(12) UK Patent Application (19) GB (11) 2522918 (13) A

(43) Date of A Publication 12.08.2015

(21) Application No: 1402292.5

(22) Date of Filing: 11.02.2014

(71) Applicant(s):
International Business Machines Corporation
New Orchard Road, Armonk 10504, New York, United States of America

(72) Inventor(s):
Pedro Barbas

(74) Agent and/or Address for Service:
IBM United Kingdom Limited
Intellectual Property Law, Hursley Park, WINCHESTER, Hampshire, SO21 2JN, United Kingdom

(51) INT CL:
G06F 21/62 (2013.01)

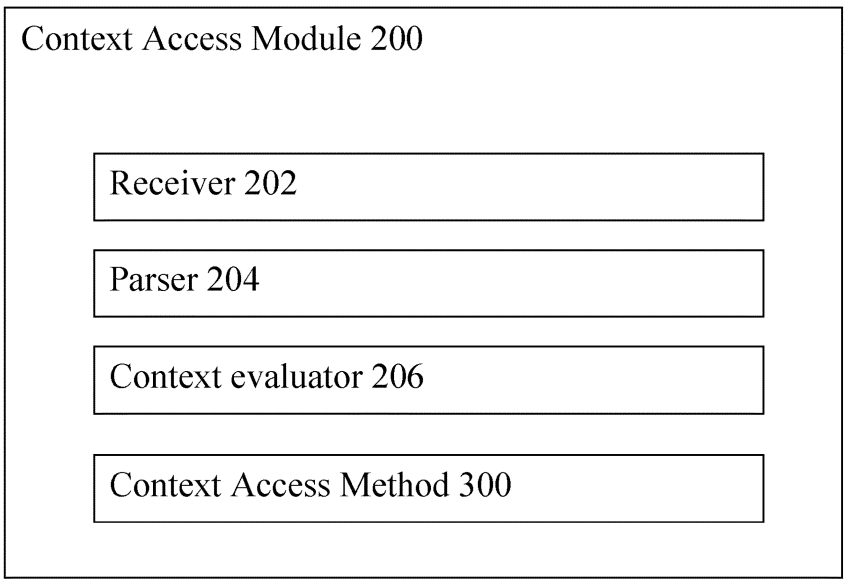
(56) Documents Cited:
WO 2012/112593 A1 **WO 2007/068546 A1**
US 6487552 B1 **US 20120109926 A1**
Bhatti et al, Engineering a Policy-based System for Federated Healthcare Databases; IEEE Transactions on Knowledge and Data Engineering, vol. 19, no.9, September 2007
HU et al, Control for Distributed Healthcare Applications, Department of Computer Science, University of Virginia

(58) Field of Search:
INT CL **G06F**
Other: **WPI, EPODOC, Internet**

(54) Title of the Invention: **Adaptive access control in relational database system**
Abstract Title: **Context-aware access control in relational database system**

(57) A request is received (202) for access to one or more tables of a relational database. A context-aware access control is defined for at least one of the tables, and it is verified whether the access request satisfies the context-aware access control (206). If so, access is permitted (300) to the one or more tables based on the identified and verified context-aware access control. Access is preferably permitted to one or more portions of the tables, such as a specific row or column or a range of tables.

FIGURE 2



GB 2522918 A

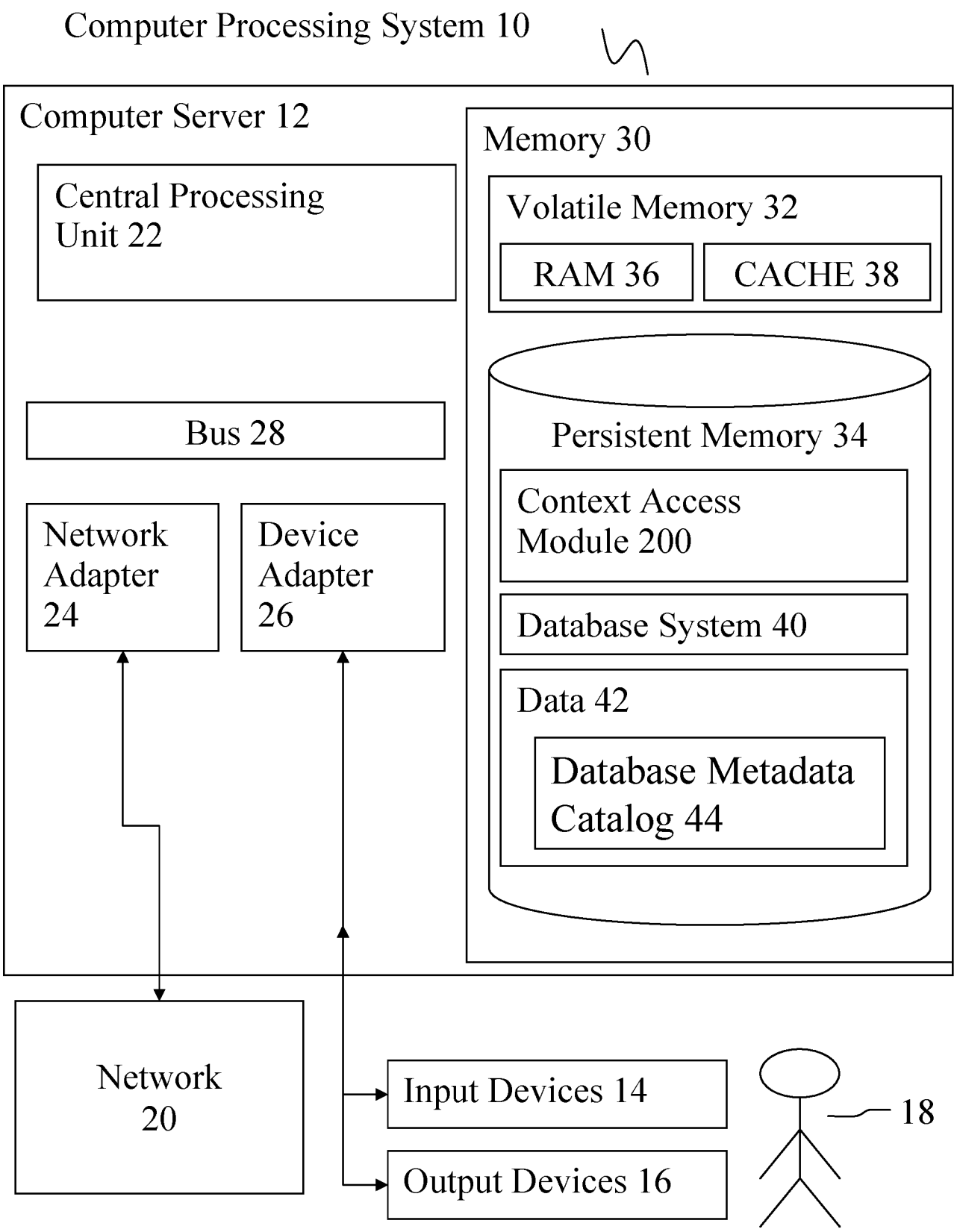

Computer Processing System 10
Computer Server 12
Memory 30
Central Processing Unit 22
Volatile Memory 32
RAM 36
CACHE 38
Persistent Memory 34
Bus 28
Context Access Module 200
Network Adapter 24
Device Adapter 26
Database System 40
Data 42
Database Metadata Catalog 44
Network 20
Input Devices 14
Output Devices 16
18

FIGURE 1

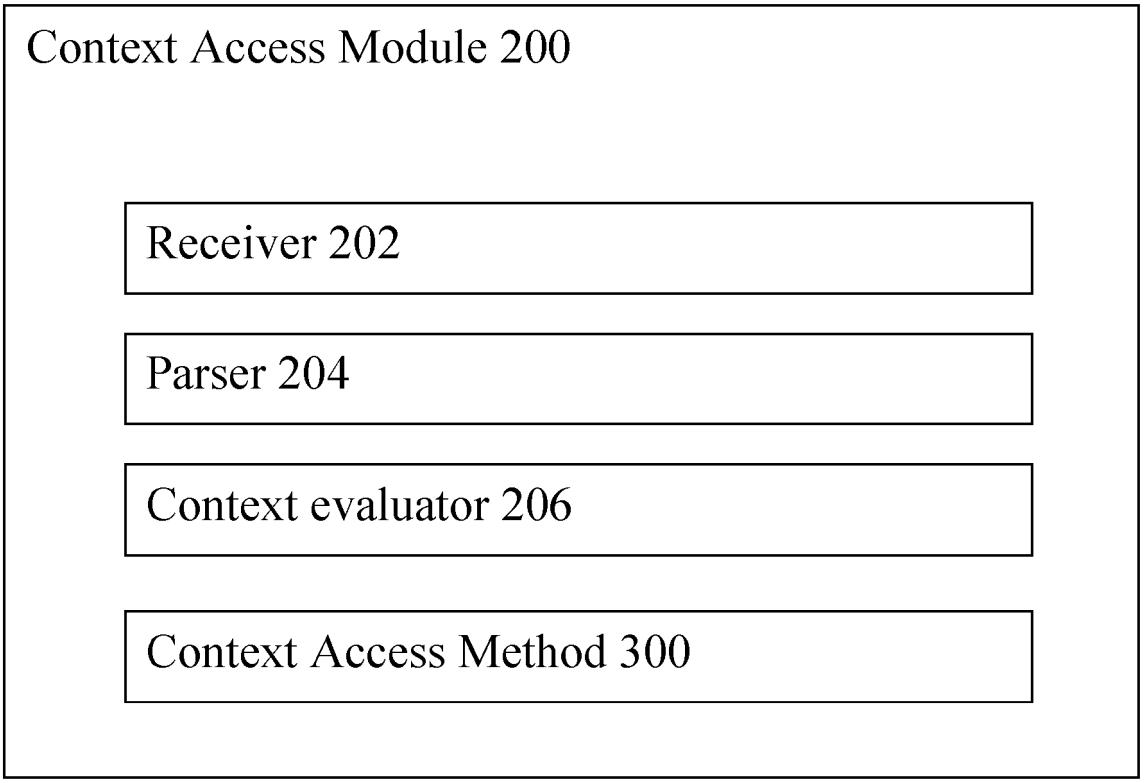

Context Access Module 200
Receiver 202
Parser 204
Context evaluator 206
Context Access Method 300

FIGURE 2

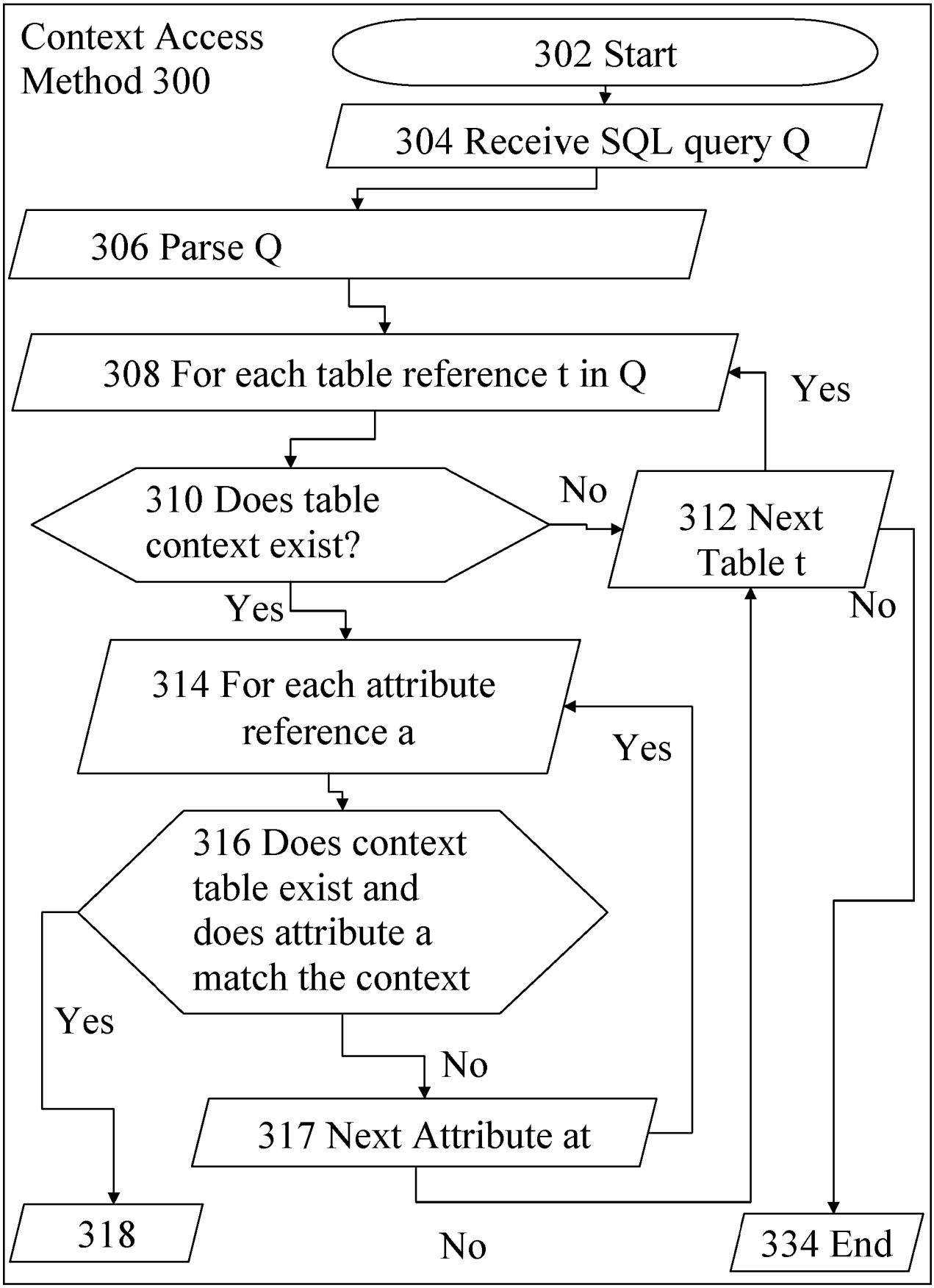

Context Access Method 300
302 Start
304 Receive SQL query Q
306 Parse Q
308 For each table reference t in Q
Yes
310 Does table context exist?
No
312 Next Table t
No
Yes
314 For each attribute reference a
Yes
316 Does context table exist and does attribute a match the context
Yes
No
317 Next Attribute at
318
No
334 End

FIGURE 3A

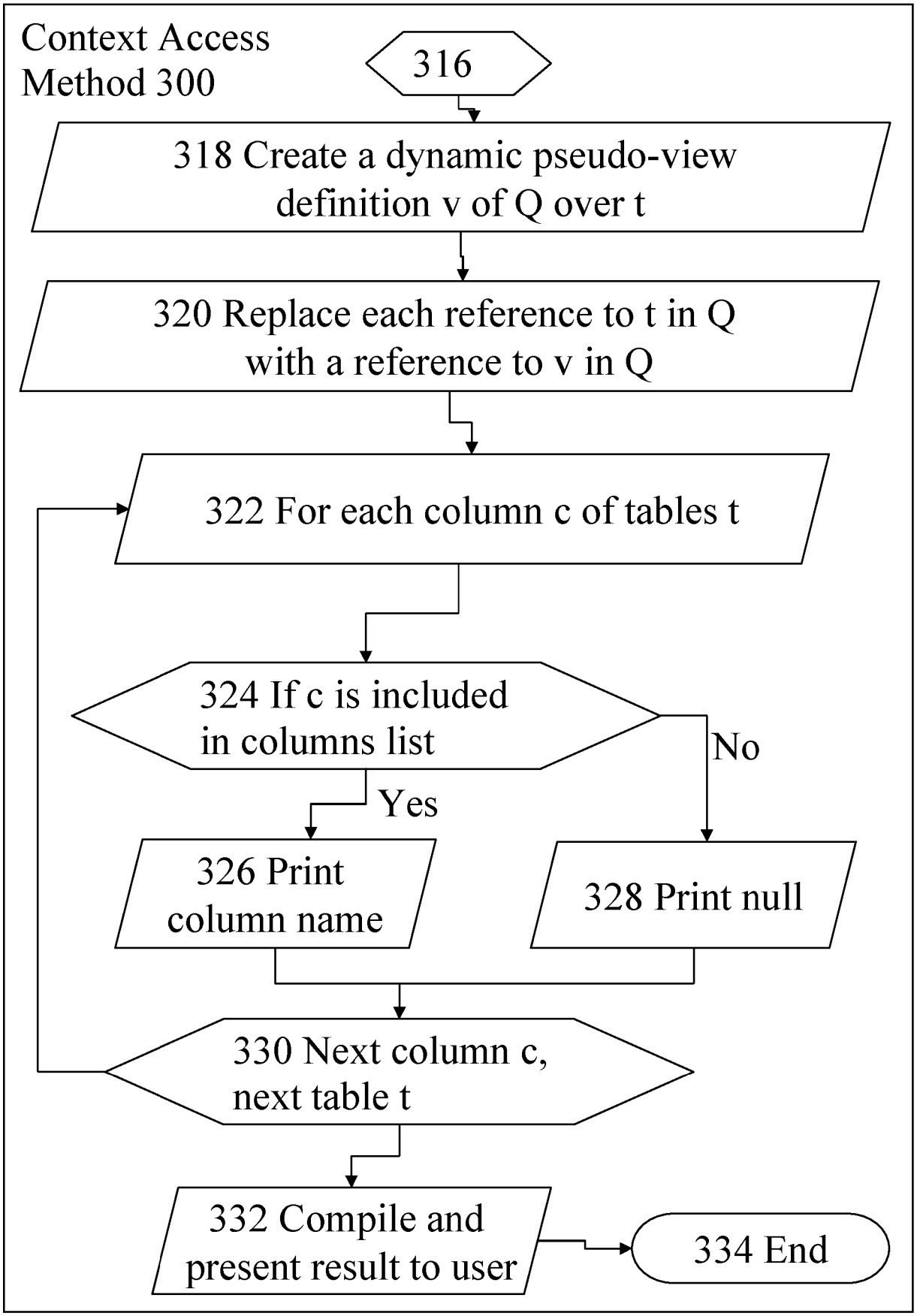

Context Access Method 300
316
318 Create a dynamic pseudo-view definition v of Q over t
320 Replace each reference to t in Q with a reference to v in Q
322 For each column c of tables t
324 If c is included in columns list
No
Yes
326 Print column name
328 Print null
330 Next column c, next table t
332 Compile and present result to user
334 End

FIGURE 3B

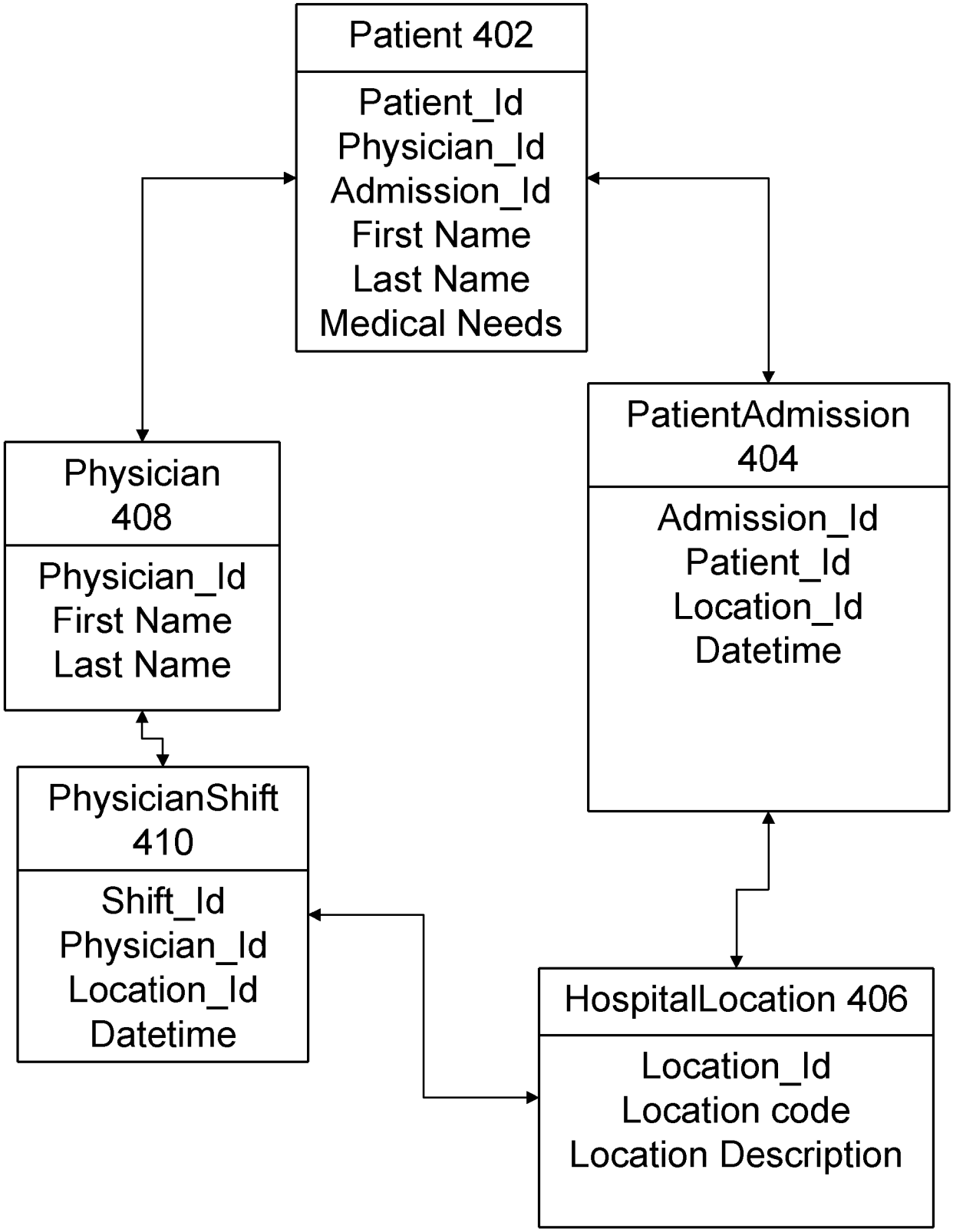

Patient 402
Patient_Id
Physician_Id
Admission_Id
First Name
Last Name
Medical Needs
PatientAdmission 404
Admission_Id
Patient_Id
Location_Id
Datetime
Physician 408
Physician_Id
First Name
Last Name
PhysicianShift 410
Shift_Id
Physician_Id
Location_Id
Datetime
HospitalLocation 406
Location_Id
Location code
Location Description

FIGURE 4

CREATE CONTEXT	*context-name*
ON TABLE	*table-name*
FOR COLUMNS	*columns-name*
ATTRIBUTES	*(table-name:attribute, table-name:attribute, ...)*
FOR PURPOSE	*purpose-name*
FOR RECIPIENT	*recipient-authorization-name*
ENABLE	

FIGURE 5A

CREATE CONTEXT	Patient-Emergency
ON TABLE	Patients
FOR COLUMNS	ALL
ATTRIBUTES	(PhysicianShift:Location_Id, PhysicianShift:Datetime)
FOR PURPOSE	Patients
FOR RECIPIENT	Physicians
ENABLE	

FIGURE 5B

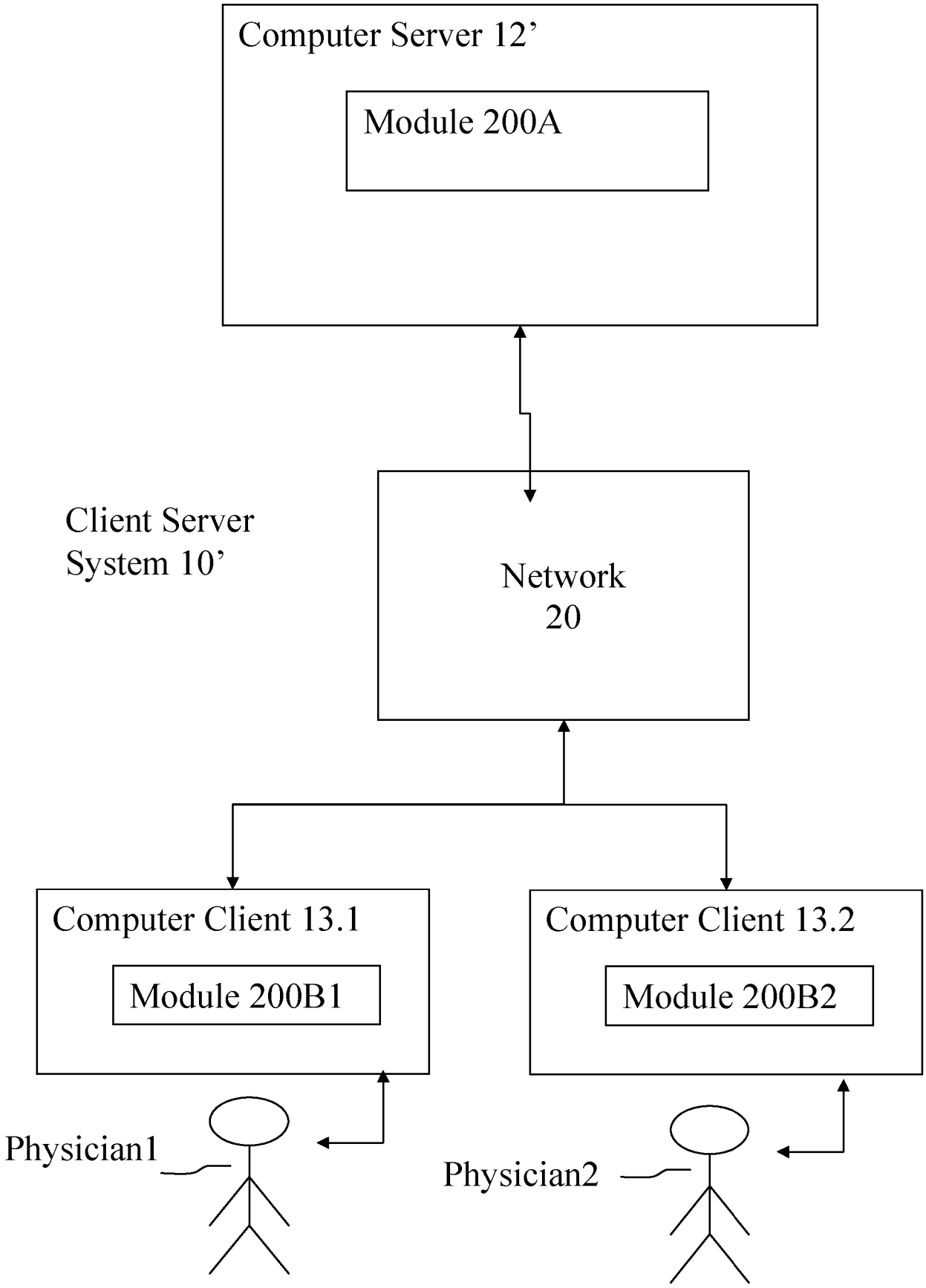

Computer Server 12'
Module 200A
Client Server System 10'
Network 20
Computer Client 13.1
Module 200B1
Computer Client 13.2
Module 200B2
Physician1
Physician2

FIGURE 6

ADAPTIVE ACCESS CONTROL IN RELATIONAL DATABASE SYSTEM

FIELD OF THE INVENTION

[0001] The invention relates to the field of database management systems. In particular, the invention relates to adaptive access control in a relational database system (RDBMS).

BACKGROUND

[0002] Due to both legal and business reasons, business enterprises are increasingly becoming sensitive to data security. Many RDBMS implement discretionary access controls (DAC) for granting table object access privileges to a user. These privileges may be managed by role-based access controls (RBAC), where a user wishing to access data in a table object must be a member of a role permitted to access the data in the table object. Another technique for controlling access to data in a table on a column-level or a row-level includes use of label-based access controls (LBAC), such that, unless a label of a user is compatible with a label associated with a row or column of a table, then the data for that row or column is not returned to the user. Due to the restrictive nature of label components, business enterprises have turned to more flexible mechanisms, for example, fine-grained access controls (FGAC) including views, triggers, virtual private databases, and row and column access controls.

[0003] These prior art solutions (built on static DAC, RBAC, LBAC and FGAC models) do not address the intricate security requirements of business enterprises, requiring dynamic authorization enforcement and flexible context-aware access control. In the prior art, security solutions are created, administered, and enforced through static access control rules present within the database system. But these database systems are unable to take into account the dynamic state changes in which data is being accessed.

[0004] In a medical example, a particular primary care physician is only permitted to view patient information for his own patients. However when this physician is working in an intensive care unit, he should also be able to see patient information from all patients

admitted in the intensive care unit in that period. Since it is impossible to know beforehand what patients will be admitted in the intensive care unit in that period, the prior art fails to adjust the security solution accordingly.

[0005] To highlight the need, if a particular patient is being transported to a hospital in an ambulance, then the paramedic who is taking care of the patient in the ambulance should also be allowed to view the patient information.

[0006] IBM Lotus Notes allows fine-grained event-based access control based on events. IBM, Lotus and Notes. IBM, Lotus and Lotus Notes are trademarks of International Business Machines Corporation in the US and/or elsewhere.

[0007] US patent publication 20110296430 discloses event-based data protection based on events that trigger state models.

BRIEF SUMMARY OF THE INVENTION

[0008] In a first aspect of the invention there is provided a system for implementing context-aware access control of data in a relational database system comprising: a receiver for receiving a request to access one or more tables of the database; and a context evaluator for identifying a context-aware access control defined for at least one of the tables, verifying whether the identified context-aware access control is satisfied; and permitting access to the one or more tables based on the identified and verified context-aware access control.

[0009] Advantageously a database security administrator can define one or more context aware access controls, and in every query submitted to the database, those same context aware access controls are validated if they reference any table from the, for example, SQL query.

[0010] In a second aspect of the invention there is provided a method for implementing context-aware access control of data in a relational database system comprising: receiving a request to access one or more tables of the database; identifying a context-aware access

control defined for at least one of the tables; verifying whether the identified context-aware access control is satisfied; and permitting access to the one or more tables based on the identified and verified context-aware access control.

[0011] The embodiments have an effect on queries made outside the database system computer by allowing them to access restricted data not allowed outside the context. The embodiments have an effect that operates at a database system level of a computer and below any overlying application level. The embodiments have an effect that results in the database system being made to operate in a new way to allow context access.

[0012] Preferably the request does not have access to said one or more tables by virtue the security level of the request and/or the originator of the request.

[0013] More preferably access is permitted to one or more portions of the tables based on the identified and verified context-aware access control.

[0014] Most preferably further comprising creating a view for the one of more portions of the table where access is permitted.

[0015] Even more preferably comprising replacing each reference to a restricted portion of a table in the request with a reference to the created view of the assessable portions of the table.

[0016] Yet more preferably the database catalog contains said one or more context aware access control.

[0017] Advantageously further certifying whether an identified applicable context-aware access control is to be applied to the user request.

[0018] More advantageously the request for access comprises a request for a specific row or column or range of tables.

[0019] Most advantageously a list of unrestricted columns is printed.

[0020] Yet more advantageously the conditions of the context-aware access control are satisfied by data in the database.

[0021] In a third aspect of the invention there is provided a computer program product for implementing context-aware access control of data in a relational database system, the computer program product comprising a computer-readable storage medium having computer-readable program code embodied therewith and the computer-readable program code configured to perform all the steps of the methods.

[0022] The computer program product comprises a series of computer-readable instructions either fixed on a tangible medium, such as a computer readable medium, for example, optical disk, magnetic disk, solid-state drive or transmittable to a computer system, using a modem or other interface device, over either a tangible medium, including but not limited to optical or analogue communications lines, or intangibly using wireless techniques, including but not limited to microwave, infrared or other transmission techniques. The series of computer readable instructions embodies all or part of the functionality previously described.

[0023] Those skilled in the art will appreciate that such computer readable instructions can be written in a number of programming languages for use with many computer architectures or operating systems. Further, such instructions may be stored using any memory technology, present or future, including but not limited to, semiconductor, magnetic, or optical, or transmitted using any communications technology, present or future, including but not limited to optical, infrared, or microwave. It is contemplated that such a computer program product may be distributed as a removable medium with accompanying printed or electronic documentation, for example, shrink-wrapped software, pre-loaded with a computer system, for example, on a system ROM or fixed disk, or distributed from a server or electronic bulletin board over a network, for example, the Internet or World Wide Web.

[0024] In a fourth aspect of the invention there is provided a computer program stored on a computer readable medium and loadable into the internal memory of a computer, comprising software code portions, when said program is run on a computer, for performing all the steps of the method claims.

[0025] In a fifth aspect of the invention there is provided a data carrier aspect of the preferred embodiment that comprises functional computer data structures to, when loaded into a computer system and operated upon thereby, enable said computer system to perform all the steps of the method claims. A suitable data-carrier could be a solid-state memory, magnetic drive or optical disk. Channels for the transmission of data may likewise comprise storage media of all descriptions as well as signal-carrying media, such as wired or wireless signal-carrying media.

BRIEF DESCRIPTION OF THE DRAWINGS

[0026] Preferred embodiments of the present invention will now be described, by way of example only, with reference to the following drawings in which:
Figure 1 is a deployment diagram of the preferred embodiment;
Figure 2 is a component diagram of the preferred embodiment;
Figure 3A and 3B together depict a flow diagram of a process of the preferred embodiment;
Figure 4 is a data table diagram of an example utilized by the preferred embodiment;
Figure 5A and 5B are database access request actions, labels and corresponding code examples according to an example of the preferred embodiment; and
Figure 6 is a deployment diagram of a client server embodiment.

DETAILED DESCRIPTION OF THE EMBODIMENTS

[0027] Referring to Figure 1, the deployment of a preferred embodiment in computer processing system 10 is described. Computer processing system 10 is operational with numerous other general purpose or special purpose computing system environments or configurations. Examples of well-known computing processing systems, environments, and/or configurations that may be suitable for use with computer processing system 10

include, but are not limited to, personal computer systems, server computer systems, thin clients, thick clients, hand-held or laptop devices, multiprocessor systems, microprocessor-based systems, set top boxes, programmable consumer electronics, network PCs, minicomputer systems, mainframe computer systems, and distributed cloud computing environments that include any of the above systems or devices.

[0028] Computer processing system 10 may be described in the general context of computer system-executable instructions, such as program modules, being executed by a computer processor. Generally, program modules may include routines, programs, objects, components, logic, and data structures that perform particular tasks or implement particular abstract data types. Computer processing system 10 may be embodied in distributed cloud computing environments where tasks are performed by remote processing devices that are linked through a communications network. In a distributed cloud computing environment, program modules may be located in both local and remote computer system storage media including memory storage devices.

[0029] Computer processing system 10 comprises: general-purpose computer server 12 and one or more input devices 14 and output devices 16 directly attached to the computer server 12. Computer processing system 10 is connected to a network 20. Computer processing system 10 communicates with a user 18 using input devices 14 and output devices 16. Input devices 14 include one or more of: a keyboard, a scanner, a mouse, trackball or another pointing device. Output devices 16 include one or more of a display or a printer. Computer processing system 10 communicates with network devices (not shown) over network 20. Network 20 can be a local area network (LAN), a wide area network (WAN), or the Internet.

[0030] Computer server 12 comprises: central processing unit (CPU) 22; network adapter 24; device adapter 26; bus 28 and memory 30.

[0031] CPU 22 loads machine instructions from memory 30 and performs machine operations in response to the instructions. Such machine operations include: incrementing or decrementing a value in a register; transferring a value from memory 30 to a register or vice

versa; branching to a different location in memory if a condition is true or false (also known as a conditional branch instruction); and adding or subtracting the values in two different registers and loading the result in another register. A typical CPU can perform many different machine operations. A set of machine instructions is called a machine code program, the machine instructions are written in a machine code language which is referred to a low level language. A computer program written in a high level language needs to be compiled to a machine code program before it can be run. Alternatively a machine code program such as a virtual machine or an interpreter can interpret a high level language in terms of machine operations.

[0032] Network adapter 24 is connected to bus 28 and network 20 for enabling communication between the computer server 12 and network devices.

[0033] Device adapter 26 is connected to bus 28 and input devices 14 and output devices 16 for enabling communication between computer server 12 and input devices 14 and output devices 16.

[0034] Bus 28 couples the main system components together including memory 30 to CPU 22. Bus 28 represents one or more of any of several types of bus structures, including a memory bus or memory controller, a peripheral bus, an accelerated graphics port, and a processor or local bus using any of a variety of bus architectures. By way of example, and not limitation, such architectures include Industry Standard Architecture (ISA) bus, Micro Channel Architecture (MCA) bus, Enhanced ISA (EISA) bus, Video Electronics Standards Association (VESA) local bus, and Peripheral Component Interconnects (PCI) bus.

[0035] Memory 30 includes computer system readable media in the form of volatile memory 32 and non-volatile or persistent memory 34. Examples of volatile memory 32 are random access memory (RAM) 36 and cache memory 38. Generally volatile memory is used because it is faster and generally non-volatile memory is used because it will hold the data for longer. Computer processing system 10 may further include other removable and/or non-removable, volatile and/or non-volatile computer system storage media. By way of example only, persistent memory 34 can be provided for reading from and writing to a non-

removable, non-volatile magnetic media (not shown and typically a magnetic hard disk or solid-state drive). Although not shown, further storage media may be provided including: an external port for removable, non-volatile solid-state memory; and an optical disk drive for reading from or writing to a removable, non-volatile optical disk such as a compact disk (CD), digital video disk (DVD) or Blu-ray. In such instances, each can be connected to bus 28 by one or more data media interfaces. As will be further depicted and described below, memory 30 may include at least one program product having a set (for example, at least one) of program modules that are configured to carry out the functions of embodiments of the invention.

[0036] The set of program modules configured to carry out the functions of the preferred embodiment comprises content access module 200; database system 40; data 42; and a database metadata catalog 44. Further program modules that support the preferred embodiment but are not shown include firmware, boot strap program, operating system, and support applications. Each of the operating system, support applications, other program modules, and program data or some combination thereof, may include an implementation of a networking environment.

[0037] Computer processing system 10 communicates with at least one network 20 (such as a local area network (LAN), a general wide area network (WAN), and/or a public network like the Internet) via network adapter 24. Network adapter 24 communicates with the other components of computer server 12 via bus 28. It should be understood that although not shown, other hardware and/or software components could be used in conjunction with computer processing system 10. Examples, include, but are not limited to: microcode, device drivers, redundant processing units, external disk drive arrays, redundant array of independent disks (RAID), tape drives, and data archival storage systems.

[0038] Referring to Figure 2, the preferred embodiment implements context-aware access control to data in a table of a relational database system. The context may be in relation to a user or a role performed by the user. Context access module 200 comprises: receiver 202; parser 204; context evaluator 206; and context access method 300.

[0039] Receiver 202 is for receiving a user SQL query after a user requests access to a table using an SQL query.

[0040] Parser 204 is for splitting the query into several components preceding compilation and for transforming the components in a compiler definition.

[0041] Context evaluator 206 is for validating if any context-aware access control has been configured for any table references. If a context-aware access control is configured, then the context evaluator evaluates if the context references any tables from the current SQL query. This decision is based on a validation of defined table based attributes with the purpose of the context. The context-aware access request is then applied to the SQL query if the validation matches the attributes with the purpose of the context. After all validations for the table have been processed, the context evaluator 206 then creates a pseudo-view object using the table columns configured in the context. After all tables have been evaluated for context-aware access control permissions and the resultant pseudo-view is created, then context evaluator 206 enforces the result instead of the original internal SQL query representation. The result is then presented to the user.

[0042] Context access method 300 is for performing the method of the preferred embodiment thereby controlling the context access module 200; context access method 300 is described in more detail below.

[0043] Referring to Figures 3A and 3B, context access method 300 comprises logical process steps 302 to 334.

[0044] Step 302 is the start of context access method 300.

[0045] Step 304 is for receiving an SQL query called Q. When a user of a database management system tries to access a table using an SQL query, a SQL compiler receives the SQL query and relevant environment information, for example a default session user name.

[0046] Step 306 is for parsing query Q. Parser 204 splits the SQL query into several components and converts them into an internal representation.

[0047] Step 308 is a loop defining each table reference t in the SQL query Q.

[0048] Step 310 is for checking if a context is defined that references any table t (or part of a table) from the current SQL query. Context evaluator 206 accesses database catalogue metadata and validates the check. If no context-aware permission definition exists, then the current security policies on the database remain unmodified and the next table reference in the query is evaluated by looping back to step 308 via 312. If a context-aware permission is defined then step 314.

[0049] Step 312 is for looping back to step 308 if no context-aware permission is defined or for exiting (end step 334) if there are no more table references t.

[0050] Step 314 is for defining a loop for each attribute reference a in table t attributes. Next step 316.

[0051] Step 316 is for determining if attribute a matches the context and progressing to step 318 if so. Else step 317. The determination is performed by context evaluator 206. The determination is evaluated upon a comparison between the defined table based attributes with the purpose of the context with respect to the recipient list. If no attributes match the context then the process exits and step 317.

[0052] Step 317 is for looping back to step 314 for the next attribute. If there are no more attributes then step 312 for next table.

[0053] Step 318 is for creating a view definition v of Q over t such that a view object is created. The view is a select statement that references each column defined in the context-aware permission command. Next, context evaluator 206 converts the view definition into internal compiler definition and enforces the result instead of the original internal SQL query

representation. This way any policy permissions that are in effect for the specific user are overridden by the context-aware permission access (for example LBAC permissions).

[0054] Step 320 is for replacing each reference of table t in query Q with a reference to table v.

[0055] Step 322 is for defining a loop for each column c of t.

[0056] Step 324 is for branching to step 326 if c is included in the columns list. Else step 328.

[0057] Step 326 is for printing the column name and continuing at step 330.

[0058] Step 328 is for not printing a value or printing a null character and continuing at step 330.

[0059] Step 330 is for looping back to step 322 for the next column c is there is one or else continuing at step 332.

[0060] Step 332 is for compiling the result and presenting it to a user.

[0061] Step 334 is the end of context access method 300.

[0062] Referring to Figure 4, an example health system database model comprises: a patient table 402; a patient admission table 404; a hospital location table 406; a physician table 408 and a physician shift table 410.

[0063] Patient table 402 comprises the following fields: Patients_Id; Physician_Id; Admission_Id; First Name; Last Name; and Medical Needs. Patients_Id is an identifier to identify the patent. Physician_Id is an identifier to identify the physician who normally looks after the patient and who would have normal access to the patient records. Admission_Id is an identifier that indicates if a patient has been admitted to a hospital. One or more

admission identifiers can be on a list when a patient has been admitted to hospital on a several occasions. First Name and Last Name are the first and last names of the patient. Medical Needs is a field describing the medical needs of the patient. In the prior art only the physician identified in the patient table would have access to the Medical Needs field in this table (or any other entry for this patient).

[0064] PatientAdmission 404 is a table holding a more detailed record for each admission in patient table 402. Access is restricted to the identified physician in the corresponding patient table record. The following fields are included: Admission_Id; Patients_Id; Location_Id; and Datetime.

[0065] HospitalLocation 406 is a table detailing the location of each hospital for each location in PatientAdmission 404 and also for each location in PhysicianShift 410. The following fields are included: Location_Id; Location code; and Location Description.

[0066] Physician 408 is a table holding records for each physician listed in patients 402. The following fields are included: Physician_Id; First Name; and Last Name.

[0067] PhysicianShift 408 is a table holding records for each shift that a physician has made comprising the following fields: Shift_Id; Physician_Id; Location_Id; and Datetime.

[0068] Referring to Figure 5A and Figure 5B, a corresponding example context-aware control data structure with respective labels and respective values is described. Context-aware permissions are defined in the database for an object such as a table and are typically stored in a database metadata catalog 44. Context-aware control data structure comprises: a CREATE CONTEXT field; an ON TABLE field; a FOR COLUMNS field; an ATTRIBUTES field; a FOR PURPOSE field; a FOR RECIPIENT field; and an ENABLE field

[0069] The CREATE CONTEXT field (for example *context-name*=Patient-Emergency) identifies a context control that operates on columns (for example *columns-name*=ALL) in a table (for example *table-name*=Patients) identified the ON TABLE field and the FOR

COLUMNS field for a particular recipient (for example *recipient-authorization-name*=Physicians). The actual context is expressed using the for purpose field (for example *purpose-name*=Patients) and the ATTRIBUTES field (for example *table-name: attribute*= PhysicianShift:Location_Id; *table-name:attribute*=PhysicianShift:Datetime). The enable field enables the context-aware control.

[0070] Those in the FOR RECEPIENT field are allowed access to table *table-name* on columns *columns-name*. In this context-aware access control example the context definition is specified using one or more *table-name:attribute* fields and *purpose-name field.*

[0071] *Context-name* is unique and cannot be the same as a previous context-aware access control that already exists. In the example of Figure 5B, the *context-name* is 'Patient-Emergency' in the database example covering patients and emergencies.

[0072] *Table-name* is the name of the table where the context-aware access control has its domain. In the example the table-name is 'Patients'.

[0073] *Columns-name* is the name of the column that can be accessed in the context domain. In the example, the value of *columns-name* is 'ALL' signifying that all column names should be accessed in the context domain.

[0074] *Table-name:attribute* is the name of a specific table base attribute or attributes, that need to be satisfied with the purpose of the context-aware access control. In the example, two attributes are listed: PhysicianShift:Location_Id and PhysicianShift:Datetime.

[0075] *Purpose-name* is the name of the purpose. In the example 'Patients' is the purpose.

[0076] *Recipient-authorization-name* is the name of the recipients that have access to table *table-name* on columns *columns-name*. In the example 'Physicians' are listed as the groups that will have access.

[0077] Once context-aware permissions are configured on a table, any SQL query that attempts to access that table will have the context imposed on that access.

[0078] In this example, a patient emergency context is defined. The current database model has implemented a context-aware access control to define that only the primary care physician of a patient will be able to see the patient's information. The goal of the patient emergency context is to allow to physician that is working in the intensive care unit, to be able to see patient information from all patients admitted in the intensive care unit during the work period only. That is in the context of the patient being in an intensive care unit at the same time as a doctor is working in that intensive care unit. In Figure 5B, the defined context-aware access control states that those in a physician's role (a database role) are allowed access the table Patients on all columns, if the attributes Location and Datetime are validated in the context Patients.

[0079] The context evaluation function accesses the catalogue metadata and the current database model in the following way:
- For a specific patient John, the database model indicates that John is in intensive care treatment in an emergency department: Location=Emergency from "Select Location_Id from patients"
- For a specific physician, Jack, the database model indicates that the Location=Emergency from "Select Location_Id from PhysicianShift"
- Both John and Jack are in the same datetime interval, that is Datetime from PhysicianShift matches Datetime from PatientAdmission.
- For a specific patient Mark, the database model shows that he is not in intensive care treatment in the emergency department: Location<>Emergency from "Select Location_Id from patients"
- Both John and Mark have as a primary care physician Joey.

[0080] If Jack issues those two queries:
1) Select 'Medical Needs' from Patients where 'First Name' = "John"
2) Select 'Medical Needs' from Patients where 'First Name' = "Mark"

[0081] On the first query, the context evaluation function validates Location and Datetime in the context Patients, Jack will be able to see John's information although Jack is not the primary care physician of John. Likewise Jack will not be able to see Mark's information as he is not is primary care physician. Although the context-aware permission applies to the table Patients, the context evaluation function does not validate Location.

[0082] A recipient Physician record is present on both tables 'Physician' and 'PhysicianShift' for the patient emergency context. Both attributes Location_Id and Datetime from table PhysicianShift need to be evaluated on the purpose of Patients. A patient's record is present on tables 'Patients' and 'PatientAdmission' and both attributes Location_Id and Datetime from table PatientAdmission need to be validated on the recipient Physician. When attributes Location_Id and Datetime from table PhysicianShift match the attributes Location_Id and Datetime from table PatientAdmission then the patient emergency context is matched. This allows any physician working in the specific location where a patient is under treatment, and where the time frame for the physician is also included in the time frame where a patient is under treatment, then the physician is able to see the patient information. This example is provided as merely for illustration purposes and many variations can be added with for example the inclusion of different attributes to be validated for the context.

[0083] Further embodiments of the invention are now described. It will be clear to one of ordinary skill in the art that all or part of the logical process steps of the preferred embodiment may be alternatively embodied in a logic apparatus, or a plurality of logic apparatus, comprising logic elements arranged to perform the logical process steps of the method and that such logic elements may comprise hardware components, firmware components or a combination thereof.

[0084] It will be equally clear to one of skill in the art that all or part of the logic components of the preferred embodiment may be alternatively embodied in logic apparatus comprising logic elements to perform the steps of the method, and that such logic elements may comprise components such as logic gates in, for example a programmable logic array or application-specific integrated circuit. Such a logic arrangement may further be embodied in

enabling elements for temporarily or permanently establishing logic structures in such an array or circuit using, for example, a virtual hardware descriptor language, which may be stored and transmitted using fixed or transmittable carrier media.

[0085] In a further alternative embodiment, the present invention may be realized in the form of a computer implemented method of deploying a service comprising steps of deploying computer program code operable to, when deployed into a computer infrastructure and executed thereon, cause the computer system to perform all the steps of the method.

[0086] It will be appreciated that the method and components of the preferred embodiment may alternatively be embodied fully or partially in a parallel computing system comprising two or more processors for executing parallel software.

[0087] Referring to Figure 6, a further alternative embodiment of the present invention may be realized in the form of a client server system 10' comprising computer server 12' and computer client 13'. Computer server 12' connects to computer clients 13.1 and 13.2 via network 20. Computer clients 13.1 and 13.2 provides computing services to physicians labeled physician1 and physician2. In this client server embodiment module 200A is located and processed on computer server 12' whereas client modules 200B1 and 200B2 are located and processed in the computer clients 13.1 and 13.2 respectively. In this client server embodiment, the method is provided as a service to the clients. In this example, two clients are shown as used by two different physicians intended to represent a physician that has access to the patient's records at all time and another physician that has access only during an emergency. In other example any number of clients can be utilized and for different types of user including nurses who also need context-aware access during emergencies and administrators who not need context access.

[0088] It will be clear to one skilled in the art that many improvements and modifications can be made to the foregoing exemplary embodiment without departing from the scope of the present invention.

CLAIMS

1. A system for implementing context-aware access control of data in a relational database system comprising:

a receiver for receiving a request to access one or more tables of the database; and

a context evaluator for identifying a context-aware access control defined for at least one of the tables, verifying whether the identified context-aware access control is satisfied; and

permitting access to the one or more tables based on the identified and verified context-aware access control.

2. A system according to claim 1 whereby the request does not have access to said one or more tables by virtue of a request security level and/or a request originator.

3. A system according to claim 1 whereby access is permitted to one or more portions of the tables based on the identified and verified context-aware access control.

4. A system according to claim 3 further comprising creating a view for the one of more portions of the table where access is permitted.

5. A system according to claim 4 further comprising replacing each reference to a restricted portion of a table in the request with a reference to the created view of the assessable portions of the table.

6. A system according to any of claims 1 to 5 wherein the database catalog contains said one or more context aware access controls.

7. A system according to any of claims 1 to 6 further certifying whether any identified applicable context-aware access controls are to be applied to the user request.

8. A system according to any of claims 1 to 7 whereby the request for access comprises a request for a specific row or column or range of tables.

9. A system according to any of claims 1 to 8 wherein a list of unrestricted columns is printed.

10. A system according to any of claims 1 to 9 wherein the conditions of the context-aware access control are satisfied by data in the database.

11. A method for implementing context-aware access control of data in a relational database system comprising:
receiving a request to access one or more tables of the database;
identifying a context-aware access control defined for at least one of the tables;
verifying whether the identified context-aware access control is satisfied; and
permitting access to the one or more tables based on the identified and verified context-aware access control.

12. A method according to claim 11 whereby the request does not have access to said one or more tables by virtue of a request security level and/or a request originator.

13. A method according to claim 11 whereby access is permitted to one or more portions of the tables based on the identified and verified context-aware access control.

14. A method according to claim 13 further comprising creating a view for the one of more portions of the table where access is permitted.

15. A method according to claim 14 further comprising replacing each reference to a restricted portion of a table in the request with a reference to the created view of the assessable portions of the table.

16. A method according to any of claims 11 to 15 wherein the database catalog contains said one or more context aware access controls.

17. A method according to any of claims 11 to 16 further certifying whether any identified applicable context-aware access controls are to be applied to the user request.

18. A method according to any of claims 11 to 17 whereby the request for access comprises a request for a specific row or column or range of tables.

19. A method according to any of claims 11 to 18 wherein a list of unrestricted columns is printed.

20. A method according to any of claims 11 to 19 wherein the conditions of the context-aware access control are satisfied by data in the database.

21. A computer program product for implementing context-aware access control of data in a relational database system, the computer program product comprising a computer-readable storage medium having computer-readable program code embodied therewith, the computer-readable program code configured to perform any of the method claims.

22. A computer program stored on a computer readable medium and loadable into the internal memory of a digital computer, comprising software code portions, when said program is run on a computer, for performing any of the method claims.

Application No: GB1402292.5 **Examiner:** Mr Jim Calvert

Claims searched: 1-22 **Date of search:** 9 July 2014

Patents Act 1977: Search Report under Section 17

Documents considered to be relevant:

Category	Relevant to claims	Identity of document and passage or figure of particular relevance
X	1-22	US6487552 B1 (LEI CHON ET AL) See e.g. abstract.
X	1-22	US2012/109926 A1 (NOVIK ET AL) See paras 0051-55
X	1-22	WO2012/112593 A1 (PROTEGRITY) See e.g. p.8, l.27- end of page 9
X	1-22	WO2007/068546 A1 (IBM) See paras 0028-32
X	1-22	Bhatti et al, Engineering a Policy-based System for Federated Healthcare Databases; IEEE Transactions on Knowledge and Data Engineering, vol. 19, no.9, September 2007 See whole document
X	1-22	HU et al, Control for Distributed Healthcare Applications, Department of Computer Science, University of Virginia See whole document

Categories:

X	Document indicating lack of novelty or inventive step	A	Document indicating technological background and/or state of the art.
Y	Document indicating lack of inventive step if combined with one or more other documents of same category.	P	Document published on or after the declared priority date but before the filing date of this invention.
&	Member of the same patent family	E	Patent document published on or after, but with priority date earlier than, the filing date of this application.

Field of Search:

Search of GB, EP, WO & US patent documents classified in the following areas of the UKC^{X} :

Worldwide search of patent documents classified in the following areas of the IPC

G06F

The following online and other databases have been used in the preparation of this search report

WPI, EPODOC, Internet

International Classification:

Subclass	Subgroup	Valid From
G06F	0021/62	01/01/2013