

DATA MODEL DEVELOPING PLATFORM

FIELD

[0001] The present disclosure relates to a general technical tool for designing a data model in an information system, which is applicable to the design of databases and data warehouses in the IT industry, and in particular relates to a data model developing platform.

BACKGROUND

[0002] Traditionally, the design of a data model can be accomplished by one person in a short period of time due to its small scale and small workload, requiring no teamwork or knowledge sharing.

[0003] Nowadays, databases or data warehouses are getting larger and larger, and thus can hardly be designed by few persons in a short period of time. In addition, data governance requires standardization of data models, which further requires team collaboration, version management, and standard introduction for the design of data models. In addition, due to the impact of the epidemic since the beginning of 2020, lots of collaboration between teams cannot be done at the same location. Therefore, a tool for distributed design of a data model is required.

[0004] It is found that there are at least the following problems in the conventional technology.

[0005] At present, tools for designing a data model commonly used in the industry include PowerDesigner and ERWin. These tools are stand-alone systems and cannot support multi-person collaboration.

SUMMARY

[0006] In view of this, the purpose of the present disclosure is to provide a data model developing platform to realize distributed development of data models that supports multi-person collaboration.

[0007] In order to achieve the above purpose, a data model developing platform is provided,

having a B/S three-tier architecture, and including:

a page client in foreground, configured to access an application server through Nginx;

the application server in background, deployed with application services associated with data model development, and

5 a data storage device, configured to store data using a relational database, and cache data using a key-value database,

where the application server in background includes:

a data standard managing module, configured to maintain a data standard and word-roots;

10 a data model managing module, configured to design a data model based on the data standard and/or the word-roots; and

a project managing module, configured to compare any two versions of a project, to generate a difference file, and export the difference file into an SQL file; and

a tenant managing module, configured to provide a multi-tenant environment that
15 isolates projects of different tenants from each other, so that each tenant is allowed to view only an owned project.

[0008] In some possible embodiments, the data standard managing module includes:

a data standard maintaining sub-module, configured to maintain the data standard by performing one or more of maintenance operations of adding, deleting, modifying, querying,
20 and synchronizing the data standard, wherein the synchronizing the data standard comprises: synchronously acquiring the data standard from an external database or an external application; and

a word-root maintaining sub-module, configured to maintain the word-roots configured to maintain the word-roots by performing one or more of maintenance operations of retrieving,
25 adding, modifying, deleting and synchronizing the word-roots, wherein the synchronizing the word-roots comprises: synchronously acquiring the word-roots from an external database or an external application.

[0009] In some possible embodiments, the data model managing module includes:

a data model duplicating sub-module, configured to: duplicate a data model to form a first replica model, in response to an operation instruction for modifying the data model, where only the first replica model is allowed to be modified by a user; and/or

5 search for an existing data model having a target subject according to a matching condition and duplicate the existing data model to form a second replica model, in response to an operation instruction for forming a data model having the target subject issued by the user, receive a modification operation on the second replica model by the user, and generate an updated data model having the target subject.

[0010] In some possible embodiments, the data model managing module further includes:

10 a data model statistics sub-module, configured to track a progress of a current project;

a data model maintaining sub-module, configured to perform one or more of operations of adding a data model, deleting a data model, modifying a data model, and submitting a data model;

15 a data model import and export sub-module, configured to import and export a model document in a format of Excel; or import a model document in a format of Excel and export a model document in a format of data definition language (DDL); and,

a data model releasing sub-module, configured to deliver a final version of a data model to a predetermined environment.

[0011] In some possible embodiments, the project managing module may further include:

20 a project creating sub-module, configured to create a project to which a data model to be developed belongs, and decompose the project according to a decomposition rule, where the decomposition rule is based on subjects and/or service systems;

a project manager designating sub-module, configured to designate a project manager in response to a selecting instruction or a configuring instruction;

25 a project member managing sub-module, configured to manage developers in a development team of the project by performing one or more of: adding a developer, deactivating a developer, enabling a developer, and deleting a developer.

[0012] In some possible embodiments, the project managing module may further include:

a project modifying sub-module, configured to modify basic information of the project;

a service system maintaining sub-module, configured to maintain a service system to which the project belongs;

a model layer maintaining sub-module, configured to maintain a layer to which data models in the project belong;

5 a data model naming rule sub-module, configured to maintain a naming rule of the data models in the project, so that the data models have a unified style of naming;

a data source managing sub-module, configured to select and determine a database corresponding to the physicalization of the data models and a database corresponding to reverse engineering; and

10 a data model verifying sub-module, configured to verify a designed data model using the data standard and submit the verified data model to a node corresponding to the project manager.

[0013] In some possible embodiments, the project managing module further includes any one or more of:

15 a project handover sub-module, configured to: in response to replacing a project manager, hand over a project corresponding to the replaced project manager to a node corresponding to a designated person;

a model handover sub-module, configured to: in response to replacing a project developer, hand over a data model designed by the replaced project developer to a node
20 corresponding to another project developer of the project; and

a reverse engineering sub-module, configured to convert a table in a database into a data model.

[0014] In some possible embodiments, the tenant managing module includes any one or more of the following sub-modules:

25 a tenant creating sub-module, configured to create a tenant;

a tenant modifying sub-module, configured to modify basic information of the tenant;

a tenant enabling sub-module, configured to enable a disabled tenant;

a tenant disabling sub-module, configured to disable a current tenant;

a tenant deleting sub-module, configured to delete the disabled tenant, and delete all projects, models, private data standards and word-roots belonging to the disabled tenant; and

a tenant handover sub-module, configured to: in response to replacing a tenant administrator, hand over a tenant to a node corresponding to another person.

5 **[0015]** In some possible embodiments, the application server in background further includes:

a data type managing module, configured to maintain mapping relations between data types of different databases, wherein the mapping relations are used to convert the data model between different databases, where the data type managing module includes:

10 a data type querying sub-module, configured to retrieve a configured mapping relation between data types of different databases; and

a data type maintaining sub-module, configured to add, modify, and delete a mapping relation between data types of different databases.

[0016] In some possible embodiments, the relational database includes a MySQL database,
15 an external database is connected with the MySQL database through an RESTful interface, and the MySQL database acquires the data standard and the word-roots from the external database;

the application server in background further includes: a log managing sub-module configured to generate a log record for a sensitive operation, where the sensitive operation
20 includes at least one of: user logging in, deleting a user, deleting a tenant, and deleting a data model.

[0017] The above technical solutions have the following advantageous effects:

[0018] 1. distributed deployment can be achieved, which eliminates the limitation of geographic space, so that team members can work in collaboration in different places, and
25 share work results online therebetween;

[0019] 2. multi-version management can be achieved, which can trace the history of version change by comparing any versions;

[0020] 3. data standards can be preset, so that designers can design a data model based on existing standards; and

[0021] 4. the multi-tenant function is provided, so that multiple different tasks can be performed on the same platform. The multi-tenant function facilitates resource isolation during team collaboration. The purpose of this function is to divide a large project into multiple small projects within a tenant, which can share private data standards under demonstration of the same tenant administrator.

BRIEF DESCRIPTION OF THE DRAWINGS

[0022] In order to more clearly illustrate embodiments of the present disclosure or technical solutions in the conventional technology, the drawing describing the embodiments or the conventional technology will be briefly described hereinafter. Apparently, the drawings in the following description are only some examples of the present disclosure, and for those skilled in the art, other drawings may be obtained based on the provided drawing without any creative efforts.

[0023] Figure 1A is a schematic diagram illustrating a technical architecture of a data model developing platform according to an embodiment of the present disclosure;

[0024] Figure 1B is a schematic diagram illustrating a technical architecture of another data model developing platform according to an embodiment of the present disclosure;

[0025] Figure 2 is a schematic diagram illustrating an overall architecture according to an embodiment of the present disclosure;

[0026] Figure 3 is a flowchart illustrating a development process according to an embodiment of the present disclosure; and

[0027] Figure 4 is a schematic diagram illustrating a functional architecture of a data model developing platform according to an embodiment of the present disclosure.

DETAILED DESCRIPTION

[0028] Technical solutions in the embodiments of the present disclosure are clearly and completely described hereinafter in conjunction with the drawings in the embodiments of the present disclosure. Apparently, the embodiments described in the following are only some embodiments of the present disclosure, rather than all embodiments. Based on the

embodiments in the present disclosure, all of other embodiments, made by the person skilled in the art without any creative efforts, fall into the scope of protection of the present disclosure.

5 **[0029]** A visualized distributed data model developing platform, which supports multi-person collaboration, and supports version management, is provided according to the embodiments of the present disclosure. One of the purposes of the embodiments of the present disclosure is to meet the requirements of multi-person collaboration and version management in the design of large-scale databases/data warehouses. The design of large-scale databases/data warehouses is complex and has a large workload, which requires multiple
10 people to collaborate. However, the team members may not work at the same place. Moreover, a relatively large number of versions exist in the iteration of a project and are complicated to manage. The embodiments of the present disclosure focus on solving at least one of technical problems of lack of support of multi-person collaboration, lack of version management function, lack of ability to apply an existing standard field library and word-root library, and
15 lack of distributed capability in conventional tools for designing a data model.

20 **[0030]** A method for distributed development of a data model is provided according to the embodiments of the present disclosure, which has the ability to apply data standards in the stage of data model design. In the embodiments of the present disclosure, multiple versions can co-exist in the development of the data model. A tool for the distributed development of the data model according to the embodiments of the present disclosure has functions of version management and/or data standard introduction.

Embodiment 1

25 **[0031]** Figure 1A is a schematic diagram illustrating a technical architecture of a data model developing platform according to an embodiment of the present disclosure. Figure 1B is a schematic diagram illustrating a technical architecture of another data model developing platform according to an embodiment of the present disclosure. As shown in Figures 1A and 1B, the data model developing platform adopts a B/S (Browser/Server) three-tier structure, which is different from the currently used model development tool adopting the C/S structure.

30 **[0032]** The design model having the B/S three-tier structure includes: foreground (Web side), background (application system, referred to as application, service, etc.), data (or storage, also known as database).

[0033] The front end, i.e., the web side, is configured to access an application server through Nginx. Nginx is a high-performance HTTP and reverse proxy web server and also provides IMAP/POP3/SMTP services. The following technologies are applied to the front-end page: Vue+JavaScript+ajax+json+HTML5+G6.

5 **[0034]** Specifically, for page presentation and layout at the front end, pages that meet the requirements of HTML5 are generated dynamically by using Vue and JavaScript. All the interactions with the back end are implemented by using ajax and json. The graphical presentation is implemented by using G6 and JavaScript technology. The relevant terms are explained as follows.

10 **[0035]** Vue is a progressive JavaScript framework used to build a user interface. Vue.js is aimed at performing data binding and composed view components with a simple API.

[0036] JavaScript (abbreviated as "JS") is a lightweight, interpreted or just-in-time compiled programming language with First-class functions. JavaScript is a high-level scripting language applied to the network, having been widely used in the development of
15 web applications. It is generally used to add various dynamic functions to web pages, to provide users with a smoother browsing effect. JavaScript scripts are generally embedded in HTML.

[0037] AJAX refers to Asynchronous JavaScript and XML. XML refers to Extensible Markup Language, which is a markup language used to mark up electronic documents to
20 make them structured.

[0038] JSON (JavaScript Object Notation, JS Object Notation) is a lightweight data exchange format. JSON is based on a subset of ECMAScript (a js specification formulated by the European Computer Association), which stores and represents data by using a text format that is completely independent of the programming language. Concise and clear hierarchy
25 makes JSON an ideal data exchange language. It is easy for humans to read and write and also easy for machines to parse and generate, and effectively improve the efficiency of network transmission.

[0039] HTML5 is an abbreviation of Hyper Text Markup Language 5. HTML5 is a language description method for constructing Web content. HTML5 technology is developed
30 based on relevant standards of HTML4.01 and is officially released in 2008, which meets the

requirements of modern network development.

[0040] G6 is a simple, easy-to-use, and complete graph visualization engine based on HTML5. It provides a series of elegantly designed and easy-to-use graph visualization solutions on the basis of high customization capabilities.

5 **[0041]** An application system (corresponding to services and applications in Figure 2) can be deployed in a cluster or at a single machine, depending on the number of accessing users. The application system refers to the application and service part of the data model developing platform. An application system refers to software, and an application server includes hardware, which refers to a server deployed with application services. The application server
10 may be deployed as a cluster or a single machine according to actual conditions. Cluster deployment refers to the networking of multiple machines to work in collaboration, which is used in a case of large access amount and heavy load. Single-machine deployment refers to that the application server is deployed at one machine, which is used in a case of light load. The following technologies are applied in the application services: SpringBoot, load
15 balancing nginx, application development language Java+ORM framework MyBatis. Object Relational Mapping (abbreviated as ORM, or O/RM, or O/R mapping) is a programming technology used to realize the conversion between data of systems of different types in object-oriented programming languages. MyBatis is a persistence layer framework that supports customized SQL, stored procedures and advanced mapping. The application server
20 in background involves: a page client in foreground, configured to access an application server through Nginx; and an application server in background, used to deploy application services associated with data model development. The application server in background includes: a data standard managing module, configured to maintain a data standard and word-roots; and a data model managing module, configured to design a data model based on
25 the data standard and/or the word-roots, where the word-roots are used to help a developer to generate corresponding English field names as a recommended name in a case that there is no standard. In a case that there is a standard, the word-roots are not used and a name in the standard can be directly selected for use. The application server in background further includes a project managing module, configured to compare any two versions of a project to
30 generate a difference file, and export the difference file into an SQL file; a tenant managing module, configured to provide a multi-tenant environment, which isolate projects of different tenants from each other, so that each tenant is allowed to view only an owned project; and a

data storage device, configured to store data using a relational database, and caches data using a key-value database.

[0042] Currently, a user database generally includes three environments, i.e., a development environment, a test environment, and a production environment. These three environments are required to be synchronized. However, in the conventional technology, comparison can only be performed for the current version. Even PowerDesigner, ERWin or other tools for data model development, can only compare the versions before and after modification, which cannot meet the requirement of comparison under the multi-environment, multi-user, and multi-version situations. In the process of project development, multiple versions may be generated. In this embodiment, versions corresponding to any two time points can be compared, to generate a difference file. For example, during the development, a new version is generated every day. If the project manager wants to know what change is made today compared to yesterday, the difference can be obtained by comparing the versions.

[0043] In the present embodiment, the SQL file is exported for continuous integration/continuous deployment (CI/CD). Files in this format can be directly executed in the production environment, while files in other formats, such as Excel, cannot meet the requirements of CI/CD. The cross-version comparison is to compare the various environments, to generate a difference script, which can be directly executed to synchronize these differences, thereby synchronizing the database structures of the development environment, the test environment, and the production environment.

[0044] In the embodiment, the isolation may be implemented through user authority. The isolation of resources between tenants is implemented by a user management function. Also, resource sharing between the tenants can be implemented through management of authority.

[0045] A relational database is used for data storage, including but not limited to: MySQL database, Oracle, DB2, Microsoft SQL Server, Microsoft Access, etc., and a key-value database is used for caching, including but not limited to Redis (Remote Dictionary Server) In-memory database or Memcached (a distributed high-speed cache system) for caching high-frequency data. The data storage part occupies relatively small resources and can be deployed at a single machine. If there are a relatively large number of concurrent users (for example, more than 50 people), it can also be deployed in a cluster.

[0046] A physical platform may include: a cluster or a single machine for application

servers, a Redis cache, a MySQL database, and a Nginx.

[0047] The above technologies in the embodiment of the present disclosure adopt a mainstream general technology system, so that product iteration, operation and maintenance are guaranteed.

5 **[0048]** Figure 2 is a schematic diagram illustrating an overall architecture of a data model developing platform according to an embodiment of the present disclosure. As shown in Figure 2, the data model developing platform is accessible through a browser, and is also connected with a standard word-root database. By accessing the data model developing platform through a browser, users are not required to install additional software. External
10 standards and word-roots are periodically or manually acquired through a RESTful interface and saved into a configuration repository. In the present embodiment, the standards and word-roots in a system-level are inputted through the RESTful interface, and private standards and word-roots are imported from an external Excel file. In an alternative solution, the RESTful interface is replaced with an RPC interface. In the present embodiment, the
15 advantages of using the RESTful interface lie in that the RESTful interface is the current mainstream standardized remote invoking interface, which is in a popular API design style using the HTTP protocol, and thus is concise and highly maintainable.

[0049] The application server in background of the data model developing platform includes the following multiple functional modules.

20 **[0050]** A project managing module has functions of modifying basic information (code, name, start and end date, etc.) and main attributes (whether a primary key is required, whether a foreign key is required, whether to enforce data standard verification) of a project, module maintenance, data source management, reverse engineering or the like.

[0051] A module maintenance module has functions of adding, deleting and modifying a
25 module name, codes and a displaying order.

[0052] A tenant managing module is configured for the management of a workspace by a system administrator. The term “tenant” and “workspace” have similar meaning in the present disclosure.

[0053] A developer managing module is configured for the management of developers in a
30 development team, having functions of adding a developer, deactivating a developer, enabling

a developer, deleting a developer, resetting a password or the like. The functions are only available for the system administrator, a workspace administrator, and a project administrator.

[0054] A data source managing module is configured for the user to manage database connection information of the current project, including IP address, port, database, schema, account and password. One project may have multiple data sources, one of which may be selected when creating a model.

[0055] A requirement managing module is configured to maintain the development requirements of the project, which facilitates a project manager to track development progress. Requirements management can be designed independently of project management.

[0056] A data model maintenance module has functions of creating data models, adding columns, modifying columns, and copying data models.

[0057] A data model examining module is configured for a project administrator to examine a model. When a developer submits a designed data model to the project administrator, if the project administrator thinks that the data model has a reasonable structure and can be adopted, the project administrator approves the model, otherwise the project administrator rejects the model, so that the developer modifies the model until the project administrator approves it. The purpose is to prevent unreasonable models from being inputted into a next process, to avoid trouble to the data development.

[0058] A standard managing module has functions of retrieving, adding, deleting, modifying and synchronizing standards (from an external database or application). After the standards are established, a standard name can be directly selected from the standards when maintaining the model, which greatly reduces the workload of the developer and helps to establish a standardized model.

[0059] A word-root management (maintenance) module has functions of searching, adding, modifying, deleting, synchronizing word-roots (from an external database or application), in order to meet the requirements of standardization of a data model. For example, in creating a data model, when a column is added and is given with a Chinese name, a recommended English name will be automatically given according to the Chinese name and the word-roots, which facilitates designers to standardize the model structure. Word-root management is a function in the system-level and the tenant-level.

[0060] A reverse engineering module is configured to reversely generate a data model from an existing table structure (i.e. physical model) in database, which is a concept in data model design/development.

[0061] A version managing module is configured to manage versions of the entire project.

5 For example, a new version may be created to modify a data model structure, or add a data model, or delete a data model. These modifications are made on the basis of the latest version, and data of an original version remains unchanged. If an error is found in the modified version, the original version can be directly used as the latest version, or the modified version may be compared with the original version to check the problem.

10 **[0062]** A log managing module is configured to record various sensitive operations in the system, where different roles are allowed to view different logs. A project administrator is allowed to view operations such as creating/deleting modules, creating/deleting tables, and submitting for examination in the current project. A space administrator is allowed to view operations such as creating/deleting projects, deactivating/enabling members. A system
15 administrator is allowed to view operations such as login, creating/deleting tenants, and creating/deleting/synchronizing data standards.

[0063] The log managing module and the project managing module are provided in parallel, not only the project has requirements for log management, but also the workspace and the entire system have requirements for log management. The project administrator needs to
20 know what sensitive operations occur in the current project, such as deleting tables and deleting modules. The workspace (tenant) administrator needs to know what sensitive operations occur in the current tenant, such as deleting projects and deleting workspace members. The system administrator needs to know what sensitive operations occur in the current system, such as login, deleting workspaces, and deleting users.

25 **[0064]** A data type managing module is configured to map data types between different databases, which facilitates the conversion of a data model between different databases. This function is only available to the system administrator.

[0065] Figure 3 is a flowchart illustrating a development process according to an embodiment of the present disclosure. As shown in Figure 3, the development process
30 includes the following steps.

[0066] The system administrator logs in to manage tenants (workspaces). The management includes: creating tenants, disabling (deactivating) tenants, enabling tenants, designating workspace administrators, maintaining a mapping relation of data types, and maintaining system-level data standards and word-roots. The platform supports multiple types of databases, and requires converting a data model between different types of databases. Therefore, the mapping relation between data types of different databases is necessary.

[0067] After logging in, the workspace administrator is allowed to create projects, designate project managers, maintain workspace members, and maintain private standards and word-roots. The private standards and word-roots can only be used in a project to which the workspace belongs. The workspace administrator is also allowed to participate in project development. The workspace is also referred to as the tenant. The workspace administrator is only one of the workspace members. After the workspace administrator maintains the workspace members, the project administrator may select from existing members when maintaining project members, instead of adding a new member.

[0068] After logging in, the project administrator is allowed to view all the projects managed by himself, to maintain project properties and/or configure project parameters, which include: basic information and parameters (whether to generate primary keys or foreign keys, whether to enforce verification) of the project, team members of the project, table naming rules (i.e., model naming rules), data sources (development library configuration), data sources of reverse engineering, module configuration, demand response, and the like. The project administrator is also allowed to participate in development. The terms “table” and “model” refer to the same thing, only being different in concepts, i.e., the term “model” is an abstract concept, and the term “table” is a concretization of the abstract concept. The module configuration is used by the project manager to uniformly configure module hierarchy of the project, which is mainly used to decompose work tasks to facilitate project management.

[0069] After logging in, the developer is allowed to view projects that they participate in, switch between the projects, create new data models, modify data models, import/export model documents in the excel format, and submit the designed models to a node corresponding to the project manager (i.e., the project administrator) for examination. Models can be modified after being duplicated to reduce the workload of development. Models in the examining process cannot be modified. Once an approved model is modified, the modified

model is in a draft state. In version management, the model in the draft state will not be created in the specified database. Models that do not pass the examination are rejected and then modified by the developer. The fields of the model are selected from existing standards, so as to create the model that conforms to the data standard. The standards here include system-level data standards and/or private standards. System-level data standards can be used globally (i.e., for all projects). Private data standards and word-roots can only be used by projects within a certain workspace. After a Chinese name is entered in a customized field, an appropriate English name will be automatically recommended based on the word-roots. The option of “whether to enforce verification” of the project controls whether to perform the standard verification when submitting the model for examination. If not enforcing verification, only verification prompt is displayed. Otherwise, the model is required to meet the standard. If it does not meet the standard, it is not allowed to be submitted for examination.

[0070] The developer may perform the following operations in the development of the model: 1. creating and/or modifying a data model; 2. exporting documents in the DDL (data definition language)/Excel format; 3. model duplication; 4. reverse engineering; 5. importing a model document in the Excel format; 6. model handover, which means that after a member of a project team is replaced, the model developed by the replaced member is required to be handed over to nodes corresponding to another developer, to facilitate later maintenance; 7. standard verification, which is implicitly and automatically executed before the submitting, instead of being performed by the developer.

[0071] The development environment provides a visualized presentation method and a table presentation method, which can be switched one to another.

[0072] Figure 4 is a schematic diagram illustrating a functional architecture of a data model developing platform according to an embodiment of the present disclosure. As shown in Figure 4, the data model developing platform includes the following modules.

[0073] A tenant managing module is configured to provide a multi-tenant environment which realizes isolation of resources. After logging in, a member of a tenant is allowed to view only the projects of the tenant to which the member belongs. The tenant managing module includes one or more of the following multiple sub-modules: a tenant creating sub-module, a tenant modifying sub-module, a tenant enabling sub-module, a tenant disabling sub-module, a tenant deleting sub-module, a tenant handover sub-module. The isolation of

resources refers to the isolation of projects of different tenants. For example, some projects are confidential, which should not be viewed by others. With the isolation between tenants (workspaces), the contents which are not expected to be viewed by others may be put in another tenant (workspace).

5 **[0074]** The tenant creating sub-module is configured to create a tenant for a user. The term “tenant” has the same meaning as the term “workspace” in this system. A tenant is necessary for creating a new project and performing subsequent processes.

[0075] The tenant modifying sub-module is configured to modify the tenant, which includes modifying basic information of the tenant, including name, password or the like.

10 **[0076]** The tenant enabling sub-module is configured to enable a disabled tenant.

[0077] The tenant disabling sub-module is configured to disable a tenant. A current tenant may be deleted after being disabled.

[0078] The tenant deleting sub-module is configured to delete the disabled tenant, and delete all projects, models, private data standards and word-roots belonging to the disabled
15 tenant.

[0079] The tenant handover sub-module is configured to, in response to replacing a tenant administrator, hand over a tenant to a node corresponding to another person.

[0080] A data type managing module is configured to match data types between different databases to facilitate the conversion of the data model between different data types. The data
20 type managing module includes a data type querying sub-module and a data type maintaining sub-module. The data type querying sub-module is configured to retrieve a configured mapping relation between data types of different databases. The data type maintaining sub-module is configured to add, modify, and delete the mapping relation between data types of different databases.

25 **[0081]** A data standard managing module includes a data standard maintaining sub-module, configured to maintain a data standard. The maintenance of the data standard includes performing one or more of adding, deleting, modifying, querying, and synchronizing on the data standard. The synchronization of the data standard includes: synchronously acquiring the data standard from an external database or an external application. The above querying refers
30 to querying the data standard in the local database, which may adopt but not limited to the

method of fuzzy retrieval. The data standard managing module further includes a word-root maintaining sub-module, configured to maintain the word-roots. The maintenance of the word-roots including one or more of retrieving, adding, modifying, deleting, and synchronizing the word-roots. The synchronization of the word-root includes synchronously acquiring the word-root from external databases or external applications. The retrieving may include querying a local database by, for example, the method of fuzzy retrieval.

[0082] A data standard maintaining sub-module is configured to maintain local data standards and external data standards. The maintenance includes functions such as adding, deleting, modifying, querying, and synchronizing the data standards. The data standards can be used in maintenance of the data model. The synchronization of the data standards refers to synchronization with an external database or external application. The IP address and port of the external database or external application server are designated in deployment. The system automatically performs the synchronization one time every day. When some data in the designated external database or external application server is modified, if it is desired to view modified data in the data model developing platform immediately, the synchronization function here may be used to synchronize the latest modified data.

[0083] The data model managing module includes one or more of a data model statistics sub-module, a data model maintaining sub-module, a data model duplicating sub-module, a data model import and export sub-module, and a data model releasing sub-module.

[0084] In addition, according to the characteristics of the hierarchical design of the model of the data warehouse, the function of model duplication is provided in an embodiment of the present disclosure. A designer of the data warehouse duplicates models sharing the same subject and makes appropriate modifications on the models to form the hierarchical models of the subject, which greatly improves the work efficiency. In some embodiments, a data model duplicating sub-module is configured to duplicate s data model to form a first replica model in response to an operation instruction for modifying the data model, where only the first replica model is allowed to be modified by the user; and/or, searching for an existing data model having a target subject according to a matching condition and duplicate the existing data model to form a second replica model, in response to an operation instruction for forming a data model having the target subject issued by the user, and receive a modification operation on the second replica model by the user, to generate an updated data model having the target

subject. The model duplication is mainly applied in the construction of a hierarchical data warehouse or data mart. Models in different layers generally have dimensions collapsed layer by layer. In this case, it is very suitable to duplicate a model in a lower layer, and then modify (deleting or modifying) the dimension to form a model in a higher layer, which can greatly
5 reduce the workload.

[0085] In some embodiments, in consideration of working habits of developers, a function of importing data from an external Excel file is provided, so that users can organize data according to a template and import the data in batches, thereby improving the work efficiency. The data model import and export sub-module is configured to import and export model
10 documents in a format of Excel; or import model documents in a format of Excel and export model documents in a format of data definition language (DDL).

[0086] In some embodiments, the data model statistics sub-module is configured to track a progress of a current project. The function of the data model statistics sub-module facilitates the project manager to track the current project to know the progress of the project. The
15 module may implement the following steps S1 to S3.

[0087] In S1, for different versions made by project developers in different time periods, the number of unfinished data models, the number of approved data models, and the number of all data models are acquired.

[0088] In S2, the number of unfinished data models, the number of approved data models,
20 and the number of all data models are counted in groups according to time period. Specifically, the number of current data models, the number of approved data models, and the total number of data models (including all data models that are in the approved status, the examining status, the rejected status, and the drafting status) are counted in the time period. The time period may be determined arbitrarily, to help the project manager to know the development situation
25 of the model in various time periods. This function is very useful for medium or large projects.

[0089] In S3, statistical results are displayed in a form of pie charts, trend charts, or time axes.

[0090] Thus, the project manager can view the number of data models for each project
30 developer and each version on the graph. The date period can be freely set for dynamic

statistics.

[0091] In some embodiments, the data model maintaining sub-module is configured to perform one or more functions of adding, deleting, modifying, and submitting the data model.

[0092] In some embodiments, the data model releasing sub-module is configured for
5 integration and delivery of the data model. The model designed by the project team will eventually be physicalized into the database. The releasing of the data model is to deliver the current version of the data model to a predetermined environment, such as development environment, test environment, and production environment. The version management function can meet the requirements of continuous delivery. A next version can overwrite the
10 old version, or incrementally upgrade on the basis of the old version.

[0093] The project management function module may include one or more of the following sub-modules: a project creating sub-module, a project modifying sub-module, a project manager designating sub-module, a project member managing sub-module, a service system maintaining sub-module, a data model layer maintaining sub-module, a data model naming
15 rule sub-module, a data source managing sub-module, a project handover sub-module, a data model handover sub-module, a version managing sub-module, a logical data model examining sub-module, a physical data model examining sub-module, and a reverse engineering sub-module.

[0094] The project creating sub-module is configured to create a project to which a data
20 model to be developed belongs, and decompose the project according to a decomposition rule. The decomposition rule is based on subjects and/or service systems. For example, subjects in a project of a data warehouse may include customers, institutions, accounts, and transactions. After a project is divided into sub-models in this way, the workload of each module is greatly reduced, which facilitates project management.

[0095] For a relatively large project that is a data model developed for multiple service
25 systems, such as a project including service systems such as CRM and HR, the data model may be divided into a CRM part and an HR part, which are assigned to different project teams. In this way, a relatively large project is decomposed into relatively small tasks. For another example, a project of a data warehouse is required to be designed according to the subjects
30 such as customers, institutions, accounts, transactions, contracts, quotas, and collaterals. The relatively large project can be divided into several parts according to the subjects, which are

designated to different teams. The above method is to divide a large project into multiple small parts to be respectively developed, which facilitates project management. Certainly, in this embodiment of the present disclosure, the project may be alternatively treated as an overall large project, for which a module hierarchy is created according to the service systems or subjects. The method of dividing a project is not limited herein.

[0096] The project modifying sub-module is configured to modify basic information of the project, such as a name, and start and end dates.

[0097] The service system maintaining sub-module is configured to maintain a service system to which the project belongs.

[0098] The model layer maintaining sub-module is configured to maintain a layer to which the data model belongs in the project, where the service system generally belongs to a top layer.

[0099] The data source managing sub-module is configured to manage a database used for physicalization and reverse engineering of the data model. The function of the data source management sub-module includes managing connection information of the database of the current project, including an IP address, a port, a database, a Schema, an account and a password. One project may have multiple data sources, one of which may be selected for physicalizing the model into a database. Another data sources may be selected for performing reverse engineering.

[0100] The data model verifying sub-module is configured to verify a designed data model using the data standard and submit the verified data model to the project manager. A physical data model verifying sub-module is integrated with a logical data model verifying sub-module to form the data model verifying sub-module. Since developers are not required to know the difference between the logical model and the physical model, these two functions are automatically processed in background according to the embodiment of the present application, without being processed by the front end. The data standard is a concept in the field of data governance, which is used to standardize the structure of the data models. Generally, the data standard can be understood as that each semantic meaning is represented by a single word, so as to avoid the situation that the same semantics meaning is represented by different words in different models, which may cause trouble for metadata management and data lineage management in data governance. The way of verifying is to compare with the

user-defined data standard. If any one of the name, code, data type, length, and precision defined in the model is not consistent with the data standard, it is determined that it does not meet the requirements of the data standard. The verification function using the data standard is controllable. The option of enforcing verification may be disabled when creating a new project, so that the verification function using the data standard will not be performed before submission.

[0101] The project handover sub-module is configured to: in response to replacing a project manager, hand over a project corresponding to the replaced project manager to a node corresponding to a designated person.

[0102] The model handover sub-module is configured to: in response to replacing a project developer, hand over a data model designed by the replaced project developer to nodes corresponding to other project developers in the project. In order to cope with the change of members of a development team, functions of project handover and model handover are provided according to the embodiment of the present disclosure. Log is recorded for sensitive operations (such as user login, deleting user, deleting tenant, deleting model, and the like), to ensure that problems can be traced back. In some embodiments, a project is handed over to a target member in the workspace. Generally, the persons involved in the handover are negotiated offline, and the handover is performed online. The original role of the target member is not limited. A model can be handed over to only members in the project, following the rules the same as that of the project handover.

[0103] The reverse engineering sub-module is configured to convert a table in a database into a data model. A part of tables may be imported to the reverse engineering sub-module based on a predetermined condition.

[0104] The data model naming rule sub-module is configured to maintain a naming rule of the data model in the project, so that models have a unified style of naming, which facilitates project management.

[0105] A data model is an abstract expression of data characteristics, which describes the static characteristics, dynamic behavior and constraints of the system in an abstract manner, and provides an abstract framework for the information representation and operation of the database system. The content described by the data model includes data structure, data operation and data constraints. Data models are classified into conceptual data models, logical

data models, and physical data models according to the application level. The conceptual data model is a user-oriented, objective-world-oriented model, mainly used to describe the conceptual structure of the world, which is used by designers of the database in the initial stage of design to analyze data and the connections between data without considering a specific technique for computer systems and DBMS. Therefore, the conceptual data model is not related to any specific data management system (DBMS). The conceptual data model is required to be converted into a logical data model in order to be implemented in the DBMS. The logical data model is a database-oriented model, and is a data model supported by a specific DBMS, such as a network data model (Network Data Model) and a hierarchical data model (Hierarchical Data Model). The logical data model is both user-oriented and system-oriented, and is mainly used in the implementation of database management systems (DBMS). The physical data model (Physical Data Model) is a model oriented to the physical representations of computers, which describes the structure of data on the storage medium. It is not only related to the specific DBMS, but also related to the operating system and hardware. Each kind of logical data model corresponds to a physical data model for implementation. Since the mapping relation between data types of the logical model and the physical model is provided according to the embodiment of the present disclosure, a logical model can be automatically converted into a physical model.

[0106] Further, the application server in background may further include: a log managing sub-module, configured to generate log records for sensitive operations, the sensitive operations includes at least one of user login, deleting a user, deleting a tenant, and deleting a data model.

[0107] The above technical solutions according to the embodiments of the present disclosure have the following advantageous technical effects.

[0108] The embodiments of the present disclosure are of great help to the design of large databases or data warehouses.

[0109] The distributed deployment scheme can meet the requirement that team members do not work in the same office, so as to be not limited by geographical spaces.

[0110] The online working mode facilitates team collaboration and knowledge sharing among team members, thereby improving the work efficiency.

[0111] The version control realizes the traceable version iterations in the development process, which meets the needs of continuous integration.

[0112] With the introduction of data standards, data structures of models can be standardized, which meets the requirements on standard data in the field of data governance.

5 **[0113]** A method for distributed development of data model is provided according to the embodiments of the present disclosure, with which data models are standardized by using data standards in the design stage, and multiple versions can coexist in the development of the data model. Multiple coexisting versions can be understood as branches or snapshots of the data produced by a developer in different periods. The developer can continue development based
10 on any one of the snapshots.

[0114] Those skilled in the art can clearly understand that, for convenience and conciseness of description, the above example of division of functional units and modules is described. In actual applications, the functions may be allocated to different functional units and modules as required. That is, the internal structure of the device can be divided into different functional
15 units or modules, to perform all or part of the functions described above. The functional units and modules in the embodiments may be integrated into a same processing unit, or may be implemented in separate physical units, or two or more units may be integrated in the same unit. The above integrated units may be implemented in a form of hardware, or in a form of software function units.

20 **[0115]** The foregoing descriptions are merely preferred embodiments of the present disclosure, but are not used to limit the protection scope of the present disclosure. Any modification, equivalent substitution, or improvement made within the spirit and the principle of the present disclosure shall fall within the protection scope of the present disclosure.

CLAIMS

1. A data model developing platform, having a B/S three-tier architecture, and comprising:

- 5 a page client in foreground, configured to access an application server through Nginx;
the application server in background, deployed with application services associated with data model development, and
a data storage device, configured to store data using a relational database, and cache data using a key-value database,

10 wherein the application server in background comprises:

a data standard managing module, configured to maintain a data standard and word-roots;

a data model managing module, configured to design a data model based on the data standard and/or the word-roots;

15 a project managing module, configured to compare any two versions of a project to generate a difference file, and export the difference file into an SQL file; and

a tenant managing module, configured to provide a multi-tenant environment that isolates projects of different tenants from each other, so that each tenant is allowed to view only an owned project.

20 2. The data model developing platform according to claim 1, wherein the data standard managing module comprises:

a data standard maintaining sub-module, configured to maintain the data standard by performing one or more of maintenance operations of adding, deleting, modifying, querying, and synchronizing the data standard, wherein the synchronizing the data standard comprises:
25 synchronously acquiring the data standard from an external database or an external application; and

a word-root maintaining sub-module, configured to maintain the word-roots by performing one or more of maintenance operations of retrieving, adding, modifying, deleting and synchronizing the word-roots, wherein the synchronizing the word-roots comprises:

synchronously acquiring the word-roots from an external database or an external application.

3. The data model developing platform according to claim 2, wherein the data model managing module comprises:

a data model duplicating sub-module, configured to:

5 duplicate a data model to form a first replica model, in response to an operation instruction for modifying the data model, wherein only the first replica model is allowed to be modified by a user; and/or

search for an existing data model having a target subject according to a matching condition and duplicate the existing data model to form a second replica model, in response to
10 an operation instruction for forming a data model having the target subject issued by the user, receive a modification operation on the second replica model by the user, and generate an updated data model having the target subject.

4. The data model developing platform according to claim 3, wherein the data model managing module further comprises:

15 a data model statistics sub-module, configured to track a progress of a current project;

a data model maintaining sub-module, configured to perform one or more of operations of adding a data model, deleting a data model, modifying a data model, and submitting a data model;

20 a data model import and export sub-module, configured to import and export a model document in a format of Excel; or import a model document in a format of Excel and export a model document in a format of data definition language (DDL); and

a data model releasing sub-module, configured to deliver a final version of a data model to a predetermined environment.

25 5. The data model developing platform according to claim 1, wherein the project managing module comprises:

a project creating sub-module, configured to create a project to which a data model to be developed belongs, and decompose the project according to a decomposition rule, wherein the decomposition rule is based on subjects and/or service systems;

a project manager designating sub-module, configured to designate a project manager in

response to a selecting instruction or a configuring instruction;

a project member managing sub-module, configured to manage developers in a development team of the project by performing one or more of: adding a developer, deactivating a developer, enabling a developer, and deleting a developer.

5 6. The data model developing platform according to claim 5, wherein the project managing module further comprises:

a project modifying sub-module, configured to modify basic information of the project;

a service system maintaining sub-module, configured to maintain a service system to which the project belongs;

10 a model layer maintaining sub-module, configured to maintain a layer to which data models in the project belong;

a data model naming rule sub-module, configured to maintain a naming rule of the data models in the project, so that the data models have a unified style of naming;

15 a data source managing sub-module, configured to select and determine a database corresponding to physicalization of the data models and a database corresponding to reverse engineering; and

a data model verifying sub-module, configured to verify a designed data model using the data standard and submit the verified data model to a node corresponding to the project manager.

20 7. The data model developing platform according to claim 6, wherein the project managing module further comprises one or more of:

a project handover sub-module, configured to: in response to replacing a project manager, hand over a project corresponding to the replaced project manager to a node corresponding to a designated person;

25 a model handover sub-module, configured to: in response to replacing a project developer, hand over a data model designed by the replaced project developer to a node corresponding to another project developer of the project; and

a reverse engineering sub-module, configured to convert a table in a database into a data model.

8. The data model developing platform according to any one of claims 1 to 7, wherein the tenant managing module comprises one or more of:

a tenant creating sub-module, configured to create a tenant;

a tenant modifying sub-module, configured to modify basic information of the tenant;

5 a tenant enabling sub-module, configured to enable a disabled tenant;

a tenant disabling sub-module, configured to disable a current tenant;

a tenant deleting sub-module, configured to delete the disabled tenant, and delete all projects, models, private data standards and word-roots belonging to the disabled tenant; and

10 a tenant handover sub-module, configured to: in response to replacing a tenant administrator, hand over a tenant to a node corresponding to another person.

9. The data model developing platform according to any one of claims 1 to 7, wherein the application server in background further comprises:

15 a data type managing module, configured to maintain mapping relations between data types of different databases, wherein the mapping relations are used to convert the data model between different databases, wherein the data type managing module comprises:

a data type querying sub-module, configured to retrieve a configured mapping relation between data types of different databases; and

a data type maintaining sub-module, configured to add, modify, and delete a mapping relation between data types of different databases.

20 10. The data model developing platform according to any one of claims 1 to 7, wherein the relational database comprises a MySQL database, an external database is connected with the MySQL database through an RESTful interface, and the MySQL database acquires the data standard and the word-roots from the external database;

25 the application server in background further comprises: a log managing sub-module configured to generate a log record for a sensitive operation, wherein the sensitive operation comprises at least one of: user login, deleting a user, deleting a tenant, and deleting a data model.