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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
28 January 2016 (28.01.2016)

WIPO | PCT

(10) International Publication Number
WO 2016/012903 A1

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

(21) International Application Number:
PCT/IB2015/055343

(22) International Filing Date:
15 July 2015 (15.07.2015)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
PI2014002164 23 July 2014 (23.07.2014) MY

(71) Applicant: **MIMOS BERHAD** [MY/MY]; Technology Park Malaysia, Kuala Lumpur, 57000 (MY).

(72) Inventors: **SINGH, Keeratpal**; c/o MIMOS Berhad, Technology Park Malaysia, Kuala Lumpur, 57000 (MY). **KARUPPIAH, Ettikan Kandasamy**; c/o MIMOS Berhad, Technology Park Malaysia, Kuala Lumpur, 57000 (MY). **MAT NOR, Fazli**; c/o MIMOS Berhad, Technology Park Malaysia, Kuala Lumpur, 57000 (MY).

(74) Agents: **KANDIAH, Geetha** et al.; KASS INTERNATIONAL SDN. BHD., Suite 8-7-2, Menara Mutiara Bangsar, Jalan Liku Off Jalan Riong, Bangsar, Kuala Lumpur, 59100 (MY).

(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,

[Continued on next page]

(54) Title: A SYSTEM FOR QUERYING HETEROGENEOUS DATA SOURCES AND A METHOD THEREOF

(57) Abstract: A system (100) and a method for querying heterogeneous data sources have been disclosed. The system (100) based on machine learning and classification rules, predicts a best route for querying at least one data source from the heterogeneous data sources. The system (100) minimizes inefficient use of hardware resources and obtains near real time response from the data source selected through the predicted route for executing the query.

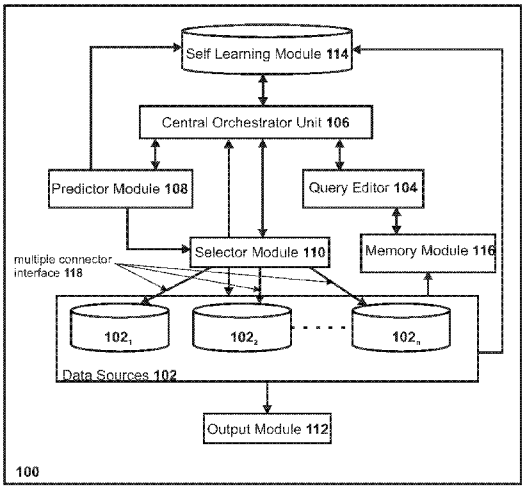


FIGURE 1

WO 2016/012903 A1

SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

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

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))*

A SYSTEM FOR QUERYING HETEROGENEOUS DATA SOURCES AND A METHOD THEREOF

FIELD OF THE INVENTION

The present invention relates to the field of data warehousing and data mining, particularly to a system and method for processing queries for heterogeneous data sources incorporated within a data warehouse.

BACKGROUND

In today's fierce market, enterprises are constantly relying on analytical and strategic decision making for staying competitive. These complex decisions rely on mining large volumes and varied sources of data for providing necessary analytics. Although, the data comes in various forms and volumes it has to be processed to provide analytical results within real-time response constraints.

These sources of data have volume ranging from big data, medium sized data and small data. Traditional relational databases such as MySQL do not have capability to cope up with increasing demands of analytics especially for providing near real time query response for big data. Likewise, when the data is small, using big data tools such as Hadoop, MapReduce, parallel distributed solutions and algorithms written for general-purpose computing on graphics processing units are not fruitful, and lead to inefficient usage of resources since big data clusters generate overheads in processing time to a query. When data is small or manageable, Relational Database Management System (RDBMS) still process data efficiently fast.

Hence, choosing big data processing technology for querying small data sets, or choosing RDBMS to perform real time queries on big data results in inefficient use of resources which leads to low output performance, inefficient use of resource time and latency in observing simulating parameter changes, latency of obtaining real time BI (Business Intelligence) reports and latency of generating analytics result.

Usage of different data sources through various databases (or file system) of different data sizes for the purpose of real time analytics and simulation, leads to longer response time to achieve query results due to inefficient selection of resource models.

There is therefore felt a need for a system and method for querying heterogeneous data sources, that is data sources having different volumes, resources and forms and obtaining analytics and/or results which meet near real time response constraints.

DEFINITIONS

The terms used throughout this specification are defined as follows, unless otherwise limited in specific instances.

The expression 'heterogeneous data sources' used hereinafter in the specification relates to a plurality of data sets or file systems having different sizes or volumes. For example, the data source may represent 'Big Data' clusters, small / medium relational databases or the like.

The expression 'query' used hereinafter in the specification relates to a structured and syntactical coding language used for retrieving information from a data source / file system.

The expression 'route' used hereinafter in the specification relates to an interface which connects to a set of resources including a data source such as a database; hardware including Random Access Memory (RAM), processor, and graphics processor and software such as operating system / file system.

The expression 'class of service' used hereinafter in the specification relates to an indicator which determines processing time based on the type of application/ query. It also determines priority for processing a query and a time within which a response for that query must be obtained. For instance, class of service may be categorized as Class 1, Class 2, Class 3 and Class 4, wherein Class 1 gets highest priority and represents applications including simulation, prescriptive analytics, BI real time reports and web response which must be processed within 15 seconds; Class 2 gets second priority and represents application such as generation of Big Data static reports which must be

generated within 60 seconds; Class 3 gets third priority and represents batch processing applications which must be processed within a time range between one minute and a hour; and Class 4 gets lowest priority and represents applications which take more than an hour.

The expression 'machine readable media' used hereinafter in the specification refers to RAM, ROM, EPROM, EEPROM, CD-ROM or other optical disk storage, magnetic disk storage or other magnetic storage devices, or any other medium which can be used to carry or store desired program code in the form of machine-executable instructions or data structures and which can be accessed by a general purpose or special purpose computer or other machine with a processor.

The expression 'computer program product' is defined as a manufactured product embodied in a machine readable medium as defined herein above.

These definitions are in addition to those expressed in the art.

SUMMARY

Inefficient selection of resources for querying data sources of varied sizes, for the purpose of real time analytics and simulation led to longer response time to achieve query results. This drawback of the prior art led the present invention to propose a system for querying heterogeneous data sources, wherein a response to the query for obtaining analytics and/or simulation results meets near real time response constraints.

The proposed system uses machine learning and classification rules for predicting a best data source to execute a query in order to be able to minimize the response time for meeting real time response constraints.

In accordance with this invention, the system comprises a plurality of data sources, wherein each data source has a discrete volume, discrete hardware resources and tools for facilitating processing of queries. The system also comprises a central orchestrator unit to select at least one machine learning module for processing of queries; a predictor module to receive at least one query for retrieving information from at least one of the data sources; a selector module having a multiple connector interface for

connecting with each of the data sources; and an output module cooperating with the data sources.

According to this invention, the predictor module predicts discrete processing time for executing the query for available routes in the system based on the machine learning module selected by the central orchestrator unit. Subsequently, the selector module compares the predicted processing time for each route generated by the predictor module with the class of service of the query. The selector module selects an appropriate data source for execution of the query via the interface based on the comparison of the predicted processing time for each route and class of service. The query is then processed by the selected data source within the processing time predicted for that route and the query results are displayed on the output module.

Additional aspects, features and advantages of the invention will become apparent to those skilled in the art upon consideration of the following detailed description of preferred embodiments of the invention in conjunction with the drawings listed below.

BRIEF DESCRIPTION OF THE ACCOMPANYING DRAWINGS

A system and method for querying heterogeneous data sources, in accordance with the present invention will now be described with the help of the accompanying drawings, in which:

FIGURE 1 discloses a schematic of the system for querying heterogeneous data sources in accordance with the present invention;

FIGURE 2 is a flowchart showing the steps for querying heterogeneous data sources in accordance with the present invention;

FIGURE 3 is a flowchart showing the steps performed by a query editor in accordance with this invention; and

FIGURE 4 is a flowchart showing the steps performed by a central orchestrator unit for selecting a machine learning module for improving accuracy of selecting a best route for queries in accordance with this invention.

DETAILED DESCRIPTION

The system for querying heterogeneous data sources, of the present disclosure will now be described with reference to the embodiment shown in the accompanying drawings. The embodiment does not limit the scope and ambit of the disclosure. The description relates purely to the exemplary embodiment and its suggested applications.

The embodiment herein and the various features and advantageous details thereof are explained with reference to the non-limiting embodiment in the following description. Descriptions of well-known components and processing techniques are omitted so as to not unnecessarily obscure the embodiments herein. The examples used herein are intended merely to facilitate an understanding of ways in which the embodiment herein may be practiced and to further enable those of skill in the art to practice the embodiment herein. Accordingly, the description should not be construed as limiting the scope of the embodiment herein.

The description herein after, of the specific embodiment will so fully reveal the general nature of the embodiments herein that others can, by applying current knowledge, readily modify and/or adapt for various applications such specific embodiment without departing from the generic concept, and, therefore, such adaptations and modifications should and are intended to be comprehended within the meaning and range of equivalents of the disclosed embodiments. It is to be understood that the phraseology or terminology employed herein is for the purpose of description and not of limitation.

Additional aspects, features and advantages of the invention will become apparent to those skilled in the art upon consideration of the following detailed description of preferred embodiments of the invention in conjunction with the drawings listed below

The present invention proposes a system and a method for querying heterogeneous data sources. The proposed system performs predictive planning via machine learning to predict and select a best proposed route for executing a query within a minimum response time bounded by benchmarks (predicted processing time).

The following detailed description of the preferred embodiments will now be described in accordance with the attached drawings, either individually or in combination.

FIGURE 1 of the accompanying drawings discloses a system (100) for querying heterogeneous data sources ($102_1 ... 102_n$), collectively represented by reference numeral (102), and **FIGURE 2** discloses a method for querying heterogeneous data sources (102).

Referring to **FIGURE 1** and **FIGURE 2**, the system (100) comprises the following components: a plurality of data sources (102); a Central Orchestrator Unit (COU) (106); a query editor (104); a predictor module (108); a selector module (110); and an output module (112). Each of these components and their operation will be described in detail hereinafter.

The data sources (102) include one or more tables that hold requisite information for performing simulation, analytics or similar complex analysis. Each of the data sources (102) have a discrete volume, discrete hardware resources and tools for facilitating processing of queries. For instance, the data source (102_1) may be a Relational Database Management System (RDMS) holding small and medium sized data and data source (102_2) may be a Big Data cluster.

The query editor (104) accepts queries for retrieving information from the data sources (102), step (2000) of **FIGURE 2**.

FIGURE 3 discloses the steps performed by a query editor (104) in accordance with this invention.

Apart from accepting queries (3000), the query editor (104) preemptively corrects and completes a query while it is being typed using information from a memory module (116). The memory module (116) is communicably linked to the data sources (102) for holding metadata of the data sources (102) and any previously executed queries. The query editor (104) uses the metadata information stored in the memory module (116) to check syntax of a query to determine if typed column names exist within a table name chosen for a data source (102), and also suggests column names or strings while the query is being typed (3002).

In addition, the query editor (104) checks the query for different syntax as the query may be executed on any one of the available data sources (102). For instance, the query editor (104) checks the syntax for query languages/tools such as Postgres or Hive (Hadoop) or SQL parser of GPU. Further, in case any error is identified in the query, then the query editor (104) highlights appropriate portions of the query with a distinctive color (such as red color) and/or an error message to indicate occurrence of the error (3002). When the query is error-free, the query editor (104) invokes the COU (106) for execution of the query (3004). Typically, the query editor (104) interface includes an execute command button. Once the execute command button is pressed, the query editor (104) invokes the COU (106) for execution of the query.

The COU (106) receives the request from the query editor (104) and facilitates in selection of a best route for a query by selecting at least one machine learning module.

The COU (106) firstly identifies a data source (102) which includes the table name appearing in the query. Once the data source (102) has been identified, the COU (106) checks the volume of that data source (102) to determine if the volume is small, general (medium) or big. The COU (106) directly routes the query to the identified data source (102) if the volume of the data source is classified as small. The query is then executed at the data source (102) and results are displayed on the output module (112). If the volume is not small and the table name is appearing in all the available data sources then COU (106) forwards the query to the predictor module (108).

The predictor module (108) uses the machine learning module selected by COU (106) to predict processing time for executing a query for each of the available data sources (102), step (2004) of **FIGURE 2**. The predictor module (108) calculates the processing time required by each of the data sources (102_1), (102_2) to (102_n) for executing the received query. The machine learning modules are a combination of at least one regression model and one classification model to facilitate in carrying out the prediction for each route. These machine learning modules include at least one module from regression analysis, decision trees and neural networks. While performing prediction, the predictor module (108) also takes into account loading capabilities of the data sources (102) and current load and utilization of available hardware resources.

The predictor module (108) forwards the processing time predicted for each of the available data sources / routes to the selector module (110). The selector module (110) receives the 'class of service' classification information from the COU (106). The selector module (110) then designates a 'class of service' to the query based on the classification rules and further compares the predicted processing time for each route with the 'class of service' of the query, step (2006) of **FIGURE 2**. The selector module (110) based on result of the comparison selects a route / data source (102) having the lowest processing time for executing the query, step (2008) and (2010) of **FIGURE 2**.

The selector module (110) has a multiple connector interface (118) for connecting to each of the data sources (102). The selector module (110) via the interface (118) sends the query to the selected data source (102) for execution. The execution of the query is carried out using tools and techniques well known in the art.

In accordance with another embodiment of the present invention, the COU (106) facilitates in final selection of a data source (102) in the event that the selector module (110) is unable to find a suitable data source (102)/ route for execution of the query. To address this situation, the COU (106) simultaneously sends the query to all available data sources based on the highest class of service of the query requested. The query is then simultaneously executed at the available data sources (102). And results

obtained from a data source (102) that has executed the query first are displayed on the output module (112).

The selected data source (102) then executes the query within the predicted processing time and returns the results to the output module (112) as per a required format, step (2012) of **FIGURE 2**.

In accordance with an aspect of the present invention, COU (106) performs additional functions to ensure accuracy of the system (100) and efficient use of resources. The functions are explained hereinafter.

The COU (106) includes a scheduler (not shown in the figures) for triggering the functions performed by it, which include: maintaining and classifying the data sources; monitoring utilization of the hardware resources; initiating a self-learning process based on new training data; and testing accuracy of the machine learning module. The scheduler carries out the aforementioned activities during off-peak hours.

I. Initiating A Self-Learning Process Based On New Training Data

At each stage of processing and routing of queries, feedback and/or execution details are saved in a self-learning module (114). The self-learning module (114) is a database which stores the processing details to enable COU (106) to initiate a self-learning process and test accuracy of the machine learning modules.

In the event that any query failed to execute within the predicted processing time then execution details of the query along with the predicted processing time and class of service are appended to a training table (not shown in the figures) in the self-learning module (114) as new training data.

Referring to **FIGURE 4** which discloses the steps followed by the COU (106) in initiating self-learning rules to re-train and correct conflicting data by selecting various models of machine learning for the purpose of improving accuracy of selecting the best route when queries are generated.

The COU (106) fetches the new training data from the self-learning module (114) for initiating the self-learning process. The COU (106) selects a machine learning module to evaluate accuracy of processing the failed queries using the new machine learning module (4000). The COU (106) then re-evaluates accuracy of processing the failed queries using a different machine learning module (4002). The accuracy of machine learning modules from step (4000) and (4002) are then compared (4006) to select a machine learning module. The machine learning modules with Area of Receiver Operator Characteristic (ROC) close to 1 are chosen. The selected machine learning module is thereafter used by the predictor module (108).

II. Testing accuracy of the machine learning module

On selection of a new machine learning module, the COU (106) regularly checks the system for any failed queries. If there is no improvement in accuracy, the previously existing machine learning module is selected for use by the predictor module (108). If after several days, accuracy further reduces, then the COU (106) randomly chooses one of the previously used machine learning modules having accuracy (>90%) and ROC closer to 1. If accuracy still degrades (<80%) after a few days for instance, 3 days after using different models, the entry of new training data are compared for “near” conflicting attributes with almost similar attributes in the training table which produce different selection of a data source / route. Then the instances are run in real time by the COU (106) to verify and take one instance that matches the entry in the self-learning module (114) for training data pertaining to regression machine learning modules. For instance, if a first regression model takes lesser time than a second regression model then the first regression model is selected by the COU (106) for prediction. Thereafter, the records in the training table which have “near” conflicting attributes and use the second regression model are eliminated. In this manner the machine learning module is self-corrected.

Still further, the COU (106) checks Linear Regression prediction for processing time for daily queries on each data source (102) during execution of queries and compares it with real query processing time. Error difference for each prediction is difference of time

predicted with real time. If error difference is above 10%, then the selected route's training data is updated with new self-learning queries for each data source (102). The COU (106) performs the aforementioned functions for testing the accuracy of the machine learning modules to ensure that an optimum route for executing the query is selected.

III. Maintaining and classifying the data sources

The COU (106) performs daily checks on all tables of the data sources (102). The COU (106) classifies the tables based on their volume into big data, general data and small data categories. For instance, if tables exceed 100GB, then they are classified as big data and tables below 500MB as classified as small or general data.

Furthermore, the COU (106) balances the volume of the data sources (102). If data is not small, then the scheduler initiates import of data from the data sources on incremental basis such as every 10 minutes to a file.

IV. Monitoring utilization of the hardware resources

The COU (106) continuously monitors real time server load for all the data sources and their hardware resources.

In addition, the COU (106) generates emails and reports for system administrators if it detects any one of the following conditions: (i) server loading issues during processing of a query, (ii) if queries appear to be taking longer time for execution than predicted, (iii) data sources are unable to meet the class of service requirements or (iv) if accuracy issues appear in routing the queries.

The technical / economical advancements offered by one or more aspects of the system (100) for querying heterogeneous data sources (102), include the realization of:

- routing of a query to an optimum data source (102) having the required resources; and
- execution of the query within a time bound by benchmarks determined by a machine learning module selected by the COU (106).

The present invention thereby achieves the objectives of efficient utilization of the available resources, usage of an optimum data source to execute the query and achieve results/ output of the query within near real time constraints.

The terminology used herein is for the purpose of describing particular example embodiments only and is not intended to be limiting. As used herein, the singular forms "a", "an" and "the" may be intended to include the plural forms as well, unless the context clearly indicates otherwise. The terms "comprises," "comprising," “including,” and “having,” are inclusive and therefore specify the presence of stated features, integers, steps, operations, elements, and/or components, but do not preclude the presence or addition of one or more other features, integers, steps, operations, elements, components, and/or groups thereof. The method steps, processes, and operations described herein are not to be construed as necessarily requiring their performance in the particular order discussed or illustrated, unless specifically identified as an order of performance. It is also to be understood that additional or alternative steps may be employed.

The use of the expression “at least” or “at least one” suggests the use of one or more elements, as the use may be in one of the embodiments to achieve one or more of the desired objects or results.

CLAIMS:

1. A system (100) for querying heterogeneous data sources (102), comprising:
   i. a plurality of data sources (102), each data source having a discrete volume, discrete hardware resources and tools for facilitating processing of queries;
   ii. a central orchestrator unit (106) for monitoring the data sources (102) and the hardware resources and to select at least one machine learning module for processing of queries;
   iii. a predictor module (108) to receive at least one query for retrieving information from at least one of said data sources (102); and
   iv. a selector module (110) having a multiple connector interface (118) for connecting with each of said data sources (102),

   wherein said predictor module (108) predicts processing time for executing a query using each of the data sources (102) based on said at least one selected machine learning module, said selector module (110) compares the predicted processing time for each data source with a class of service of the query to route the query to a data source (102) via the interface (118) for execution of the query within the predicted processing time for that data source (102).

2. A system (100) as claimed in claim 1, wherein said central orchestrator unit (106) includes a scheduler for (i) maintaining and classifying the data sources; (ii) monitoring utilization of the hardware resources; (iii) initiating a self-learning process based on new training data; and (iv) testing accuracy of the machine learning module.

3. A system (100) as claimed in claim 1, wherein said machine learning module is selected from the group consisting of regression analysis, decision trees and neural networks.

4. A system (100) as claimed in claims 1, wherein said selector module (110) routes the query to a data source (102) having a lowest processing time based on the

comparison of predicted processing time for each data source with a class of service of the query.

5. A system (100) as claimed in claim 1, the system (100) includes a query editor (104) to receive at least one query for retrieving information from at least one of said data sources (102).

6. A system (100) as claimed in claim 5, wherein said query editor (104) co-operates with a memory module (116) to preemptively auto-complete and auto-correct the query.

7. A system (100) as claimed in claim 5, wherein said query editor (104) co-operates with the central orchestrator unit (106) to initiate routing of the query to at least one data source in the event said query editor (104) receives instructions to execute the query.

8. A system (100) as claimed in claim 6, wherein said memory module (116) is communicably linked to the data sources (102) for holding metadata of the data sources (102) and any previously executed queries.

9. A method for querying heterogeneous data sources, wherein each data source (102) having a discrete volume, discrete hardware resources and tools for facilitating processing of queries, the method comprising the following steps:

i. receiving at least one query for retrieving information from the data sources (102);

ii. selecting at least one machine learning module for routing a query to an appropriate data source (102);

iii. predicting processing time for executing the query for each of the data sources (102) based on the machine learning module;

iv. comparing the processing time for each of the data sources (102) with a class of service of the query;

v. selecting a data source (102) having a lowest processing time based on result of comparison obtained from step (iv); and

vi. executing the query at the selected data source (102) within the predicted processing time for the selected data source (102).

10. A method as claimed in claim 9, wherein the step of receiving at least one query for retrieving information from said data sources (102) includes:
    i. receiving the query via a query editor (104) interface;
    ii. checking the query while it is being typed in the query editor (104) (a) to determine if typed column names exist within a table name chosen for an data source (102), and (b) to suggest column names or strings while the query is being typed;
    iii. checking the query language syntax and detecting and highlighting appropriate portions of the query in the event an error is detected in (a) or (b) of step (ii); and
    iv. invoking a central orchestrator unit (106) on receiving instructions for executing the query.

11. A method as claimed in claim 9, wherein the step of executing the query at the selected data source (102) within the predicted processing time further includes adding execution details of the query along with the predicted processing time and class of service in a self-learning module (114) as a training data if the query failed to execute within the predicted processing time.

12. A method as claimed in claim 9, wherein the method further includes the steps of:
    i. scheduling checks on all tables of the data sources (102);
    ii. classifying the tables based on their volume into big data, general data and small data categories;
    iii. balancing the volume of the data sources (102); and
    iv. monitor real-time load on the hardware resources associated with the data sources (102).

13. A method as claimed in claim 10, wherein prior to the step of invoking a central orchestrator unit (106) the method includes determining a data source which

includes the table name appearing in the query; checking the volume of that data source; routing the query to that data source and skipping steps (iii) to (v) of Claim 9 if the volume of the data source is classified as small.

14. A method as claimed in claims 11, wherein prior to the step of selecting at least one machine learning module for routing a query, the method further includes the steps of:

i. using the training data in the self-learning module (114) to evaluate accuracy of processing failed queries using a new machine learning module;

ii. re-evaluating accuracy of processing the failed queries using another machine learning module; and

iii. comparing accuracy of machine learning model selected in step (i) and step (ii) to select a machine learning module for routing a query.

15. A computer program product for querying heterogeneous data sources (102), wherein each data source (102) having a discrete volume, discrete hardware resources and tools for facilitating processing of queries, said computer program product having instructions operable to cause one or more modules to perform the following steps:

i. receiving at least one query for retrieving information from the data sources (102);

ii. selecting at least one machine learning module for routing a query to an appropriate data source (102);

iii. predicting processing time for executing the query for each of the data sources (102) based on the machine learning module;

iv. comparing the processing time for each of the data sources (102) with a class of service of the query;

v. selecting a data source (102) having a lowest processing time based on result of comparison obtained from step (iv); and

vi. executing the query at the selected data source (102) within the predicted processing time for the selected data source (102).

16. A computer program product for querying heterogeneous data sources (102), wherein each data source (102) having a discrete volume, discrete hardware resources and tools for facilitating processing of queries, said computer program product being tangibly implemented on a machine readable media and comprising:

i. a query editor (104) to receive at least one query for retrieving information from the data sources (102);

ii. a central orchestrator unit (106) for monitoring the data sources (102) and the hardware resources and to select at least one machine learning module for processing of queries;

iii. a predictor module (108) to receive at least one query from the query editor (104);

iv. a selector module (110) having a multiple connector interface for connecting with each of said data sources (102); and

v. an output module (112) cooperating with the data sources (102),

wherein said predictor module (108) predicts processing time for executing a query using each of the data sources (102) based on said at least one selected machine learning module, said selector module (110) compares the predicted processing time for each data source with a class of service of the query to route the query to a data source (102) via the interface (118) for execution of said query within the predicted processing time for that data source (102) and display of the query results on the output module (112).

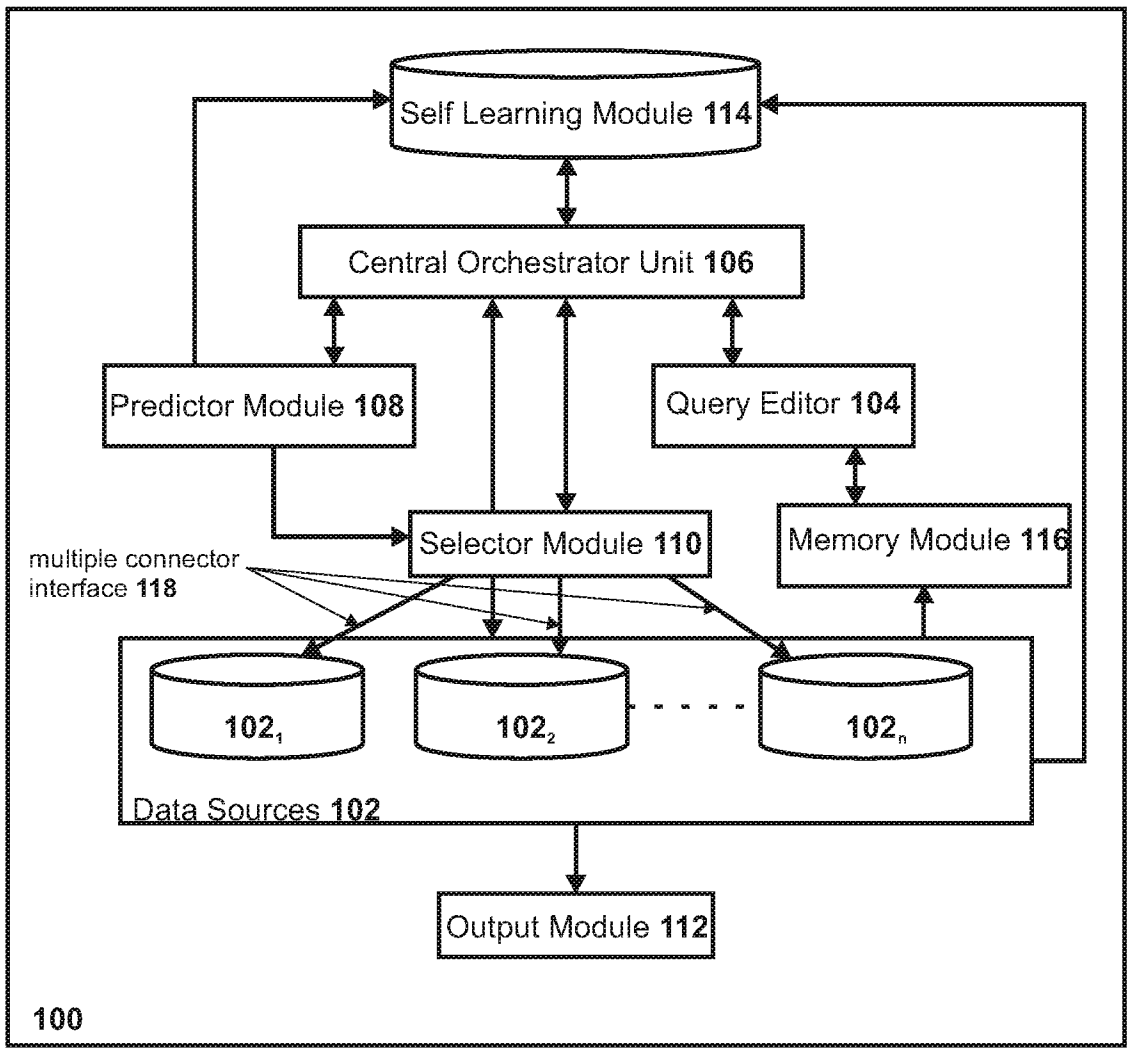

Self Learning Module 114
Central Orchestrator Unit 106
Predictor Module 108
Query Editor 104
Selector Module 110
Memory Module 116
multiple connector interface 118
102_1
102_2
102_n
Data Sources 102
Output Module 112
100

FIGURE 1

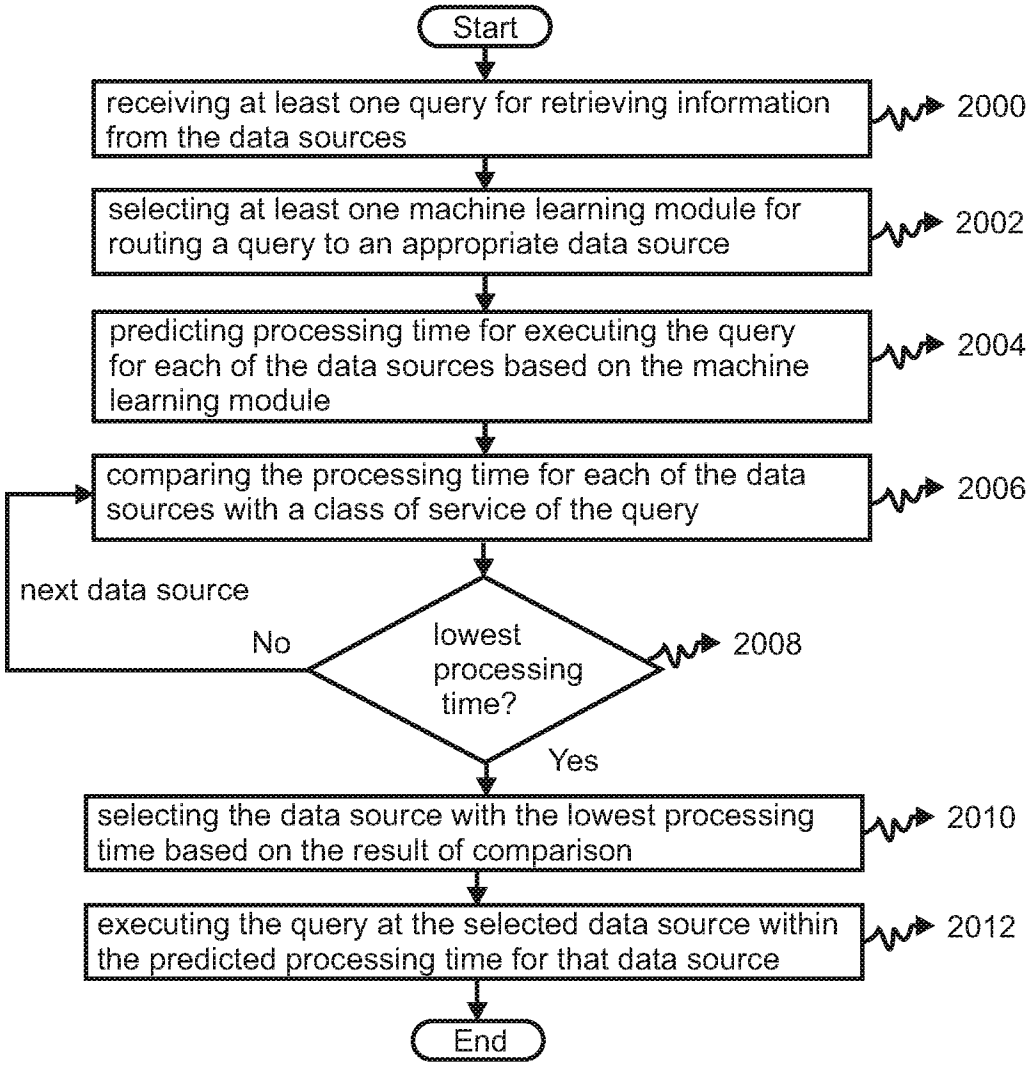

Start
receiving at least one query for retrieving information from the data sources
2000
selecting at least one machine learning module for routing a query to an appropriate data source
2002
predicting processing time for executing the query for each of the data sources based on the machine learning module
2004
comparing the processing time for each of the data sources with a class of service of the query
2006
next data source
No
lowest processing time?
2008
Yes
selecting the data source with the lowest processing time based on the result of comparison
2010
executing the query at the selected data source within the predicted processing time for that data source
2012
End

FIGURE 2

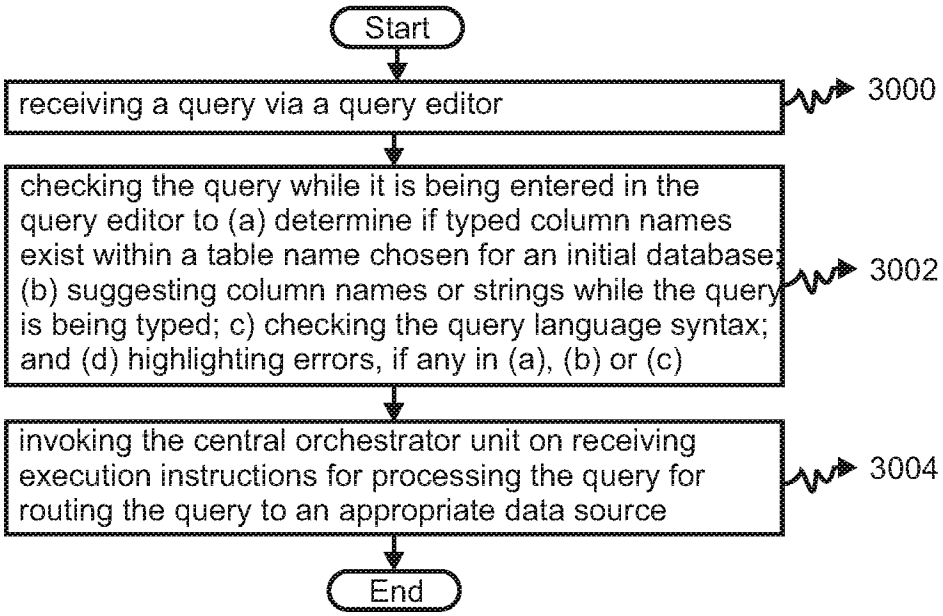


FIGURE 3

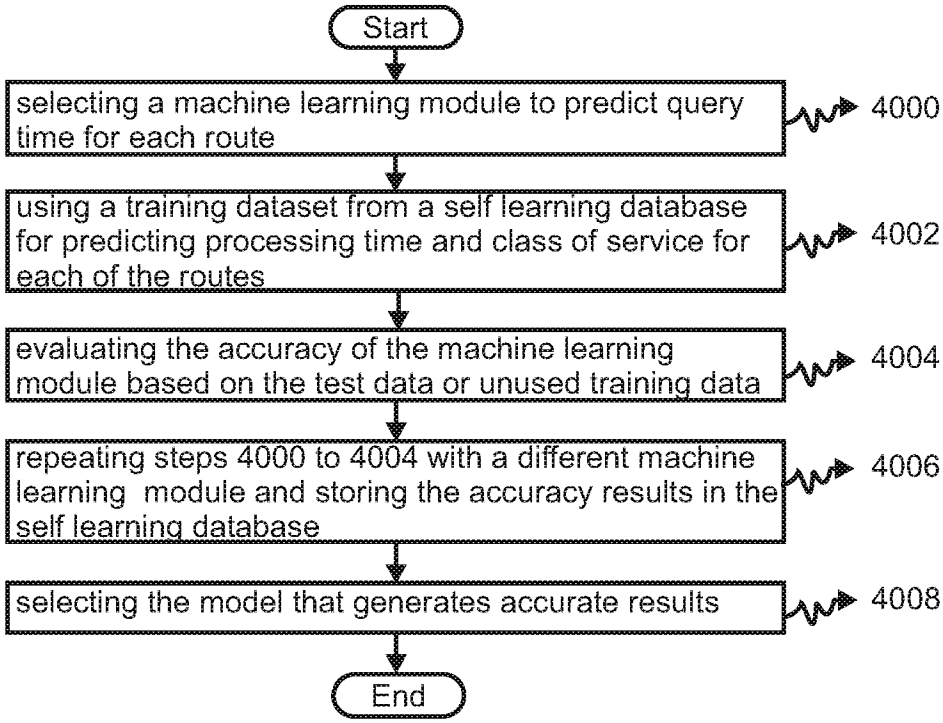


FIGURE 4

INTERNATIONAL SEARCH REPORT

International application No
PCT/IB2015/055343

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06F17/30
ADD.

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

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
EPO-Internal, COMPENDEX, INSPEC, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	WO 2012/112980 A2 (MICROSOFT CORP [US]) 23 August 2012 (2012-08-23) paragraph [0018] - paragraph [0022]; figure 1 paragraph [0026] - paragraph [0027]; figure 2 paragraph [0033] - paragraph [0035]; figure 3 paragraph [0043] paragraph [0045] - paragraph [0047]; figures 5-7 ----- -/--	1-16

[X] Further documents are listed in the continuation of Box C.

[X] 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	Date of mailing of the international search report
10 November 2015	17/11/2015

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Polzer, Andreas

INTERNATIONAL SEARCH REPORT

International application No
PCT/IB2015/055343

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	WO 2009/097438 A2 (HEWLETT PACKARD DEVELOPMENT CO [US]; PAPAEMMANOUIL OLGA [US]; BASU SUJ) 6 August 2009 (2009-08-06) paragraph [0023] - paragraph [0027]; figure 1 paragraph [0030] paragraph [0035] paragraph [0041] - paragraph [0056]; figures 4,5	1,9,15, 16
A	US 2006/195416 A1 (EWEN STEPHAN E [DE] ET AL) 31 August 2006 (2006-08-31) paragraph [0020]; figure 1 paragraph [0024] - paragraph [0027]; figure 2 paragraph [0036] - paragraph [0043]; figures 4,5	1,9,15, 16

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/IB2015/055343

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 2012112980 A2	23-08-2012	CN 102708121 A	03-10-2012
		EP 2676192 A2	25-12-2013
		US 2012215763 A1	23-08-2012
		WO 2012112980 A2	23-08-2012
WO 2009097438 A2	06-08-2009	CN 101933018 A	29-12-2010
		JP 2011514577 A	06-05-2011
		KR 20100113098 A	20-10-2010
		US 2009192981 A1	30-07-2009
		WO 2009097438 A2	06-08-2009
US 2006195416 A1	31-08-2006	NONE	