(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
16 March 2017 (16.03.2017)

WIPO | PCT

(10) International Publication Number
WO 2017/044119 A1

(51) International Patent Classification: *G06F 17/30* (2006.01)

(21) International Application Number: PCT/US2015/049668

(22) International Filing Date: 11 September 2015 (11.09.2015)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **HEWLETT PACKARD ENTERPRISE DEVELOPMENT LP** [US/US]; 11445 Compaq Center Drive West, Houston, TX 77070 (US).

(72) Inventors: **DAS, Mahashweta**; 1501 Page Mill Road, Palo Alto, California 94304-1100 (US). **SIMITSIS, Alkiviadis**; 1501 Page Mill Road, Palo Alto, California 94304-1100 (US). **WILKINSON, William K.**; 1501 Page Mill Road, Palo Alto, California 94304-1100 (US).

(74) Agents: **GONCALVES, Brian** et al.; Hewlett Packard Enterprise, 3404 E. Harmony Road, Mail Stop 79, Fort Collins, CO 80528 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:
— *as to the identity of the inventor (Rule 4.17(i))*
— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

Published:
— *with international search report (Art. 21(3))*

(54) Title: GRAPH DATABASE AND RELATIONAL DATABASE MAPPING

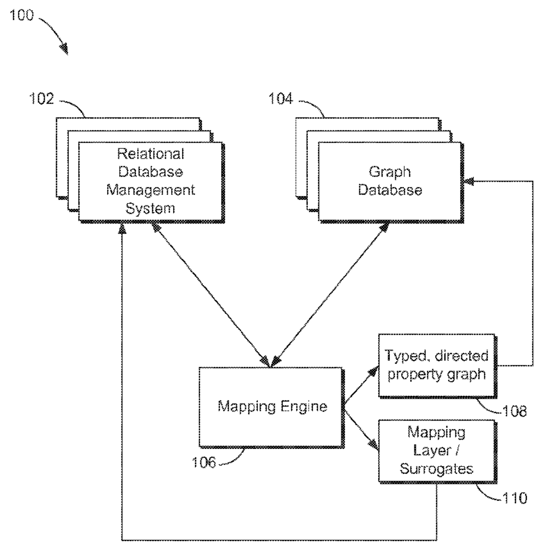
100
102
Relational Database Management System
104
Graph Database
Typed, directed property graph
108
Mapping Engine
106
Mapping Layer / Surrogates
110

FIG. 1

(57) Abstract: Examples for mapping a relational database to a graph database include a mapping engine to execute an arbitrary query on a relational database, identify a result column tag based on a tag of an underlying base table, process the result column into a typed, directed property graph based on the result column tag, and output the typed, directed property graph to a graph database. Examples for mapping a graph database to a relational database include processing a graph transaction by updating a mapping layer with a surrogate describing a change to a database object, determining, for an object in the mapping layer, if a database constraint defined on the object is satisfied, collecting database changes defined by the surrogate into a database change request, submitting the change request to a relational database as a transaction, and deleting the surrogate for the object in the mapping layer.

GRAPH DATABASE AND RELATIONAL DATABASE MAPPING

BACKGROUND

[0001] Computing systems, devices, and electronic components may access, store, process, or communicate with a database or databases. A database may store data or information in various formats, models, structures, or systems, such as in a relational database system or a graph database structure. Users or processes may access or query the databases to fetch or retrieve data in a database, or to transfer or otherwise share information between varying database systems, such as between relational databases and graph database structures.

BRIEF DESCRIPTION OF THE DRAWINGS

[0002] The following detailed description references the drawings, wherein:

[0003] FIG. 1 is a block diagram of a system to map data between a relational database and a graph database, according to an example;

[0004] FIG. 2 is a flowchart of mapping a relational database to a graph database, according to an example;

[0005] FIG. 3 is a flowchart of processing columns from a relational database for mapping to a graph database, according to an example;

[0006] FIG. 4 is a flowchart of mapping a graph database to a relational database; and

[0007] FIG. 5 is a block diagram of a system to map data between a relational database and a graph database, according to an example.

DETAILED DESCRIPTION

[0008] Various examples described below provide for mapping a relational database to a graph database and/or mapping a graph database to a relational database. In an example of mapping a relational database to a graph database, a mapping engine may execute an arbitrary query on a relational database, identify a result column tag based on a tag of an underlying base table, process the result column into a typed, directed property graph based on the result column tag, and output the typed, directed property graph to a graph database.

[0009] In an example of mapping a graph database to a relational database, a graph transaction may be processed by updating a mapping layer with a surrogate describing a change to a database object. For an object in the mapping layer, a determination may be made if a database constraint defined on the object is satisfied. Database changes defined by the surrogate may be collected into a database change request, the change request may be submitted to a relational database as a transaction, and the surrogate for the object in the mapping layer may be deleted.

[0010] As the amount of information stored on computing devices has continued to expand, companies, organizations, and information technology departments have adopted new technologies to accommodate the increased size and complexity of data sets, often referred to as big data. Traditional data processing or database storage systems and techniques such as relational databases or relational database management systems ("RDBMS"), which rely on a relational model and/or a rigid schema, may not be ideal for scaling to big data sets. Similarly, such databases may not be ideal or optimized for handling certain data, such as associative data sets.

[0011] Organizations may employ a graph database to collect, store, query, and/or analyze all or a subset of the organization's data, and in particular large data sets. A graph database may be employed within an organization alone, in combination with other graph databases, or in combination with relational databases or other types of databases. In some examples, a graph database may provide applications with a more natural view of object classes and relationships, and may allow for easier expression of computation.

[0012] However, as organizations may store much of their data in relational database or structured formats, organizations may be faced with the challenge of copying and/or converting data from a relational database to a graph database. In some examples, organizations may also wish to copy or convert data from a graph database to a relational database, or to propagate data back and forth such that data from a relational database can be manipulated in a graph database and then propagated back to the relational database. In either case, the complexity of copying or converting data between the two different database systems and/or structures may be heightened even further.

[0013] FIG. 1 is a block diagram of a system to map data between a relational database and a graph database, according to an example. FIG. 1 may be referred to as database environment 100.

[0014] In the example of FIG. 1, a database environment 100 may comprise a relational database management system or RDBMS 102 and a graph database 104. The relational database 102 may be any database type that employs a relational model, e.g., a collection of tables with each table comprising rows and columns, with a unique key for each row. The graph database 104 may be any database type that employs graph structures to store data using, for example, edges, vertices, and/or properties to represent and/or store data. In one example, a graph may be a typed, directed property graph, e.g., a graph comprising vertices and directed edges where properties (key-value pairs) may be associated with a vertex or an edge and where a vertex may be assigned one or more labels or types. In other examples, a graph may be, e.g., a bipartite graph, a simple undirected graph, or other graph.

[0015] In some examples, both a graph database and a relational database may have a common conceptual model. Otherwise, there may be no correspondence between the two databases and no reason to map between the two databases. In an example, a conceptual model may comprise object classes and relationships among classes. An object class may be typed (e.g., Person, Order) and a relationship may link classes (e.g., fatherOf, ShipTo). Classes and relationships may have associated properties (e.g., Age, shipDate).

Given this model, tables, rows, columns, vertices, edges and properties may model some aspect of the underlying conceptual model.

[0016] RDBMS 102 and graph database 104 may be coupled to or communicate with a mapping engine 106. Mapping engine 106 may allow for the copy or conversion of data between a RDBMS and a graph database in either direction. For example, mapping engine 106 may allow data stored in a relational model, e.g., in tables, columns, and/or rows, to be converted into a graph structure, e.g., as edges, vertices, and/or properties. Similarly, mapping engine 106 may allow data stored in a graph structure, e.g., as edges, vertices, and/or properties to be converted into a relational model, e.g., in tables, columns, and/or rows.

[0017] During the conversion process, the methods described herein may export a typed, directed property graph 108 when converting from a relational database to a graph database, or may export a mapping layer describing surrogate objects for classes and relationships when converting from a graph database to a relational database. Typed, directed property graph 108 may support, for example, n-ary relationships and properties of relationships that may not be expressed in other models, such as the Resource Description Framework ("RDF").

[0018] RDBMS 102, graph database 104, and/or mapping engine 106 may reside in a data center, cloud service, or virtualized server infrastructure (hereinafter "data center"), which may refer to a collection of servers and other computing devices that may be on-site, off-site, private, public, co-located, or located across a geographic area or areas. A data center may comprise or communicate with computing devices such as servers, blade enclosures, workstations, desktop computers, laptops or notebook computers, point of sale devices, tablet computers, mobile phones, smart devices, or any other processing device or equipment including a processing resource. In examples described herein, a processing resource may include, for example, one processor or multiple processors included in a single computing device or distributed across multiple computing devices.

[0019] In the example of FIG. 1, RDBMS 102, graph database 104, and/or mapping engine 106 may reside on a computing device that includes a processing resource and a machine-readable storage medium comprising or encoded with instructions executable by the processing resource, as discussed below in more detail with respect to FIGS. 2-5. In some examples, the instructions may be implemented as engines or circuitry comprising any combination of hardware and programming to implement the functionalities of the engines or circuitry, as described below.

[0020] Database environment 100 may also comprise a synchronization engine (not shown) to extract updates from a graph database and aggregate the updates as relational database transactions. Database environment 100 may also include external connectors, which may be connectors to external systems, processes, or databases, such as a connector to a relational database, legacy system, or other system for ingesting data or exporting data.

[0021] In the example of FIG. 1, RDBMS 102, graph database 104, and/or mapping engine 106 may be directly coupled or communicate directly, or may communicate over a network. A network may be a local network, virtual network, private network, public network, or other wired or wireless communications network accessible by a user or users, servers, or other components. As used herein, a network or computer network may include, for example, a local area network (LAN), a wireless local area network (WLAN), a virtual private network (VPN), the Internet, a cellular network, or a combination thereof.

[0022] FIG. 2 is a flowchart of mapping a relational database to a graph database, according to an example.

[0023] In block 200, in an example, an arbitrary RDBMS query is executed. The query may be received from, e.g., an application, process, tool, script, or other engine for purposes of communicating with a relational database, such as RDBMS 102.

[0024] The query may be executed on a RDBMS with an arbitrary database schema, where each table comprises either descriptive data for one object type, e.g., a class table, or relationship data that links two objects types, e.g., a relationship table. In such examples, each row in a class table may have data for one class object, and each row in a relationship table may have data for one relationship instance.

[0025] In block 204, in an example, each column resulting from the query is identified based on the tag of the underlying base table, e.g., the relational database. For example, the column may be identified as containing or describing a primary key (e.g., a unique identifier for a row), a foreign key (e.g., a column that references another table), class data, relationship data, derived data, or other types of data. Identifying the provenance of a result column may be accomplished by, for example, parsing an execution tree of operators and working down the tree from the result, e.g., the tree root, to find the source operand(s) that correspond to each result column.

[0026] In some examples, block 204 may further comprise identifying whether a column is derived class data or derived relationship data depending on whether the computation is a function of a single class table. Derived data may be, for example, a functional expression or computation over source data, e.g., "price*1.5" or "avg(price)". Some expressions may have a unique inverse, e.g., "price*1.5" while other expressions may not be invertible, e.g., "avg(price)". In some examples, block 204 may also comprise identifying a primary key column corresponding to class data, identifying a corresponding primary key for a foreign key, and/or identifying a primary key and foreign key for relationship data.

[0027] In block 206, each result column or each result row may be processed based on a tag. In some examples, processing the result column comprises processing the data into a typed, directed property graph as shown in FIG. 1. Processing each column of each result row is described in more detail with respect to FIG. 3, described below.

[0028] In block 208, the typed, directed property graph with query results may be output, e.g., from mapping engine 106 to graph database 104 or other output.

[0029] FIG. 3 is a flowchart of processing columns from a relational database for mapping to a graph database, according to an example. In the flow of FIG. 3, each result column or each result row identified in FIG. 2 may be processed, e.g., into a typed, directed property graph as shown in FIG. 1. The query result may create a new graph, or may add to an existing graph.

[0030] In block 302, in the event that a column comprises or describes a primary key, a vertex in a graph may be created using the primary key as the vertex identifier with the type associated with the class of the underlying table, e.g., a person, product, vendor, etc. in a table of the relational database.

[0031] In block 304, in the event that a column comprises or describes class data, a property value may be added to the graph for the vertex associated with the class object. In some examples, block 304 may be based on identification of the vertex for the data, e.g., if the result set includes a column containing the primary key for the table that is the source of the class data.

[0032] In block 306, in the event that a column comprises or describes a foreign key, a vertex may be created in the graph using the foreign key as the vertex identifier and with the type associated with a foreign key class. In some examples, a graph edge may be created from the vertex with the edge associated with the primary key to the vertex associated with the foreign key. In such examples, the edge may be labeled with the relationship.

[0033] In block 308, in the event that a column comprises or describes relationship data, a property value may be added to the edge associated with the relationship. In some examples, block 308 may be based on identification of the objects linked by the relationship, i.e., the result may include columns containing the keys that identify the vertices in the relationship.

[0034] In block 310, in the event that a column comprises or describes derived data, the value may be added as a property to either a vertex or an edge of the graph. In some examples, block 310 may further comprise determining if there is a unique inverse computation.

[0035] In some examples, a class associated with a key or a label associated with an edge may not be unique, e.g., a mapping may be ambiguous. In such examples, metadata from other columns may be referenced for disambiguation.

[0036] FIG. 4 is a flowchart of mapping a graph database to a relational database. A graph action to create a vertex may generate a database action to create a new row in a class table associated with a vertex type. A graph action to delete an edge may generate a database action to either remove a row in the database table that corresponds to the relationship or to nullify a foreign key column in a class table.

[0037] Accordingly, in examples, it may not be possible to convert a graph transaction immediately into a relational database transaction, as the underlying relational database may have consistency constraints relating to the transaction. For example, an object relating to a person may have required columns in a relational database, so a new person vertex in a graph database cannot be immediately created as a new row in a relational database table. In such examples, changes to a graph database may be cached until the consistency constraints are satisfied by way of creating surrogate objects for each class and relationship in a mapping layer, and deleting the surrogates only after the constraints have been satisfied and the graph transaction committed to the relational database.

[0038] Surrogates may be particularly useful in the case where there is a cycle in foreign key references. For example, in the case of three tables, a second table may reference a third table, which in turn may reference a first table. The combination of surrogate objects and caching until database consistency constraints are satisfied through transaction validation may allow for mapping a graph database to a relational database, even in such examples.

[0039] In block 402, a graph transaction comprising a sequence of changes to a graph may be processed. The changes may be, for example, adding a vertex, adding an edge, adding a property value to a vertex or an edge, deleting a vertex, deleting an edge, deleting a property value, or modifying a property value. Each vertex, edge, and/or property value may have associated metadata describing its provenance in the underlying relational database, e.g., an underlying table, column, type, derivation, etc.

[0040] Block 402 may comprise creating, linking, or updating surrogates. The surrogates may be stored or collected in a mapping layer, e.g., mapping layer 110, which may contain a set of surrogates describing changes to class and relationship objects to apply to the relational database, e.g., RDMBS 102.

[0041] For example, if the database change is to add a vertex, if a property-create surrogate exists for the vertex, it may be marked, e.g, as class-create. If no property-create surrogate exists, a new class surrogate corresponding to the vertex type may be created. If a delete surrogate exists, it may be linked to the class-create.

[0042] In the database change is to add an edge, in an example, if a property-create surrogate for the relationship exists, it may be marked as relationship-create. If a surrogate does not exist, a new relationship surrogate may be created for the edge with the source vertex identifier as the primary key and the target vertex as the foreign key, and the relationship may be marked as relationship-create. If a delete surrogate exists, it may be linked to the relationship-create surrogate.

[0043] If the database change is to add a property, a surrogate may be located for the class or relationship using vertex identifiers, and the property value may be added. If a surrogate does not exist, a new surrogate may be created and marked as property-crate. If a delete surrogate exists for a class or relationship, it may be linked to the new property surrogate.

[0044] If the database change is to delete a vertex, if a property-delete surrogate exists, it may be marked as vertex-delete. If the surrogate does not exist, a new class surrogate may be created and marked as, e.g., class-delete. If a class-create or property-create surrogate exists for the vertex identifier, it may be linked to the delete surrogate.

[0045] If the database change is to delete an edge, if a property-delete surrogate exists, it may be marked as relationship-delete. If the surrogate does not exist, a new class surrogate may be created and marked as, e.g., relationship-delete. If a class-create or relationship-create surrogate exists for the relationship, it may be linked to the delete surrogate.

[0046] If the database change is to delete a property, the surrogate for the class or relationship may be located using, e.g., vertex identifiers. If a surrogate for the property does not exist, a new surrogate may be created and marked as, e.g., property-delete. If an existing create surrogate exists for the class or relationship, it may be linked to the new surrogate.

[0047] If the database change is to modify a property, the surrogate for the class or relationship may be located using, e.g., vertex identifiers. If a surrogate for the property does not exist, a new surrogate may be created and marked as, e.g., property-modify. If an existing create surrogate exists for the class or relationship, it may be linked to the new surrogate.

[0048] In examples where, for example, a vertex is created, deleted, and then created, a plurality of surrogates for the object may exist. In some examples, the plurality of surrogates may be linked in order of time.

[0049] In block 404, the transactions reflected in the mapping layer, e.g., a set of surrogates describing changes to class and relationship objects, may be validated. Transaction validation may comprise determining whether a change may violate a database consistency constraint in the RDBMS. In examples, transaction validation may run independently from a mapping engine, and may be invoked periodically or after processing a graph transaction in block 402.

[0050] In block 406, a validation engine may start with the oldest surrogate in the mapping layer and check if constraints defined on the object are satisfied. If the constraints are not satisfied, the transactions should not be submitted to the RDBMS. If the constraints are satisfied, the changes defined by the surrogate may be collected into database change requests in block 408 and submitted or committed to the database, e.g., to a RDBMS, in block 410. In some examples, the surrogates may then be deleted or removed from the mapping layer in block 412.

[0051] FIG. 5 is a block diagram of a system to map data between a relational database and a graph database, according to an example.

[0052] The computing system 500 of FIG. 5 may comprise a processing resource or processor 502. As used herein, a processing resource may be at least one of a central processing unit (CPU), a semiconductor-based microprocessor, a graphics processing unit (GPU), a field-programmable gate array (FPGA) configured to retrieve and execute instructions, other electronic circuitry suitable for the retrieval and execution of instructions stored on a machine-readable storage medium, or a combination thereof. Processing resource 502 may fetch, decode, and execute instructions, e.g., instructions or engine 510, stored on memory or storage medium 504 to perform the functionalities described herein. In examples, the functionalities of any of the instructions of storage medium 504 may be implemented in the form of electronic circuitry, in the form of executable instructions encoded on a machine-readable storage medium, or a combination thereof.

[0053] As used herein, a "machine-readable storage medium" may be any electronic, magnetic, optical, or other physical storage apparatus to contain or store information such as executable instructions, data, and the like. For example, any machine-readable storage medium described herein may be any of Random Access Memory (RAM), volatile memory, non-volatile memory, flash memory, a hard drive, a solid state drive, any type of storage disc or optical disc, and the like, or a combination thereof. Further, any machine-readable storage medium described herein may be non-transitory.

[0054] System 500 may also include persistent storage and/or memory. In some examples, persistent storage may be implemented by at least one non-volatile machine-readable storage medium, as described herein, and may be memory utilized by system 500. In some examples, a memory may temporarily store data portions while performing processing operations on them, such as for managing a graph database or mapping between databases.

[0055] In examples described herein, a machine-readable storage medium or media is part of an article or article of manufacture. An article or article of manufacture may refer to any manufactured single component or multiple components. The storage medium may be located either in the computing device executing the machine-readable instructions, or remote from but accessible to the computing device (e.g., via a computer network) for execution.

[0056] In some examples, instructions or engine 510 may be part of an installation package that, when installed, may be executed by processing resource 502 to implement the functionalities described herein in relation to instructions or engine 510. In such examples, storage medium 504 may be a portable medium or flash drive, or a memory maintained by a server from which the installation package can be downloaded and installed. In other examples, instructions or engine 510 may be part of an application, applications, or component(s) already installed on a computing device including a processing resource, e.g., a computing device running any of the components of graph database environment 100 of FIG. 1.

[0057] System 500 may also include a power source 506 and a network interface device 508, as described above, which may receive data such as data from engines 512 and 514, e.g., via direct connection or a network.

[0058] As discussed above, the instructions 510 in or on the memory or machine-readable storage of system 500 may comprise an engine. In the engine of block 510, the instructions may map a relational database to a graph database and map the graph database to the relational database after a change to the graph database. A determination may be made, for an object changed in the graph database, whether a database constraint defined on the object in the relational database is satisfied, and in the event that the constraint defined on the object is not satisfied, the change may be cached as a surrogate object in a mapping layer until the constraint is satisfied.

[0059] Although the instructions of FIGS. 2-5 show a specific order of performance of certain functionalities, the instructions of FIGS. 2-5 are not limited to that order. For example, the functionalities shown in succession may be performed in a different order, may be executed concurrently or with partial concurrence, or a combination thereof.

[0060] All of the features disclosed in this specification, including any accompanying claims, abstract and drawings, and/or all of the elements of any method or process so disclosed, may be combined in any combination, except combinations where at least some of such features and/or elements are mutually exclusive.

CLAIMS

What is claimed is:

1. A database management system for mapping a relational database to a graph database, comprising:

a relational database connection;

a graph database connection; and

a mapping engine to:

execute an arbitrary query on a relational database via the relational database connection,

identify a tag of a result column based on a tag of an underlying base table,

process the result column into a typed, directed property graph based on the result column tag, and

output the typed, directed property graph to a graph database.

2. The system of claim 1, wherein when the result column tag is a primary key, the mapping engine is to create a vertex in the graph using the primary key as an identifier with a type associated with a class.

3. The system of claim 1, wherein when the result column tag is class data, the mapping engine is to add a property value into the graph for a vertex associated with the class data.

4. The system of claim 1, wherein when the result column tag is a foreign key, the mapping engine is to create a vertex in the graph using the foreign key as an identifier with a type associated with a foreign key class.

5. The system of claim 1, wherein when the result column tag is relationship data, the mapping engine is to add a property value into the graph for an edge associated with the relationship data.

6. The system of claim 1, wherein when the result column tag is derived data, the mapping engine is to add a property value into the graph for a vertex.

7. The system of claim 1, wherein when the result column tag is derived data, the mapping engine is to add a property value into the graph for an edge.

8. A method for mapping a graph database to a relational database, comprising:

processing a graph transaction by updating a mapping layer with a surrogate describing a change to a database object;

determining, for an object in the mapping layer, if a database constraint defined on the object is satisfied; and

when the constraint defined on the object is satisfied, collecting database changes defined by the surrogate into a database change request, submitting the change request to a relational database as a transaction, and deleting the surrogate for the object in the mapping layer.

9. The method of claim 8, further comprising caching graph transactions until the database constraint defined on the object is satisfied.

10. The method of claim 8, wherein the surrogate further describes a change to a class.

11. The method of claim 8, wherein the surrogate further describes a change to a relationship.

12. An article comprising at least one non-transitory machine-readable storage medium comprising instructions executable by a processing resource of a graph database system to:

map a relational database to a graph database;

map the graph database to the relational database after a change to the graph database;

determine, for an object changed in the graph database, if a database constraint defined on the object in the relational database is satisfied; and

in the event that the constraint defined on the object is not satisfied, cache the change as surrogate object in a mapping layer until the constraint is satisfied.

13. The article of claim 12, wherein the surrogate describes a change to a class.

14. The article of claim 12, wherein the surrogate describes a change to a relationship.

15. The article of claim 12, further comprising a plurality of surrogates for the object linked in order of time.

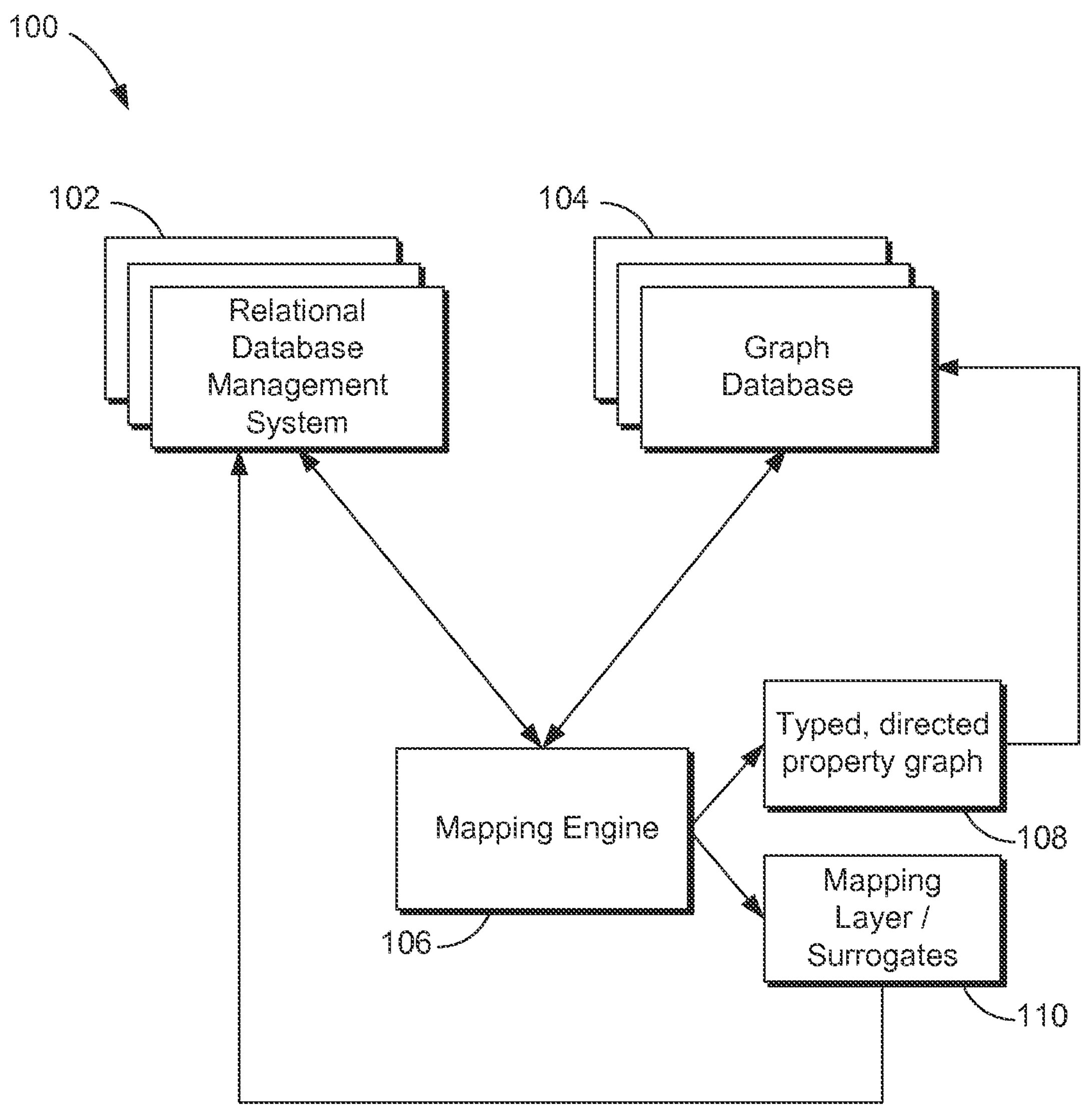
100
102
Relational Database Management System
104
Graph Database
Mapping Engine
106
Typed, directed property graph
108
Mapping Layer / Surrogates
110

FIG. 1

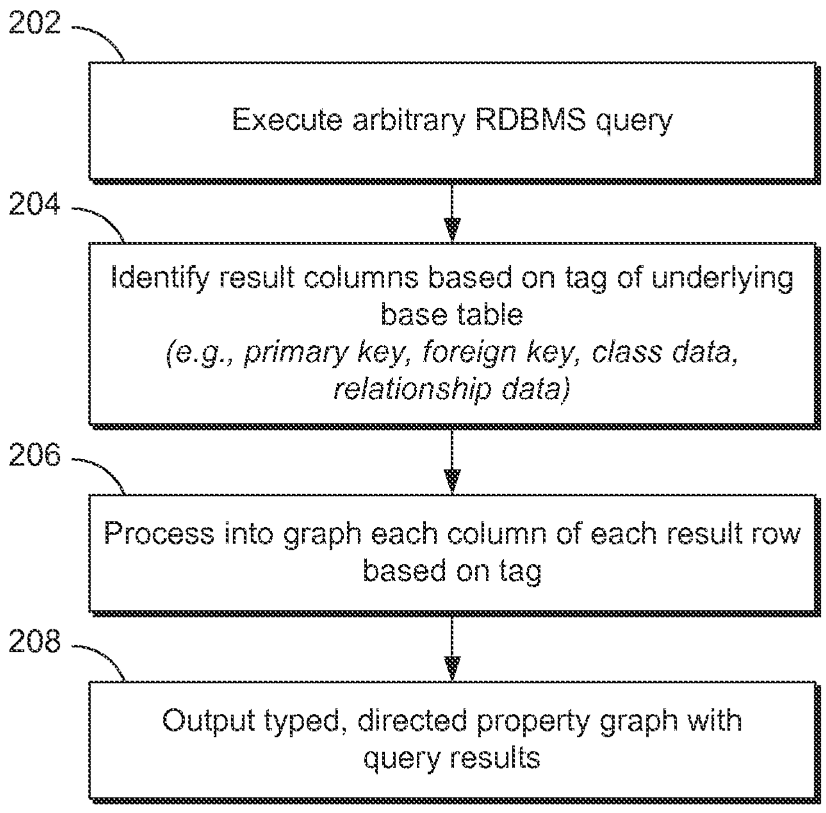
202
Execute arbitrary RDBMS query
204
Identify result columns based on tag of underlying base table
(e.g., primary key, foreign key, class data, relationship data)
206
Process into graph each column of each result row based on tag
208
Output typed, directed property graph with query results

FIG. 2

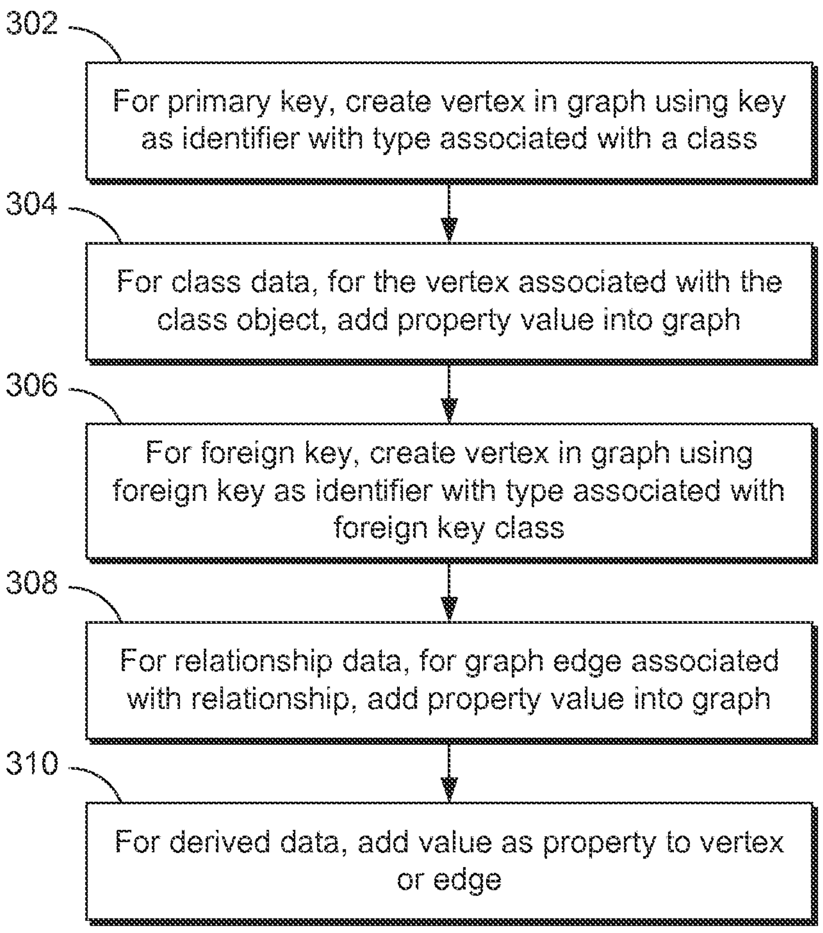
302
For primary key, create vertex in graph using key as identifier with type associated with a class
304
For class data, for the vertex associated with the class object, add property value into graph
306
For foreign key, create vertex in graph using foreign key as identifier with type associated with foreign key class
308
For relationship data, for graph edge associated with relationship, add property value into graph
310
For derived data, add value as property to vertex or edge

FIG. 3

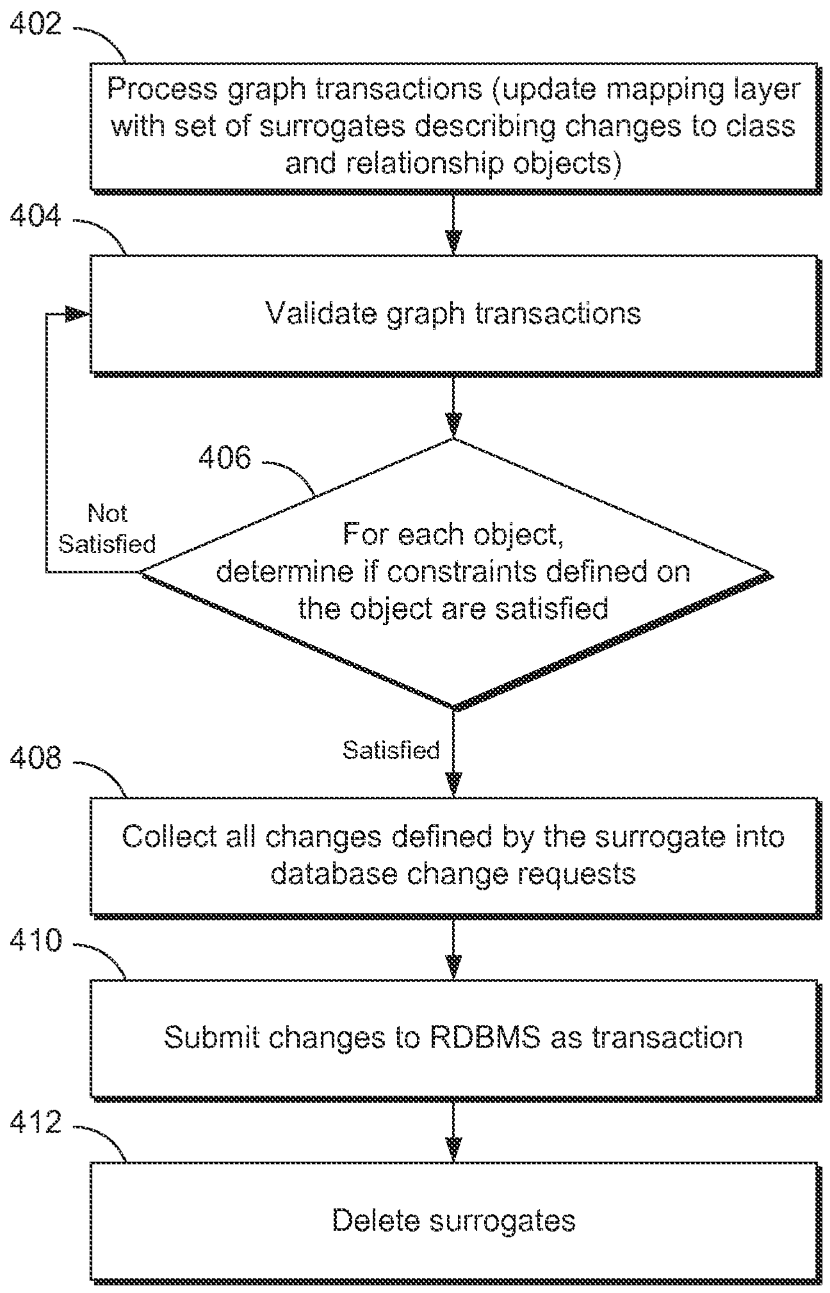
402
Process graph transactions (update mapping layer with set of surrogates describing changes to class and relationship objects)
404
Validate graph transactions
406
For each object, determine if constraints defined on the object are satisfied
Not Satisfied
Satisfied
408
Collect all changes defined by the surrogate into database change requests
410
Submit changes to RDBMS as transaction
412
Delete surrogates

FIG. 4

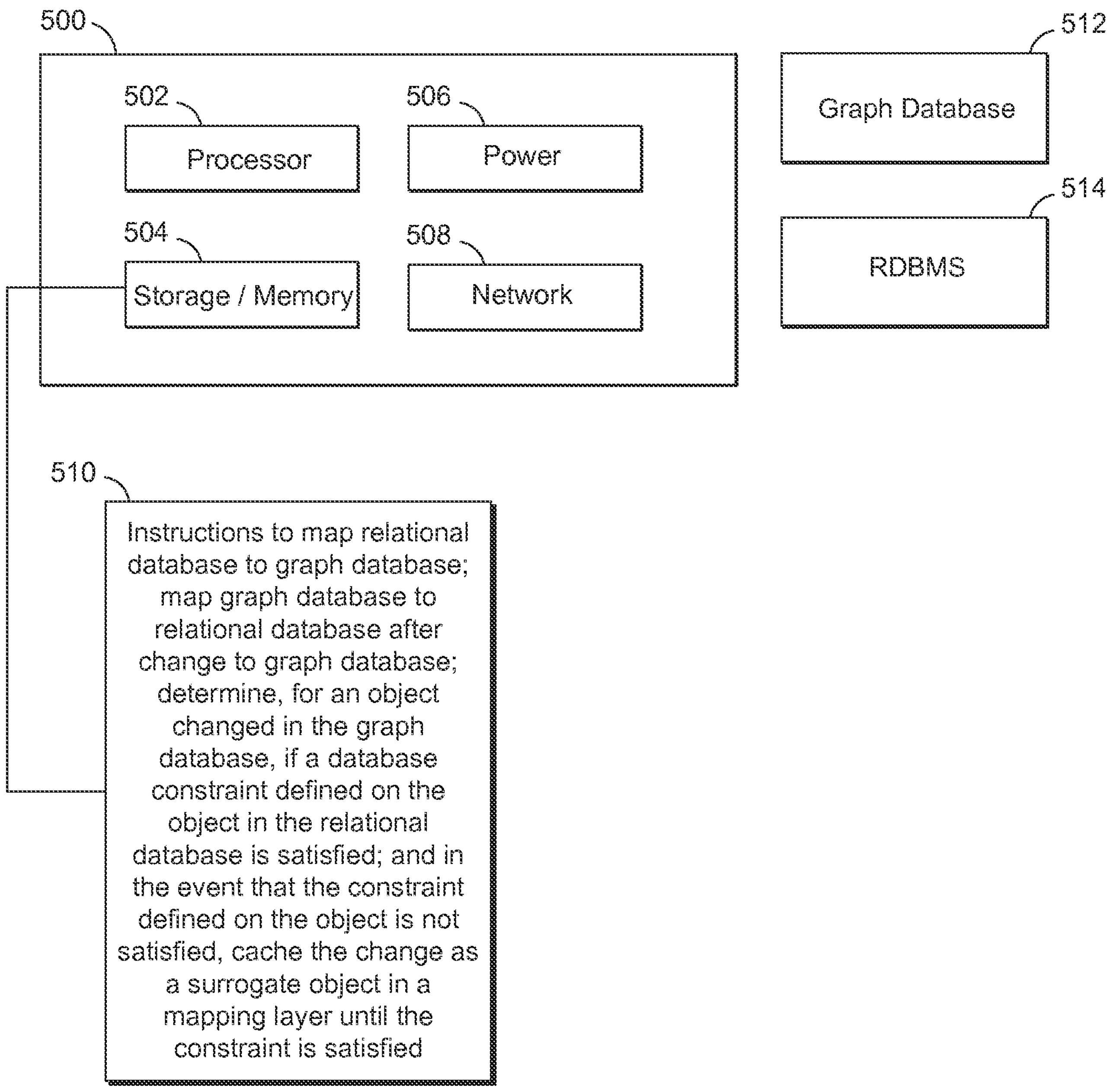
500
502
Processor
506
Power
504
Storage / Memory
508
Network
512
Graph Database
514
RDBMS
510
Instructions to map relational database to graph database; map graph database to relational database after change to graph database; determine, for an object changed in the graph database, if a database constraint defined on the object in the relational database is satisfied; and in the event that the constraint defined on the object is not satisfied, cache the change as a surrogate object in a mapping layer until the constraint is satisfied

FIG. 5

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2015/049668

A. CLASSIFICATION OF SUBJECT MATTER

G06F 17/30(2006.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06F 17/30

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
Korean utility models and applications for utility models
Japanese utility models and applications for utility models

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
eKOMPASS(KIPO internal) & Keywords: relational, graph, database, mapping, result column tag, change, update, surrogate, and similar terms.

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 2014-0201234 A1 (FUJITSU LIMITED) 17 July 2014 See paragraphs [0051], [0064]-[0065], [0070], [0072], and [0076]; claims 1, 3, and 6; and figures 1-3.	1-7
A		8-15
Y	US 2007-0022103 A1 (MICHAEL RYS et al.) 25 January 2007 See paragraphs [0041], [0045]-[0047], and [0070]-[0071]; and figures 2-3.	1-7
Y	US 2015-0120775 A1 (MICROSOFT CORPORATION) 30 April 2015 See paragraph [0028] and figure 1.	2,4
X	US 2006-0004851 A1 (STEVEN ALLEN GOLD et al.) 05 January 2006 See paragraphs [0029]-[0030] and [0041]-[0048]; and figures 1, 3 and 9-10.	8-15
A	US 2013-0339385 A1 (HOWARD A. ABRAMS et al.) 19 December 2013 See paragraphs [0025]-[0030] and figure 1.	1-15

☐ Further documents are listed in the continuation of Box C. ☒ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents,such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search 08 July 2016 (08.07.2016)	Date of mailing of the international search report **11 July 2016 (11.07.2016)**
Name and mailing address of the ISA/KR International Application Division Korean Intellectual Property Office 189 Cheongsa-ro, Seo-gu, Daejeon, 35208, Republic of Korea Facsimile No. +82-42-481-8578	Authorized officer LEE, EUN KYU Telephone No. +82-42-481-3580

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/US2015/049668

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2014-0201234 A1	17/07/2014	EP 2755148 A1	16/07/2014
		JP 2014-137820 A	28/07/2014
US 2007-0022103 A1	25/01/2007	US 2004-0199524 A1	07/10/2004
		US 2005-0004896 A1	06/01/2005
		US 6620138 B1	16/09/2003
		US 6708164 B1	16/03/2004
		US 7213017 B2	01/05/2007
		US 7444321 B2	28/10/2008
		US 7730048 B2	01/06/2010
US 2015-0120775 A1	30/04/2015	None	
US 2006-0004851 A1	05/01/2006	US 7493335 B2	17/02/2009
		WO 2006-014228 A1	09/02/2006
US 2013-0339385 A1	19/12/2013	US 8977646 B2	10/03/2015