemented by using the cloud computing service center, a graph database function implemented by the computing device cluster may be abstracted by a cloud service provider into a graph database cloud service on a cloud platform. In this case, the user can access the cloud management platform by using the client 20. After purchasing the graph database cloud service on the cloud management platform, the user can use, via the cloud management platform, the graph database cloud service provided by the computing device cluster. Optionally, the cloud management platform may be a cloud management platform of a central cloud, a cloud management platform of an edge cloud, or a cloud management platform including a central cloud and an edge cloud. This is not specifically limited in embodiments of this application. In addition, optionally, the graph database cloud service is provided by the cloud management platform as an independent cloud service, or the graph database cloud service is provided as an additional cloud service of another cloud service. This is not specifically limited in embodiments of this application.

[0072] When the computing device cluster is implemented by the cloud computing service center, as shown in FIG. 4, an implementation scenario related to a graph database system and a query method therefor according to an embodiment of this application includes a data center 1 and a client 20. A communication connection can be established between the data center 1 and the client 20 through a network. Optionally, the network may be the Internet, or another network. This is not limited in embodiments of this application. A tenant can interact with the data center 1 by using the client 20. For example, the tenant may send information such as a data query request to the data center 1 by using the client 20. The data center 1 is configured to respond based on the information sent by the client 20.

[0073] A large quantity of infrastructure resources owned by a cloud service provider are deployed in the data center 1, for example, a computing resource, a storage resource, and a network resource. For example, the computing resource may be a computing device (such as a server) that can provide a computing capability. In this case, the computing device 10 is deployed in the data center 1. As shown in FIG. 4, the data center 1 includes a cloud management platform and infrastructure (not shown in FIG. 4). The cloud management platform and the infrastructure are connected through an internal network of the data center. The cloud management platform is configured to manage infrastructure. The infrastructure is configured to provide a public cloud service, for example, the foregoing graph database cloud service. The infrastructure includes a plurality of servers. Optionally, a cloud service is deployed on the server. The cloud service is implemented by running a virtual instance. Therefore, the virtual instance used to implement a tenant service is deployed on the server. The tenant can send a cloud service request and related

information to the server by using the client 20 used by the tenant, and the server can process the cloud service request and the related information, and provide a cloud service for the tenant based on a processed cloud service request and processed related information. For example, the server can receive, based on the query method for a graph database system according to embodiments of this application, a data query request sent by a first user, query for and obtain second graph data based on the data query request, fill the second graph data with missing information, to obtain first graph data, and return a response to the data query request to the first user based on the first graph data.

[0074] The cloud management platform can be logically divided into: a tenant console, a computing management service, a network management service, a storage management service, an authentication service, and an image management service. The tenant console provides an interface or an application programming interface (application programming interface, API) for interaction with the tenant. The computing management service is used to manage the server running the virtual instance and a bare metal server. The network management service is used to manage a network service (for example, a gateway and a firewall). The storage management service is used to manage a storage service (for example, a data bucket service). The authentication service is used to manage an account and a password of the tenant. The image management service is used to manage an image of the virtual instance.

[0075] In the implementation scenario shown in FIG. 4, a plurality of servers are deployed in one data center. The server includes a hardware layer and a software layer. The hardware layer is a common configuration of the server. Hardware devices such as a processor, a memory, a network adapter, a disk, and a bus are deployed at the hardware layer. The software layer includes an operating system installed and running on the server. Such operating system, relative to an operating system of a virtual machine, may be referred to as a host operating system. A virtual machine manager (also referred to as a Hypervisor) runs on the host operating system. The virtual machine manager is configured to implement compute virtualization, network virtualization, and storage virtualization for the virtual machine, and manage the virtual machine.

[0076] A cloud management platform client runs in the virtual machine manager. The cloud management platform client can receive a control plane command sent by the cloud management platform, create a virtual instance on the server based on the control plane control command, and perform life-cycle management on the virtual instance. For example, the cloud management platform client can detect, in real time, a hardware resource usage status of a server on which the cloud management platform client is deployed, and report the usage status to the cloud management platform. When determining to create a virtual instance on a specific server, the cloud management platform sends a virtual instance creation command to the cloud management

platform client on the server. After receiving the command, the cloud management platform client creates the virtual instance on the server. In this way, the tenant can create, manage, log in to, and operate on the virtual instance in the data center via the cloud management platform.

[0077] The server may be configured to run virtual machines of different specifications. Virtual machines are categorized into the following types: general-purpose computing type, memory-optimized type, ultra-large memory type, and the like. Each type has a specific specification. After the tenant selects a virtual machine specification, the cloud management platform selects a server that supports the specification in the data center, determines that the server has sufficient available hardware resources, and creates a virtual machine with the specification on the server. Configuring the server via the cloud management platform enables analysis and planning of hardware resources of the server. A computing product corresponding to physical hardware is planned based on hardware performance of the server, for example, virtual machines of different specifications are planned, so as to meet differentiated requirements of different tenants. In addition, differentiated pricing strategies can be formulated based on performance disparities among virtual machines of different specifications. For example, a high-performance virtual instance is sold at a higher price, and a common-performance virtual instance is sold at a low price, so that a tenant can purchase the virtual instance based on actual needs.

[0078] In an implementation, as shown in FIG. 5, a location of an infrastructure resource in a data center may be described by a cloud resource deployment region (region) and availability zone (availability zone, AZ). A tenant can deploy a cloud service based on a resource in a specified region and AZ. Regions are determined based on geographical location and network latency. A same resource pool is shared within a same region. It may be understood that public services, such as elastic computing, block storage, object storage, a virtual private cloud (virtual private cloud, VPC) network, an elastic internet protocol (elastic internet protocol, EIP) address, and an image, are shared. Regions are classified as universal regions and dedicated regions. A universal region provides universal cloud services for common tenants. A dedicated region provides services of a same type only or for specific tenants. A region usually contains a plurality of AZs. The plurality of AZs within the region are interconnected using high-speed optical fibers to meet a tenant requirement for constructing a cross-AZ high-availability system. One AZ is a set of one or more data centers shown in FIG. 5. Within an AZ, computing, network, storage, and other resources are logically divided into a plurality of clusters.

[0079] The tenant may send an instruction to the cloud management platform by using the client 20 used by the tenant, so as to create, manage, log in to, and operate on a virtual instance on the server, and use a cloud service provided by the virtual instance. For example, the cloud management platform can provide an access interface. Optionally, the access interface is provided

in a form of an interface or API. The tenant can remotely access the access interface by using a client to register a cloud account and set a password therefor on the cloud management platform and use the cloud account and password to log in to the cloud management platform. The cloud management platform may further authenticate the cloud account and password. After authentication is successful, the tenant can select and purchase a virtual instance of a specific specification (processor, memory, and disk) on the cloud management platform. After the tenant purchases the virtual instance, the cloud management platform provides the tenant with a remote login account and password for the virtual instance. The tenant may remotely log in to the virtual instance on the client using the remote login account and password, and install and run an application of the tenant in the virtual instance, so as to implement a service of the tenant by using the application.

[0080] In an implementation, the graph database system and the query method therefor provided in embodiments of this application may be implemented by running an executable program on one or more computing devices 10. For example, optionally, the executable program of the graph database system and the query method therefor may be presented in a form of an application installation package. After the application installation package is installed on the one or more computing devices 10, the graph database system and the query method therefor can be implemented by running the executable program. When the computing device cluster is implemented by the cloud computing service center, the graph database system and the query method therefor provided in embodiments of this application may be implemented by running the executable program on a computing device in the data center 1.

[0081] It should be understood that the foregoing content is examples for describing the implementation scenarios of the graph database system and the query method therefor provided in embodiments of this application, and does not constitute any limitation on the implementation scenarios of the graph database system and the query method therefor. A person of ordinary skill in the art can learn that, as service requirements change, the implementation scenarios of the graph database system and the query method therefor may be adjusted based on application requirements, and are not specifically limited in embodiments of this application. In addition, when the graph database system and the query method therefor provided in embodiments of this application are used in another scenario, the executable program of the method may be presented in a form of an application installation package or presented in another manner, which is not enumerated in embodiments of this application.

[0082] The following first describes an implementation of the graph database system provided in embodiments of this application. FIG. 6 is a diagram of an architecture of a graph database system according to an embodiment of this application. As shown in FIG. 6, a graph database

system 60 includes an interaction module 601 and a processing module 602.

[0083] The interaction module 601 is configured to receive a data query request sent by a first user, and provide the data query request to the processing module 602. The data query request indicates to query the graph database system 60 for a specified event. For example, the data query request indicates to query for potential high-net-worth customers and potential buyers of a specified commodity. When the first user needs to query for the specified event, the first user may perform a specified operation on a client used by the first user, to trigger the client to send the data query request to the graph database system 60. After the client sends the data query request to the graph database system 60, the interaction module 601 receives the data query request. In a possible implementation, the graph database system 60 can provide an application programming interface (application programming interface, API) for the client, and the client can trigger the data query request by accessing the API.

[0084] The processing module 602 is configured to obtain, based on the data query request, first graph data related to the specified event. The first graph data is graph data obtained by supplementing missing information to second graph data. In other words, the first graph data carries information included in the second graph data and the information missing from the second graph data. The second graph data is obtained by querying, based on the data query request, graph data stored in the graph database system 60. If the data query request indicates to query the graph database system 60 for the specified event, the second graph data is graph data that is related to the specified event and that is in the graph database system 60. That the graph data is related to the specified event means that the graph data records a query result of the specified event. For example, it is assumed that the graph data shown in FIG. 1 and FIG. 2 is recorded in the graph database system 60. When the specified event for which the data query request indicates to query is potential high-net-worth customers and potential buyers of a specified commodity, and because the graph data shown in FIG. 2 records the query result of the specified event, but the graph data shown in FIG. 1 does not record the query result of the specified event, the processing module 602 may determine that the graph data shown in FIG. 2 is the second graph data. It should be noted that the graph database system 60 generally records a large amount of graph data, and there may be a plurality of pieces of graph data recording the query result of the specified event. In this case, the plurality of pieces of graph data recording the query result of the specified event are all second graph data. Correspondingly, the processing module 602 can obtain a plurality of pieces of first graph data that are in a one-to-one correspondence with the plurality of pieces of second graph data. Any piece of first graph data is graph data supplemented with the information missing from the corresponding second graph data.

[0085] In addition, each of the first graph data and the second graph data may be a complete

graph, or a subgraph (subgraph) in a complete graph. A subgraph is a part of a complete graph, including some vertices and some edges in the complete graph. A subgraph is also referred to as a partition (partition). A complete graph may have a plurality of subgraphs, at least some of vertices included in different subgraphs are different, and at least some of edges included in different subgraphs are different. When the second graph data is a subgraph, the subgraph may be optionally obtained by dividing a complete graph based on the data query request. For example, a complete graph represents a relationship between a plurality of entities having a same property, and the plurality of entities may be divided into a plurality of entity sets. Each entity set includes some entities in the plurality of entities, each entity set has one subproperty, and different entity sets have different subproperties. A graph formed by a plurality of entities in one entity set is a subgraph in the complete graph. The data query request may indicate to query for a case in which an entity having a subproperty meets a specified event. In this case, the second graph data obtained based on the data query request is a subgraph indicating a plurality of entities having the subproperty and a relationship between the plurality of entities. For example, it is assumed that a specific complete graph records information about customers who have browsed all commodities of a specific brand. In other words, a same property of a plurality of entities in the complete graph is that the plurality of entities all have browsed the commodities of the brand. When the data query request indicates to query for potential buyers of a specific commodity of the brand, information indicating customers who have browsed the commodity may be determined in the complete graph based on the data query request, to obtain a subgraph shown in FIG. 2. In this case, the subproperty of a plurality of entities in the subgraph is that the plurality of entities all have browsed a specified commodity of the brand. It should be noted that, the foregoing is merely an example for describing properties corresponding to a complete graph and a subgraph. The complete graph and the subgraph may further have other properties, which are not enumerated herein.

[0086] The interaction module 601 is further configured to: after the processing module 602 obtains the first graph data that is related to the specified event and that is queried for based on the data query request, return a response to the data query request to the first user based on the first graph data. The response carries data that matches the specified event. After completing the second graph data to obtain the first graph data, the processing module 602 obtains a query result of the data query request based on the first graph data, and then provides the query result to the interaction module 601. After obtaining the query result, the interaction module 601 returns, to the first user, a response indicating the query result. Alternatively, after obtaining the first graph data, the processing module 602 provides the first graph data to the interaction module 601. After obtaining the first graph data, the interaction module 601 obtains a query result of the data query request based on the first graph data, and then returns, to the first user, a response indicating the query

result. This embodiment of this application is described by using an example in which the processing module 602 provides the query result of the data query request to the interaction module 601, and the interaction module 601 returns, to the first user, the response indicating the query result. The data that matches the specified event indicates an entity indicated by a vertex that is in the first graph data and that matches the specified event. For example, when the data query request indicates to query potential high-net-worth customers and potential buyers of a specified commodity, a query result of the graph data shown in FIG. 2 in response to the data query request indicates that the potential buyers of the specified commodity are Xiao Li, Xiao Zhang, Xiao Wang, and Xiao O, and the potential high-net-worth customers are Xiao O and Xiao Li, that is, data carried in the response to the data query request indicates that "the potential buyers of the specified commodity are Xiao Li, Xiao Zhang, Xiao Wang, and Xiao O, and the potential high-net-worth customers are Xiao O and Xiao Li."

[0087] Optionally, the response may carry not only the data that matches the specified event, but also a probability that an entity indicated by the data matches the specified event. In other words, the response carries the entity that matches the specified event and the probability that the entity matches the specified event. In this way, the graph database system provides a more detailed query result for the first user, and can provide more reference information for the first user, so that the first user can make a more appropriate decision based on the probability indicating a degree to which the entity matches the specified event. Further, optionally, the entity carried in the response is an entity whose probability of matching the specified event is greater than a specified probability threshold. In other words, the probability that the entity carried in the response matches the specified event is greater than the specified probability threshold, and the response further carries a specific value of the probability that the entity matches the specified event. A value of the specified threshold may be determined based on an application requirement. This is not specifically limited in embodiments of this application. For example, Table 1 and Table 2 show query results of the data query request in the foregoing examples. It can be seen from Table 1 that, in addition to names of the potential buyers of the specified commodity, the query result further indicates a probability that these potential customers purchase the specified commodity. It can be seen from Table 2 that the query result indicates not only names of the potential high-net-worth customers, but also a probability that these customers are high-net-worth customers.

Table 1

f:Person {rich:true}	Probability
Xiao Li	1
Xiao Zhang	0.7656757
Xiao Wang	0.202511

f:Person {rich:true}	Probability
Xiao N	0.31452
Xiao O	1

Table 2

v:Person{buyer:true}	Probability
Xiao Li	1
Xiao Zhang	0.8656757
Xiao Wang	0.1231242
Xiao N	0.4251211
Xiao O	0.7546752

[0088] In a possible implementation, after obtaining the data query request, the processing module 602 may first obtain the second graph data based on the data query request, obtain target information missing from the second graph data, obtain a value of the target information based on the second graph data and an association relationship between the target information and the specified event, and add the value of the target information to the second graph data, to obtain the first graph data. For example, after obtaining the data query request, the processing module 602 may obtain the specified event for which the data query request indicates to query, select, from graph data recorded in the graph database system 60, graph data recording a query result of the specified event, and determine the selected graph data as the second graph data. For example, as shown in FIG. 7, the graph database system 60 further includes a storage module 603. The storage module 603 is configured to store graph data. After obtaining the data query request, the processing module 602 may first obtain, from the storage module 603 based on the specified event for which the data query request indicates to query, the second graph data related to the specified event.

[0089] The target information indicates a type of information to be supplemented to the second graph data. In a possible implementation, the target information indicates one or more of the following: an identifier of an entity indicated by a vertex in graph data, property information of an entity indicated by a vertex in graph data, a relationship among different entities indicated by different vertices in graph data, or property information of the relationship. After obtaining the data query request, the processing module 602 may obtain information that needs to be recorded in the graph data and for which the data query request indicates to query. The information that needs to be recorded in the graph data and for which the data query request indicates to query may be indicated by configuration information for the graph data. After obtaining the configuration information of the second graph data, the processing module 602 can determine, based on the configuration information, the target information missing from the second graph data. The

configuration information indicates a type of the information that needs to be recorded in the graph data and for which the data query request indicates to query. For example, the configuration information indicates one or more of the following: an identifier of an entity indicated by a vertex in graph data, property information of an entity indicated by a vertex in graph data, a relationship among different entities indicated by different vertices in graph data, or property information of the relationship.

[0090] In a possible implementation, after obtaining the configuration information, the processing module 602 may check, based on the configuration information, whether all information indicated by the configuration information is recorded in the second graph data, and determine, as the target information missing from the second graph data, information that is indicated by the configuration information but is not recorded in the second graph data. For example, it is assumed that the configuration information of the second graph data indicates: Each vertex in the second graph data needs to record an identifier of an entity indicated by the vertex and property information that respectively indicates a plurality of specified properties, whether there is an edge between every two vertices is determined based on a relationship between entities indicated by the two vertices, and property information of each edge needs to be recorded. In this case, when a vertex in the second graph data does not include an identifier of an entity indicated by the vertex, content indicated by the target information missing from the second graph data includes: The identifier of the entity indicated by the vertex needs to be supplemented to the second graph data. When a vertex in the second graph data does not include property information of a first property in a plurality of specified properties, content indicated by the target information missing from the second graph data includes: The property information of the first property carried by the vertex needs to be supplemented to the second graph data. The first property is any one of the plurality of specified properties. When two vertices in the second graph data are not connected by an edge, but a relationship between entities indicated by the two vertices indicates that the two entities are associated, content indicated by the target information missing from the second graph data includes: The edge between the two vertices needs to be supplemented to the second graph data. When an edge between two vertices in the second graph data does not include property information, content indicated by the target information missing from the second graph data includes: The property information of the edge between the two vertices needs to be supplemented to the second graph data. For example, it is assumed that the second graph data determined by the processing module 602 based on the data query request is the graph data shown in FIG. 2. The configuration information for the graph data indicates that information that needs to be recorded in the graph data includes: each query object, property information about whether the query object is a high-net-worth individual and a potential buyer of a specified commodity, and a social

relationship between query objects that have browsed the specified commodity. The processing module 602 can determine, based on the configuration information, that the graph shown in FIG. 2 lacks the following information: The vertex indicating Xiao Zhang lacks property information about whether Xiao Zhang is a high-net-worth individual and a potential buyer of the specified commodity; the vertex indicating Xiao Li lacks property information about whether Xiao Li is a high-net-worth individual; the vertex indicating Xiao Wang lacks property information about whether Xiao Wang is a high-net-worth individual and a potential buyer of the specified commodity; the vertex indicating Xiao N lacks property information about whether Xiao N is a high-net-worth individual and a potential buyer of the specified commodity; and the vertex indicating Xiao O lacks property information about whether Xiao O is a potential buyer of the specified commodity. The missing information is the target information missing from the graph data shown in FIG. 2. It should be noted that the configuration information may be a part of the information actually missing from the graph data. In this case, information that is not indicated by the configuration information may be considered as information that requires no attention or has low reference value during service implementation, that is, such information is not considered as the information missing from the graph data. For example, the graph data shown in FIG. 2 actually lacks an age and a gender of a customer indicated by each vertex. However, the age and the gender of a customer have low reference value for a query. Therefore, it may be considered that the age and the gender of a customer are not the information missing from the graph data shown in FIG. 2.

[0091] Optionally, the configuration information for the graph data may be set by a user of the graph database. The interaction module 601 is further configured to receive configuration information set by a second user for the second graph data. In this way, the second graph data can be completed in a customized manner. The user may set the configuration information for the graph data for a plurality of times based on actual needs. For example, after setting the configuration information for the graph data for the first time, the user may further modify, add, or delete the configuration information for the graph data. The second user and the first user may be the same or different. For example, the second user is an administrator of the graph database system 60, and the first user is a user of the graph database system 60. In this case, the first user is different from the second user. For another example, both the first user and the second user are users who use the graph database system 60 for a query.

[0092] The second user may set the configuration information for the graph data by using a data definition language (data definition language, DDL). The DDL indicates information that needs to be recorded in the graph data, for example, property information of a vertex and/or an edge in a graph, an edge between vertices, and the like. The following describes the configuration

information represented by the DDL by using an example.

[0093] [Create|Drop] ML Index on ([node selection query | edge selection query] {property selection query}) indicates to create/delete some specific property information of some vertices or edges in/from the graph data. When the DDL indicates to create some specific property information of some vertices or edges in the graph data, it means that the graph data needs to include such property information. In this case, the DDL indicates the configuration information for the graph data. When the graph data lacks such property information, the processing module 602 can fill the graph data with such property information based on the DDL. The node selection query indicates a vertex that needs to be created or deleted, the edge selection query indicates an edge that needs to be created or deleted, and the property selection query indicates property information that needs to be created or deleted. For example, for FIG. 2, Create ML Index on (p: Person {buyer, rich}) indicates to create, on each vertex indicating a person, property information indicating whether the person is a buyer of a specified commodity and whether the person is rich.

[0094] [Create|Drop] ML Edge on ((node selection query 1) - [:`edgeName`] -> (node selection query 2)) indicates to create/delete, in/from the graph data, edges named "edgeName" between nodes selected by the node selection query 1 and nodes selected by the node selection query 2. When the DDL indicates to create, in the graph data, the edges named "edgeName" between the nodes selected by the node selection query 1 and the nodes selected by the node selection query 2, it means that the graph data needs to include these edges. In this case, the DDL indicates the configuration information for the graph data. When the graph data lacks these edges, the processing module 602 can fill the graph data with these edges based on the DDL. For example, for FIG. 2, Create ML Edge on (p: Person - [:Know] -> p2: Person) indicates to create, among all vertices indicating persons, edges indicating whether the persons indicated by the vertices know (Know) each other.

[0095] It should be noted that the foregoing is merely an example of the configuration information represented by the DDL, and there may be more implementations. For example, the DDL can further indicate effective time and an effective scope of the configuration information, a manner of completing the graph data based on the configuration information, and the like. The effective time of the configuration information refers to a time when the configuration information takes effect after the configuration information is set for the graph data. For example, the effective time of the configuration information indicates that the configuration information takes effect immediately after being set. To be specific, after the configuration information is set, the information missing from the graph data is determined based on the new configuration information. Alternatively, the effective time of the configuration information indicates that the configuration information takes effect only after the configuration information is set and an effective cycle of

the configuration information is reached. To be specific, effective duration of the configuration information is set in the graph data system; and after the configuration information is set, if elapsed effective duration of old configuration information does not reach the effective duration, the information missing from the graph data is still determined based on the old configuration information. After the elapsed effective duration of the old configuration information reaches the effective duration, the information missing from the graph data is determined based on the new configuration information. The effective scope indicates an applicable scope of the configuration information. For example, the effective scope indicates that the configuration information is applicable to a complete graph, or the effective scope indicates that configuration information is applicable to some specific subgraphs of a complete graph.

[0096] In a possible implementation, an operation of completing the second graph data may be obtained through prediction. A prediction process may be performed by the processing module 602, or may be performed by the processing module 602 by invoking another module. For example, as shown in FIG. 7, the graph database system 60 further includes a prediction module 604. The processing module 602 may invoke the prediction module 604 to complete the second graph data, to obtain the first graph data. In other words, the processing module 602 is configured to: after obtaining the second graph data and the target information missing from the second graph data based on the data query request, provide the second graph data and the target information to the prediction module 604. The prediction module 604 is configured to: after obtaining the second graph data and the target information, predict a value of the target information based on the second graph data and an association relationship between the target information and the specified event for which the data query request indicates to query, add the value of the target information obtained through prediction to the second graph data, to obtain the first graph data, and provide the first graph data to the processing module 602. An implementation process is as follows: After obtaining the second graph data and the target information, the prediction module 604 analyzes existing information in the second graph data, the target information, and the specified event, to obtain various types of data in the second graph data and the association relationship between the target information and the specified event, predicts the value of the target information based on the association relationship, and completes the second graph data based on the value of the target information obtained through prediction, to obtain the first graph data.

[0097] In a possible implementation, the prediction module 604 is implemented by using a machine learning (machine learning, ML) technology. For example, the prediction module 604 is implemented based on a pre-trained machine learning model. For example, optionally, the machine learning model is a deep learning model. For example, the machine learning model is a graph neural network (graph neural network, GNN), a graph convolutional network (graph convolutional

network, GCN), a graph attention network (graph attention network, GAT), graph sample and aggregate (graph sample and aggregate, GraphSAGE), or another machine learning model that can implement a corresponding function. In this case, after obtaining the second graph data and the target information missing from the second graph data, the processing module 602 may input the second graph data and the target information missing from the second graph data to the machine learning model, and receive the first graph data output by the machine learning model. Correspondingly, the graph database system 60 in this application is also referred to as a neural graph database (neural graph database) system.

[0098] When the prediction module 604 is implemented based on the deep learning model, the output of the deep learning model further includes a confidence of the value of the target information. A confidence of a value of any target information indicates how credible the value of the target information is. In this case, when a query result of the data query request is obtained based on the first graph data provided by the prediction module 604, a probability that an entity included in the query result matches the specified event can be further obtained based on the confidence. For example, the graph data shown in FIG. 2 is still used as an example. It is assumed that the provided information missing from the first graph data output by the deep learning model indicates the following content: Property information of the vertex indicating Xiao Li indicates that a confidence that Xiao Li is a high-net-worth individual is 1; property information of the vertex indicating Xiao Zhang indicates that a confidence that Xiao Zhang is a high-net-worth individual is 0.7656757 and a confidence that Xiao Zhang is a potential buyer of the specified commodity is 0.8656757; property information of the vertex indicating Xiao Wang indicates that a confidence that Xiao Wang is a high-net-worth individual is 0.202511 and a confidence that Xiao Wang is a potential buyer of the specified commodity is 0.1231242; property information of the vertex indicating Xiao N indicates that a confidence that Xiao N is a high-net-worth individual is 0.31452 and a confidence that Xiao N is a potential buyer of the specified commodity is 0.4251211; and property information of the vertex indicating Xiao O indicates that a confidence that Xiao O is a potential buyer of the specified commodity is 0.7546752. When the query result is obtained based on the first graph data, the confidence may be directly determined as the probability that an entity included in the query result matches the specified event, that is, results shown in Table 1 and Table 2 are obtained.

[0099] When the prediction module 604 is implemented based on the pre-trained machine learning model, the machine learning model is further configured to: obtain first computing data pre-computed by using the pre-trained machine learning model, and predict the value of the target information based on the first computing data, the second graph data, and the target information, to obtain the first graph data. Optionally, the first computing data is obtained through computation

based on training data. For example, the first computing data is a computation result that is obtained by an embedding (embedding) layer of a pre-trained deep learning model based on the training data. In a possible implementation, after obtaining the first computing data, the second graph data, and the target information, the machine learning model may filter, from the first computing data based on the second graph data and the target information, for computing data that matches the second graph data and the target information, predict the value of the target information based on the computing data obtained through filtering, and complete the second graph data based on the value of the target information, to obtain the first graph data. For example, the graph data used to train the deep learning model includes graph data in the graph database system 60 used by a user for a query. The graph data is generally represented by text. In a training process of the deep learning model, the embedding layer of the deep learning model converts the graph data represented by text into a vector (vector) representation, and then performs a subsequent training process based on the vector. After the training process of the deep learning model is completed, the prediction module 604 may: obtain and store the vector obtained by the embedding layer through conversion in the training process; when predicting the value of the target information by using the deep learning model, filter, based on the second graph data and the target information, for a vector used to represent the second graph data from the vector obtained by the embedding layer through conversion in the training process; and predict the value of the target information based on the vector obtained through filtering. That the embedding layer converts the graph data represented by text into the vector representation is an implementation of converting the graph data represented by text into a machine language. The deep learning model may further convert graph data identified by text into another type of machine language based on actual needs. Examples are not enumerated in embodiments of this application.

[00100] Optionally, the first computing data may be stored in the storage module 603. In a possible implementation, optionally, the first computing data obtained based on any graph data is stored in the graph data as property information of the graph data. In this way, when the graph data is obtained, the first computing data obtained based on the graph data can be directly obtained. Alternatively, the graph database system 60 includes at least two storage modules 603. One storage module 603 is configured to store the graph data, and the other storage module 603 is configured to store the first computing data. For example, as shown in FIG. 8, the graph database system 60 includes a graph data storage module 6031 and a computing data storage module 6032. The graph data storage module 6031 is configured to store the graph data, and the computing data storage module 6032 is configured to store the first computing data. After storing the first computing data, the computing data storage module 6032 may further store related data such as metadata of the first computing data in the graph data storage module 6031, so that the processing module 602 can

obtain the related data when reading the second graph data in the graph data storage module 6031, and obtain, from the computing data storage module 6032 based on the related data, the first computing data indicated by the related data.

[00101] When the first graph data is obtained based on the first computing data pre-computed, a computation process of obtaining the first computing data through computation based on the second graph data does not need to be performed in a process of obtaining the first graph data, thereby reducing a computation amount in the process of obtaining the first graph data, accelerating the process of obtaining the first graph data, and accelerating a process of querying a graph database system 60. In addition, because the machine learning model is obtained through pre-training, a relationship reflected by the first computing data is highly likely to match a relationship between various data in the second graph data and the target information. Therefore, even if the first computing data is not obtained through computation based on the second graph data and the target information, impact of the first computing data on accuracy of the obtained first graph data can be controlled within an acceptable range, thereby ensuring accuracy of the obtained first graph data.

[00102] When the first graph data is obtained by using the machine learning model to complete the second graph data, the graph database system 60 can automatically complete the second graph data imperceptibly to users, that is, users do not perceive the information missing from the graph data. This can improve user trust in the graph database system 60 and help improve usability and user stickiness to the graph database system 60. In addition, when the first graph data is obtained by using the machine learning model to complete the second graph data, a requirement of the graph database system 60 on information completeness of the graph data can be reduced, so that graph data labeled with only a small amount of vertex and edge information can also be added to the graph data system. When the vertex and edge information in the graph data is obtained by the user (for example, an administrator of the graph data system) through labeling, workload of labeling graph data by the user can be reduced. In addition, because the prediction module 604 is a built-in module of the graph database system 60, the prediction module 604 can be deeply integrated with an existing function of the current graph database system 60, so that the user can use the prediction module 604 to perform a query in a manner that is transparent to the user in a process of using the graph database system 60, thereby implementing transparent access to the graph database system 60 with a prediction function, and improving user experience. In addition, in comparison with a technology in which the graph database system 60 can only invoke the prediction function externally, the graph database system 60 provided herein obviates repeated data migration between the graph database system 60 and the external prediction module 604 in a process of implementing the prediction function, thereby reducing query overheads of the graph database system 60 and

lowering a usage entry barrier of the graph database system 60. This helps further improve usability of the graph database system 60. That the user uses the prediction module 604 to perform a query in a manner that is transparent to the user includes but is not limited to that a query language is transparent to the user, interaction between an original function of the graph database and the prediction module 604 is transparent to the user, and the like.

[00103] In addition, optionally, after obtaining the first graph data, the processing module 602 stores the first graph data. When receiving the data query request again, optionally, the processing module 602 first determines, based on the data query request, whether the first graph data is graph data for which the data query request indicates to query. When the first graph data is the graph data for which the data query request indicates to query, the processing module 602 may directly return a response to the data query request based on the first graph data. In this case, there is no need to perform an operation of completing the second graph data, which can simplify an arithmetic operation process of the processing module 602 and reduce computation overheads. When the first graph data is not the graph data for which the data query request indicates to query, the processing module 602 obtains the second graph data based on the data query request, completes the second graph data to obtain the first graph data, and returns a response based on the first graph data obtained by completing the second graph data.

[00104] It should be noted that in a process of obtaining the first graph data based on the data query request, the processing module 602 may first perform preprocessing based on the data query request, and then obtain the first graph data based on a preprocessing result. For example, a function of the processing module 602 may be implemented by using a parsing unit (parser), an optimization unit (optimizer), and a processing unit (processor). The parsing unit is configured to parse the data query request represented by text, and provide a parsing result to the optimization unit. The parsing result indicates a query requirement and information that needs to be recorded in the graph data for which the data query request indicates to query, that is, the parsing result indicates a query requirement of the data query request and configuration information corresponding to the query requirement. The parsing result may be represented by an abstract syntax tree (abstract syntax tree, AST). The abstract syntax tree is an abstract representation of a syntax structure of source code. The abstract syntax tree indicates a syntax structure of programming language in a tree form. Each node on the tree indicates a structure in the source code. The optimization unit is configured to generate, based on the parsing result, an operation flow that needs to be executed by the data query request, and provide the operation flow to the processing unit. The processing unit is configured to obtain, based on the operation flow, the second graph data for which the data query request indicates to query. The operation flow indicates a plurality of operations that have a sequence. The operation flow may be represented by a query

plan tree (query plan tree). The query plan tree is a tree-shaped data structure that can be recognized by the graph database system 60 and has a plurality of nodes. Each node indicates an operation command. Execution of the operation flow starts from a root node of the query plan tree. Execution at a node other than the root node depends on an execution result of a parent node of the node. Because the operation flow indicates a plurality of operations that have a sequence, and data that needs to be referenced for performing the plurality of operations is different, the operation flow also indicates a sequence of data that needs to be obtained. Correspondingly, when obtaining the second graph data from the storage module 603, the processing unit may sequentially obtain data in the second graph data based on the data indicated by the operation flow.

[00105] It can be learned that the graph database system 60 in this application may be implemented by a plurality of components. FIG. 9 is a diagram of implementing data query by using a plurality of components in the graph database system 60 shown in FIG. 8. As shown in FIG. 9, the interaction module 601 is also referred to as a graph query interface (graph query interface), and is configured to connect to a user, and provide a graph query capability for the user. For example, the interaction module 601 is configured to receive a data query request sent by a first user, and return a response to the data query request to the first user. The processing module 602 includes a parsing unit 6021, an optimization unit 6022, and a processing unit 6023. The parsing unit is configured to receive the data query request provided by the interaction module 601, as shown in step ① in FIG. 10, parse the data query request of the user into an abstract syntax tree, and provide the abstract syntax tree to the optimization unit, as shown in step ② in FIG. 10. The optimization unit is configured to convert the abstract syntax tree into a query plan tree, optimize the query plan tree as required, and provide an optimized query plan tree to the processing unit, as shown in step ③ in FIG. 10. The processing unit is configured to query the graph data storage module 6031 for related vertices and edges based on the query plan tree, as shown in step ④ in FIG. 10. The graph data storage module 6031 provides the related vertices and edges to the processing unit, as shown in step ⑤ in FIG. 10. After obtaining the vertices and edges provided by the graph data storage module 6031, the processing unit provides the vertices and edges obtained from the graph data storage module 6031 to the prediction module 604, as shown in step ⑥ in FIG. 10. The vertices and edges that are obtained by the processing unit from the graph data storage module 6031 form the second graph data. After obtaining the second graph data provided by the processing unit, the prediction module 604 completes the second graph data based on the second graph data, to obtain first graph data, and outputs the first graph data to the processing unit, as shown in step ⑨ in FIG. 10. The processing unit receives the first graph data output by the prediction module 604, and provides a query result of the data query request to the interaction module 601 based on the first graph data, as shown in step ⑩ in FIG. 10. If the computing data

storage module 6032 stores first computing data, when completing the second graph data, the prediction module 604 may first obtain the first computing data from the computing data storage module 6032, as shown in step ⑦ in FIG. 10. The computing data storage module 6032 provides the first computing data to the prediction module 604, as shown in step ⑧ in FIG. 10. After obtaining the first computing data, the prediction module 604 completes the second graph data based on the first computing data. A sequence of steps in FIG. 10 indicates an execution sequence of corresponding steps, and a step with a smaller number is executed first.

[00106] For example, it is assumed that the machine learning model is a GCN model, and the query in FIG. 2 is still used as an example for description. When the specified event for which the data query request indicates to query is whether Xiao Li is a high-net-worth customer, FIG. 11 is a diagram of a query plan tree generated by the processing unit. As shown in FIG. 11, the query plan tree includes two branches, namely, a left branch converging into a Union operator from the left and a right branch converging into the Union operator from the right, as shown in FIG. 11. A root node of the left branch indicates to first obtain, from the graph data storage module 6031, all vertices indicating Xiao Li and edges between the vertices. This function is implemented by a NodeByPropertyScan (x.name=Xiao Li) operator in FIG. 11. A first-level child node of the root node indicates to select, from all the obtained vertices, all vertices whose property information indicating whether entities indicated by the vertices are high-net-worth individuals has a value of true. This function is implemented by a Filter(x.rich != null & x.rich = true) operator shown in FIG. 11. A root node of the right branch indicates to first obtain, from the graph data storage module 6031, all vertices indicating Xiao Li and edges between the vertices. This function is implemented by a NodeByPropertyScan (x.name=Xiao Li) operator in FIG. 11. A first-level child node of the root node indicates to select, from all the obtained vertices, all vertices whose property information indicating whether entities indicated by the vertices are high-net-worth individuals has a value of null. This function is implemented by a Filter(x.rich=null) operator in FIG. 11. A second-level child node (that is, a child node of the first-level child node) of the root node indicates to obtain, from the graph data storage module 6031, all vertices and edges within two hops of a selected vertex, and obtain the second graph data based on all the vertices and edges obtained from the graph data storage module 6031. This function is implemented by an ExpandAsSubgraph(x,2) operator in FIG. 11. Vertices within two hops of any given vertex refer to vertices reachable from such vertex as a starting point by traversing two edges. Because the second graph data is a part of a graph stored in the graph data storage module 6031, the second graph data is a subgraph of the graph. A third-level child node (that is, a child node of the second-level child node) of the root node indicates the processing unit to invoke an interface of the prediction module 604 to use the GCN model. The GCN model predicts, based on the second graph data, all property information

whose value is null and that indicates whether entities indicated by the vertices are high-net-worth individuals, records a prediction result in the second graph data, to obtain the first graph data, and provides the first graph data to the processing unit. This function is implemented by a GNNInference(GCN,Person.rich) operator in FIG. 11. A fourth-level child node (that is, a child node of the third-level child node) of the root node indicates the processing unit to filter, from the first graph data, for all vertices whose property information indicating customers indicated by the vertices are high-net-worth individuals, to obtain the query result of the data query request. This function is implemented by a Filter(x.rich=true) operator in FIG. 11. The Union operator summarizes query results of the left and right branches and provides a summary result to a ProduceResults operator. The ProduceResults operator provides the query result to the interaction module 601 based on the summary result.

[00107] The following describes a training process of the machine learning model. The training process includes a plurality of training subprocesses that are sequentially executed. The training subprocess includes: The processing module 602 obtains a data query request used for training, obtains, based on the data query request, graph data used for training to which the data query request indicates to query, determines target information missing from the graph data used for training, and inputs, to the machine learning model, the target information and the graph data used for training, so that the machine learning model performs model training based on the target information and the graph data used for training. After completing the plurality of training subprocesses, the processing module 602 may obtain the first computing data obtained through computation of each piece of graph data used for training by using a trained machine learning model, and store the first computing data.

[00108] Similar to a query process of the graph database system 60, alternatively, the processing module 602 may first preprocess a data query request used for training, and then perform a subsequent training operation based on the preprocessing. For example, when a function of the processing module 602 is implemented by the parsing unit, the optimization unit, and the processing unit, the parsing unit is configured to: obtain the data query request used for training provided by the interaction module 601, as shown in step ① in FIG. 12, parse the data query request used for training represented by text, and provide a parsing result to the optimization unit, as shown in step ② in FIG. 12. The parsing result indicates information that needs to be recorded in graph data for which the data query request used for training indicates to query, that is, the parsing result indicates configuration information corresponding to the data query request used for training. The optimization unit is configured to generate, based on the parsing result, an operation flow that needs to be executed by the data query request used for training, and provide the operation flow to the processing unit, as shown in step ③ in FIG. 12. The processing unit is

configured to obtain, from the graph data storage module 6031 based on the operation flow, the second graph data for which the data query request used for training indicates to query, as shown in step ④ in FIG. 12. The graph data storage module 6031 provides the related vertices and edges to the processing unit, as shown in step ⑤ in FIG. 12. When obtaining the second graph data from the storage module 603, the processing unit may sequentially obtain data in the second graph data based on the data indicated by the operation flow. After obtaining the vertices and edges provided by the graph data storage module 6031, the processing unit provides the vertices and edges obtained from the graph data storage module 6031 to the prediction module 604, as shown in step ⑥ in FIG. 12. The vertices and edges that are obtained by the processing unit from the graph data storage module 6031 form the second graph data. After obtaining the second graph data provided by the processing unit, the prediction module 604 completes the second graph data based on the second graph data, to obtain first graph data, and outputs the first graph data to the processing unit, as shown in step ⑦ in FIG. 12. After completing the plurality of training subprocesses, the processing module 602 may obtain the first computing data obtained through computation of each piece of graph data used for training by using a trained machine learning model, and store the first computing data in the computing data storage module 6032, as shown in step ⑧ in FIG. 12. After storing the first computing data, the computing data storage module 6032 may store related data such as metadata of the first computing data in the graph data storage module 6031, as shown in step ⑨ in FIG. 12. A sequence of steps in FIG. 12 indicates an execution sequence of corresponding steps, and a step with a smaller number is executed first.

[00109] For example, it is assumed that the machine learning model is a GCN model, and the query in FIG. 2 is still used as an example for description. When the specified event for which the data query request used for training indicates to query is a potential high-net-worth customer, FIG. 13 is a diagram of a query plan tree used for training generated by the processing unit. As shown in FIG. 13, a root node of the query plan tree indicates to first obtain, from the graph data storage module 6031, all vertices indicating persons and edges between the vertices. This function is implemented by a NodeByPropertyScan(Person) operator in FIG. 13. A first-level child node of the root node indicates to obtain, from the graph data storage module 6031, all vertices and edges within two hops of all vertices obtained from the root node, and obtain the second graph data (that is, a subgraph) based on all the vertices and edges obtained from the graph data storage module 6031. This function is implemented by an ExpandAsSubgraph(2) operator in FIG. 13. A second-level child node (that is, a child node of the first-level child node) of the root node indicates the processing unit to invoke an interface of the prediction module 604 to train the GCN model. In a training process, the GCN model needs to predict, based on the second graph data, all property information whose value is null and that indicates whether entities indicated by the vertices are

high-net-worth individuals, records a prediction result in the second graph data, to obtain the first graph data, and provides the first graph data to the processing unit. This function is implemented by a GNNTraining(GCN) operator in FIG. 13. A third-level child node (that is, a child node of the second-level child node) of the root node indicates the processing unit to obtain, from the prediction module 604, a result of computation performed by the embedding layer of the GCN model on the graph data for training (that is, the first computing data), and then store the first computing data in the computing data storage module 6032. After storing the first computing data, the computing data storage module 6032 may store related data such as metadata of the first computing data in the graph data storage module 6031. A function of obtaining the first computing data is implemented by a Pre-ComputeEmbedding operator in FIG. 13. A fourth-level child node (that is, a child node of the third-level child node) of the root node indicates to store a GCN model that has been trained. This function is implemented by a StoreModel operator in FIG. 13.

[00110] It can be learned that, after receiving the data query request sent by the user, the graph database system in this application can automatically provide the information missing from the second graph data for which the data query request indicates to query, to obtain the first graph data, and then return a response to the data query request to the user based on the first graph data. Because the response to the data query request is obtained based on the graph data supplemented with the missing information, failure to retrieve all query results due to the information missing from the graph data can be avoided, which helps improve accuracy of the query results. In addition, the graph database system in this application automatically provides the information missing from the graph data imperceptibly to users, that is, users do not perceive the information missing from the graph data. This can improve user trust in the graph database and help improve usability and user stickiness to the graph database. In addition, the graph database system provided in this application can reuse a function of the existing graph database system to a large extent, and during an upgrade and update of the graph database system, some executable programs used to implement the existing graph database system can be reused, thereby reducing repeated development work, lowering development and maintenance costs, and improving upgrade and update efficiency of the graph database system.

[00111] The following describes a process of implementing a query method for a graph database system according to embodiments of this application. The query method for a graph database system is applied to the graph database system according to embodiments of this application. FIG. 14 is a flowchart of a query method for a graph database system according to an embodiment of this application. As shown in FIG. 14, the query method for a graph database system includes the following steps:

[00112] Step 1401: Receive a data query request sent by a first user, where the data query

request indicates to query the graph database system for a specified event.

[00113] For an implementation process of step 1401, refer to related descriptions of the foregoing graph database system. Details are not described herein again.

[00114] Step 1402: Obtain, based on the data query request, first graph data related to the specified event, where the first graph data is graph data obtained by supplementing missing information to second graph data, the second graph data is obtained by querying, based on the data query request, graph data stored in the graph database system, and the graph data indicates a plurality of entities and a relationship among different entities.

[00115] In a possible implementation, the obtaining, based on the data query request, the first graph data related to the specified event includes: obtaining, based on the data query request, the second graph data related to the specified event and target information missing from the second graph data, obtaining a value of the target information based on the second graph data and an association relationship between the target information and the specified event, and adding the value of the target information to the second graph data, to obtain the first graph data. The target information indicates one or more of the following: an identifier of the entity, property information of the entity, the relationship among different entities, or property information of the relationship.

[00116] Optionally, the obtaining the value of the target information based on the second graph data and the association relationship between the target information and the specified event includes: predicting the value of the target information based on the second graph data and the target information by using a pre-trained machine learning model. The predicting the value of the target information based on the second graph data and the target information by using the pre-trained machine learning model includes: obtaining first computing data pre-computed by using the pre-trained machine learning model, and predicting the value of the target information based on the first computing data, the second graph data, and the target information by using the pre-trained machine learning model.

[00117] In a possible implementation, the machine learning model includes a deep learning model, the first computing data is a computation result that is obtained by an embedding layer of a pre-trained deep learning model based on graph data used for training, and the first computing data is stored as property information of the graph data used for training.

[00118] In a possible implementation, as shown in FIG. 15, the method further includes step 1404: Receive configuration information set by a second user for the second graph data, where the configuration information indicates information that needs to be recorded in the second graph data. Correspondingly, the obtaining the target information missing from the second graph data includes: determining, based on the configuration information, the target information missing from the second graph data. In other words, step 1402 includes step 14021: Obtain, based on the data query

request, the second graph data related to the specified event, determine, based on the configuration information, the target information missing from the second graph data based on the configuration information, obtain the value of the target information based on the second graph data and an association relationship between the target information and the specified event, and add the value of the target information to the second graph data, to obtain the first graph data. The configuration information indicates a type of the information that needs to be recorded in the second graph data, and the type indicates one or more of the following: the identifier of the entity, the property information of the entity, the relationship among different entities, or the property information of the relationship.

[00119] For an implementation process of step 1402, refer to related descriptions of the foregoing graph database system. Details are not described herein again.

[00120] Step 1403: Return a response to the data query request to the first user based on the first graph data, where the response carries data that matches the specified event.

[00121] Optionally, the response further carries a probability that the data matches the specified event. For an implementation process of step 1403, refer to related descriptions of the foregoing graph database system. Details are not described herein again.

[00122] In conclusion, in the query method for a graph database according to embodiments of this application, after receiving the data query request sent by the user, the graph database system can automatically provide the information missing from the second graph data for which the data query request indicates to query, to obtain the first graph data, and then return a response to the data query request to the user based on the first graph data. Because the response to the data query request is obtained based on the graph data supplemented with the missing information, failure to retrieve all query results due to the information missing from the graph data can be avoided, which helps improve accuracy of the query results. In addition, the graph database system in this application automatically provides the information missing from the graph data imperceptibly to users, that is, users do not perceive the information missing from the graph data. This can improve user trust in the graph database and help improve usability and user stickiness to the graph database. In addition, the graph database system provided in this application can reuse a function of the existing graph database system to a large extent, and during an upgrade and update of the graph database system, some executable programs used to implement the existing graph database system can be reused, thereby reducing repeated development work, lowering development and maintenance costs, and improving upgrade and update efficiency of the graph database system.

[00123] It should be noted that a sequence of steps in the query method for a graph database system according to embodiments of this application may be appropriately adjusted, and a new step may be added or an existing step may be deleted based on actual needs. Any variation method

readily figured out by a person skilled in the art within the technical scope disclosed in this application shall fall within the protection scope of this application. Therefore, details are not described.

[00124] In this application, the interaction module 601, the processing module 602, the storage module 603, and the prediction module 604 may be implemented by software, or may be implemented by hardware. For example, the following uses the interaction module 601 as an example to describe an implementation of the interaction module 601. Similarly, for implementations of the processing module 602, the storage module 603, and the prediction module 604, refer to the implementation of the interaction module 601.

[00125] As an example of a software functional unit, the interaction module 601 may include code run on a computing instance. The computing instance may include at least one of a physical host (computing device), a virtual machine, and a container. Further, there may be one or more computing instances. For example, the interaction module 601 may include code run on a plurality of hosts/virtual machines/containers. It should be noted that when the graph database system in this application is deployed on a cloud platform, the plurality of hosts/virtual machines/containers configured to run the code may be distributed in a same region (region), or may be distributed in different regions. Further, the plurality of hosts/virtual machines/containers configured to run the code may be distributed in a same availability zone (availability zone, AZ), or may be distributed in different AZs. Each AZ includes one cloud data center or a plurality of cloud data centers that are geographically close to each other. Usually, one region may include a plurality of AZs.

[00126] Similarly, the plurality of hosts/virtual machines/containers configured to run the code may be distributed in a same virtual private cloud (virtual private cloud, VPC), or may be distributed in a plurality of VPCs. Usually, one VPC is disposed in one region. A communication gateway needs to be disposed in each VPC for communication between two VPCs in a same region and cross-region communication between VPCs in different regions. The VPCs are interconnected through the communication gateway.

[00127] As an example of a hardware functional unit, the interaction module 601 may include at least one computing device, for example, a server. Alternatively, the interaction module 601 may be a device implemented by an application-specific integrated circuit (application-specific integrated circuit, ASIC) or a programmable logic device (programmable logic device, PLD), or the like. The PLD may be implemented by a complex programmable logic device (complex programmable logic device, CPLD), a field-programmable gate array (field-programmable gate array, FPGA), generic array logic (generic array logic, GAL), or any combination thereof.

[00128] A plurality of computing devices included in the interaction module 601 may be distributed in a same region, or may be distributed in different regions. The plurality of computing

devices included in the interaction module 601 may be distributed in a same AZ, or may be distributed in different AZs. Similarly, the plurality of computing devices included in the interaction module 601 may be distributed in a same VPC, or may be distributed in a plurality of VPCs. The plurality of computing devices may be any combination of computing devices such as a server, an ASIC, a PLD, a CPLD, an FPGA, and GAL.

[00129] It should be noted that, in another embodiment, any one of the interaction module 601, the processing module 602, the storage module 603, and the prediction module 604 may be configured to perform any step in the query method for a graph database system. Steps implemented by the interaction module 601, the processing module 602, the storage module 603, and the prediction module 604 may be specified as required. The interaction module 601, the processing module 602, the storage module 603, and the prediction module 604 respectively implement different steps in the query method for a graph database system, to implement all functions of an apparatus for querying the graph database system.

[00130] A person skilled in the art can clearly understand that, for the purpose of convenient and brief description, for detailed working processes of the foregoing components, refer to corresponding content in the foregoing method embodiments. Details are not described herein again.

[00131] The following describes a basic hardware structure in embodiments of this application by using an example.

[00132] This application further provides a computing device 1600. As shown in FIG. 16, the computing device 1600 includes a bus 1602, a processor 1604, a memory 1606, and a communication interface 1608. The processor 1604, the memory 1606, and the communication interface 1608 communicate with each other through the bus 1602. The computing device 1600 may be a server or a terminal device. It should be understood that quantities of processors and memories in the computing device 1600 are not limited in this application.

[00133] The bus 1602 may be a peripheral component interconnect (peripheral component interconnect, PCI) bus, an extended industry standard architecture (extended industry standard architecture, EISA) bus, or the like. Buses may be classified into an address bus, a data bus, a control bus, and the like. For ease of representation, only one line is used to represent the bus in FIG. 16, but this does not mean that there is only one bus or only one type of bus. The bus 1602 may include a path for transferring information between components (for example, the memory 1606, the processor 1604, and the communication interface 1608) of the computing device 1600.

[00134] The processor 1604 may include any one or more of processors such as a central processing unit (central processing unit, CPU), a graphics processing unit (graphics processing unit, GPU), a microprocessor (microprocessor, MP), or a digital signal processor (digital signal

processor, DSP).

[00135] The memory 1606 may include a volatile memory (volatile memory), for example, a random access memory (random access memory, RAM). The processor 1604 may further include a non-volatile memory (non-volatile memory), for example, a read-only memory (read-only memory, ROM), a flash memory, a hard disk drive (hard disk drive, HDD), or a solid-state drive (solid-state drive, SSD).

[00136] The memory 1606 stores executable program code, and the processor 1604 executes the executable program code to separately implement functions of the interaction module 601, the processing module 602, the storage module 603, and the prediction module 604, so as to implement the query method for a graph database system. In other words, the memory 1606 stores instructions for performing the query method for a graph database system.

[00137] The communication interface 1608 uses a transceiver module, for example, but not limited to, a network interface card or a transceiver, to implement communication between the computing device 1600 and another device or a communication network.

[00138] An embodiment of this application further provides a computing device cluster. The computing device cluster includes at least one computing device. The computing device may be a server, for example, a central server, an edge server, or a local server in a local data center. In some embodiments, the computing device may alternatively be a terminal device, for example, a desktop computer, a notebook computer, or a smartphone.

[00139] As shown in FIG. 17, the computing device cluster includes at least one computing device 1600. A memory 1606 in the one or more computing devices 1600 in the computing device cluster may store same instructions for performing the query method for a graph database system.

[00140] In some possible implementations, the memory 1606 in the one or more computing devices 1600 in the computing device cluster may alternatively separately store some instructions for performing the query method for a graph database system. In other words, a combination of the one or more computing devices 1600 may jointly execute the instructions for performing the query method for a graph database system.

[00141] It should be noted that memories 1606 in different computing devices 1600 in the computing device cluster may store different instructions respectively used for performing some functions of the apparatus for querying the graph database system. In other words, the instructions stored in the memories 1606 in different computing devices 1600 may implement functions of one or more of the interaction module 601, the processing module 602, the storage module 603, and the prediction module 604.

[00142] In some possible implementations, the one or more computing devices in the computing device cluster may be connected through a network. The network may be a wide area

network, a local area network, or the like. FIG. 18 shows a possible implementation. As shown in FIG. 18, two computing devices 1600A and 1600B are connected through a network. Specifically, the computing devices are connected to the network through communication interfaces of the computing devices. In such a possible implementation, a memory 1606 in the computing device 1600A stores instructions for performing functions of the interaction module 601 and the processing module 602. In addition, a memory 1606 in the computing device 1600B stores instructions for performing functions of the storage module 603 and the prediction module 604.

[00143] A manner of connection between computing device clusters shown in FIG. 18 may take into consideration that the query method for a graph database system in this application needs to store a large amount of data. Therefore, it is considered that functions implemented by the storage module 603 and the prediction module 604 are performed by the computing device 1600B.

[00144] It should be understood that a function of the computing device 1600A shown in FIG. 18 may alternatively be completed by a plurality of computing devices 1600. Similarly, a function of the computing device 1600B may alternatively be completed by a plurality of computing devices 1600.

[00145] An embodiment of this application further provides another computing device cluster. For a connection relationship among computing devices in the computing device cluster, refer to the connection manners in the computing device cluster in FIG. 17 and FIG. 18 similarly. A difference is that a memory 1606 in one or more computing devices 1600 in the computing device cluster may store same instructions for performing the query method for a graph database system.

[00146] In some possible implementations, the memory 1606 in the one or more computing devices 1600 in the computing device cluster may alternatively separately store some instructions for performing the query method for a graph database system. In other words, a combination of the one or more computing devices 1600 may jointly execute the instructions for performing the query method for a graph database system.

[00147] An embodiment of this application further provides a computer program product including instructions. The computer program product may be software or a program product that includes instructions and that can be run on a computing device or stored in any usable medium. When the computer program product runs on at least one computing device, the at least one computing device is caused to perform a query method for a graph database system.

[00148] An embodiment of this application further provides a computer-readable storage medium. The computer-readable storage medium may be any usable medium that can be stored in a computing device, or a data storage device, for example, a data center, including one or more usable media. The usable medium may be a magnetic medium (for example, a floppy disk, a hard disk, or a magnetic tape), an optical medium (for example, a DVD), a semiconductor medium (for

example, a solid-state drive), or the like. The computer-readable storage medium includes instructions. The instructions instruct the computing device to perform a query method for a graph database system, or instruct the computing device to perform a query method for a graph database system.

[00149] A person of ordinary skill in the art may understand that all or some of the steps of embodiments may be implemented by hardware or a program instructing related hardware. The program may be stored in a computer-readable storage medium. The storage medium may include: a read-only memory, a magnetic disk, or an optical disc.

[00150] It should be noted that information (including but not limited to user equipment information, user personal information, and the like), data (including but not limited to data for analysis, data for storage, data for display, and the like), and signals in this application are all authorized by users or fully authorized by all parties, and collection, use, and processing of related data need to comply with related laws, regulations, and standards of related countries and regions. For example, original data, executable code, and the like in this application are obtained under full authorization.

[00151] In embodiments of this application, the terms "first", "second", and "third" are merely used for description, but cannot be understood as an indication or implication of relative importance. The term "at least one" means one or more, and the term "a plurality of" means two or more, unless otherwise expressly limited.

[00152] The term "and/or" in this application describes only an association relationship for describing associated objects and represents that three relationships may exist. For example, A and/or B may represent the following three cases: Only A exists, both A and B exist, and only B exists. In addition, the character "/" in this specification generally indicates an "or" relationship between the associated objects.

[00153] Finally, it should be noted that the foregoing embodiments are merely intended for describing the technical solutions of this application other than limiting this application. Although this application is described in detail with reference to the foregoing embodiments, a person of ordinary skill in the art should understand that modifications may still be made to the technical solutions described in the foregoing embodiments or equivalent replacements may still be made to some technical features thereof. However, these modifications or replacements do not make the essence of the corresponding technical solutions depart from the protection scope of the technical solutions in embodiments of this application.

CLAIMS

1. A graph database system, wherein the graph database system comprises:

an interaction module, configured to receive a data query request sent by a first user, wherein the data query request indicates to query the graph database system for a specified event; and

a processing module, configured to obtain, based on the data query request, first graph data related to the specified event, wherein the first graph data is graph data obtained by supplementing missing information to second graph data, the second graph data is obtained by querying, based on the data query request, graph data stored in the graph database system, and the graph data indicates a plurality of entities and a relationship among different entities; and

the interaction module is further configured to return a response to the data query request to the first user based on the first graph data, wherein the response carries data that matches the specified event.

2. The system according to claim 1, wherein the response further carries a probability that the data matches the specified event.

3. The system according to claim 1 or 2, wherein

the processing module is configured to: obtain, based on the data query request, the second graph data related to the specified event and target information missing from the second graph data, obtain a value of the target information based on the second graph data and an association relationship between the target information and the specified event, and add the value of the target information to the second graph data, to obtain the first graph data, wherein

the target information indicates one or more of the following: an identifier of the entity, property information of the entity, the relationship among different entities, or property information of the relationship.

4. The system according to claim 3, wherein the graph database system further comprises a storage module and a prediction module, and the storage module is configured to store graph data;

the processing module is configured to: obtain, based on the data query request, the second graph data related to the specified event from the storage module, determine the target information missing from the second graph data, and provide the second graph data and the target information to the prediction module;

the prediction module is configured to: predict the value of the target information based on the second graph data and the target information, add the value of the target information to the second graph data, to obtain the first graph data, and provide the first graph data to the processing module; and

the processing module is configured to obtain the response to the data query request based on

the first graph data.

5. The system according to claim 4, wherein the prediction module is implemented based on a pre-trained machine learning model.

6. The system according to claim 5, wherein

the prediction module is configured to: obtain first computing data pre-computed by using the pre-trained machine learning model, and predict the value of the target information based on the first computing data, the second graph data, and the target information.

7. The system according to claim 6, wherein the machine learning model comprises a deep learning model, the first computing data is a computation result that is obtained by an embedding layer of a pre-trained deep learning model based on graph data used for training, and the first computing data is stored in the storage module as property information of the graph data used for training.

8. The system according to any one of claims 3 to 7, wherein

the interaction module is further configured to receive configuration information set by a second user for the second graph data, wherein the configuration information indicates information that needs to be recorded in the second graph data; and

the processing module is configured to determine, based on the configuration information, the target information missing from the second graph data.

9. The system according to claim 8, wherein the configuration information indicates a type of the information that needs to be recorded in the second graph data, and the type indicates one or more of the following: the identifier of the entity, the property information of the entity, the relationship among different entities, or the property information of the relationship.

10. A query method for a graph database system, wherein the method is applied to a graph database system, and the graph database method comprises:

receiving a data query request sent by a first user, wherein the data query request indicates to query the graph database system for a specified event;

obtaining, based on the data query request, first graph data related to the specified event, wherein the first graph data is graph data obtained by supplementing missing information to second graph data, the second graph data is obtained by querying, based on the data query request, graph data stored in the graph database system, and the graph data indicates a plurality of entities and a relationship among different entities; and

returning a response to the data query request to the first user based on the first graph data, wherein the response carries data that matches the specified event.

11. The method according to claim 10, wherein the response further carries a probability that the data matches the specified event.

12. The method according to claim 10 or 11, wherein the obtaining, based on the data query request, the first graph data related to the specified event comprises:

obtaining, based on the data query request, the second graph data related to the specified event and target information missing from the second graph data, obtaining a value of the target information based on the second graph data and an association relationship between the target information and the specified event, and adding the value of the target information to the second graph data, to obtain the first graph data, wherein

the target information indicates one or more of the following: an identifier of the entity, property information of the entity, the relationship among different entities, or property information of the relationship.

13. The method according to claim 12, wherein the obtaining the value of the target information based on the second graph data and the association relationship between the target information and the specified event comprises:

predicting the value of the target information based on the second graph data and the target information by using a pre-trained machine learning model.

14. The method according to claim 13, wherein the predicting the value of the target information based on the second graph data and the target information by using the pre-trained machine learning model comprises:

obtaining first computing data pre-computed by using the pre-trained machine learning model, and predicting the value of the target information based on the first computing data, the second graph data, and the target information by using the pre-trained machine learning model.

15. The method according to claim 14, wherein the machine learning model comprises a deep learning model, the first computing data is a computation result that is obtained by an embedding layer of a pre-trained deep learning model based on graph data used for training, and the first computing data is stored as property information of the graph data used for training.

16. The method according to any one of claims 12 to 15, wherein the method further comprises:

receiving configuration information set by a second user for the second graph data, wherein the configuration information indicates information that needs to be recorded in the second graph data; and

the obtaining the target information missing from the second graph data comprises:

determining, based on the configuration information, the target information missing from the second graph data.

17. The method according to claim 16, wherein the configuration information indicates a type of the information that needs to be recorded in the second graph data, and the type indicates one

or more of the following: the identifier of the entity, the property information of the entity, the relationship among different entities, or the property information of the relationship.

18. A computing device cluster, comprising a plurality of computing devices, wherein the plurality of computing devices comprise a plurality of processors and a plurality of memories, the plurality of memories store program instructions, and the plurality of processors run the program instructions, to cause the computing device cluster to implement the graph database system according to any one of claims 1 to 9.

19. A computer-readable storage medium, comprising program instructions, wherein when the program instructions are run on a computing device, the computing device is caused to implement the graph database system according to any one of claims 1 to 9.

20. A computer program product comprising instructions, wherein when the instructions are run by a computing device cluster, the computing device cluster is caused to implement the graph database system according to any one of claims 1 to 9.