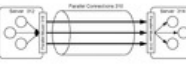
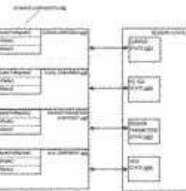
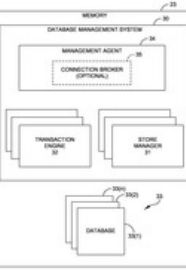
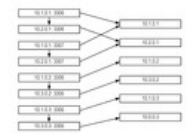
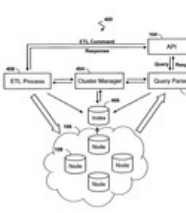
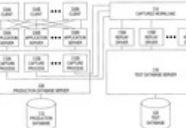
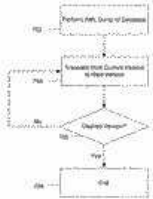
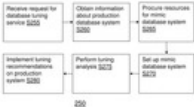
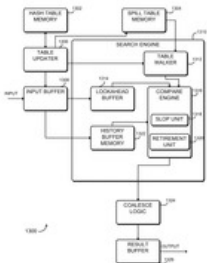
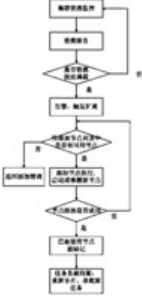
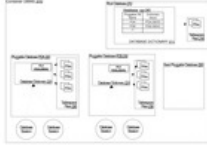
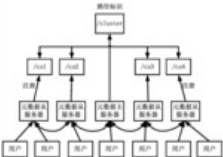
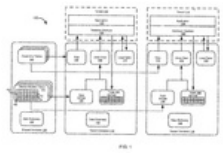
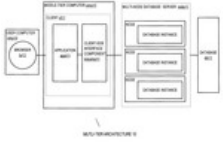
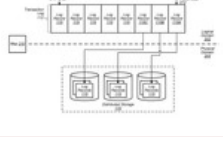
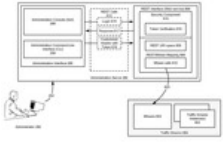
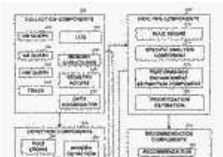
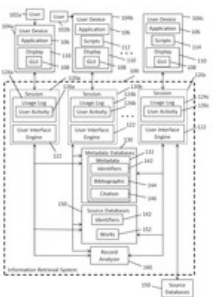
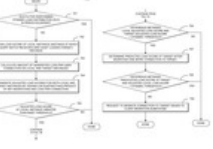
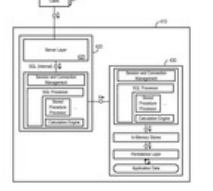
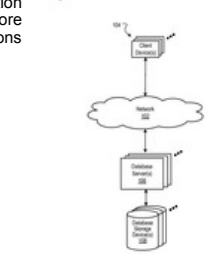


940221 results for Similar patents - Collection: FAMPAT

#	 	Title	Publication number	Publ. date	Applicant/Assignee	
1.		Dynamic deactivation of cold database in database service	EP3479234	2019-05-08	MICROSOFT TECHNOLOGY LICENSING MICROSOFT	
Managing databases implemented in a cloud computing environment. A method includes detecting that a database implemented in the cloud computing environment is in a state of non-use. The method further includes as a result of detecting that a database implemented in the cloud computing environment is in a state of non-use, instantiating a workload in the cloud computing environment to deactivate the database. The workload is configured to store metadata for the database and database data remotely in cloud storage such that the database can be reactivated at a later time.						
2.		System including a middleware machine environment	EP2616966	2013-07-24	ORACLE INTERNATIONAL	
(EP2616967) A system and method for providing a middleware machine or similar platform. In accordance with an embodiment the system (referred to herein in some implementations as "Exalogic") comprises a combination of high performance hardware, together with an application server or middleware environment, to provide a complete Java EE application server complex which includes a massively parallel in-memory grid, can be provisioned quickly, and can scale on demand. In accordance with an embodiment, the system can be deployed as a full, half, or quarter rack, or other configuration, that provides an application server grid, storage area network, and InfiniBand network, which support the execution of an application server, middleware or other functionality such as, for example, WebLogic Server, JRockit or Hotspot JVM, Oracle Linux or Solaris, and Oracle VM. Additional features of the system can include, e.g. Zero Buffer Copies, Scatter/Gather I/O, T3 Connections, and Lazy Deserialization.						
3.		Transparent migration of stateless sessions across servers	EP1654645	2006-05-10	ORACLE INTERNATIONAL	
(EP1654858) Techniques allow a client to be migrated from a session that has been established for a client on a server to another session on another server in a way that is transparent to the application for which the initial session was established. The sessions are migrated under conditions in which a copy of session state of the session to migrate from the source node does not have to be captured and restored to a destination session.						
4.		Database management system with database hibernation and bursting	WO2014/168913	2014-10-16	NUODB	
A data processing network includes a database management for a plurality of databases serviced by a server comprising a plurality of data processors. Each server data processor includes a copy of the database management system and a monitor for measuring and analyzing various parameters for each server data processor and each included database and performing various methods transparently to a user. For example, when a parameter or combination thereof indicates that a database in one data processor is inactive or requires additional resources, that database is set to an inactive state or is transferred to another data processor with greater resources, respectively. An inactive database is recalled upon receipt of a request for access to that database. A database in a data processor of greater resources can be returned to a server data processor or another data processor with fewer resources. These operations are transparent to database users and independent of user or administrator intervention.						
5.		Flexible scheduling in a database system	WO2017/007684	2017-01-12	ALIBABA HOLDING	
Apparatuses and methods are disclosed for database scheduling. An exemplary method may include obtaining a performance parameter associated with each of a plurality of database servers that are subject to scheduling. The method may also include selecting, among the plurality of database servers subject to scheduling, a first database server, wherein the performance parameter associated with the first database server exceeds a predetermined threshold. In addition, the method may include selecting a first database instance operating on the first database server. Moreover, the method may include migrating the first database instance to a second database server that is capable of accepting the first database instance.						
6.		Data management and indexing across a distributed database	EP2859473	2015-04-15	CITIBANK RACKSPACE	
In one embodiment, a distributed database system supporting flexible configuration of data clusters is disclosed. The system includes a cluster manager, an index, and a dataset distributed over one or more database clusters. Where the nodes of the clusters may report ownership of a particular range, the index contains an alternate range. The cluster manager receives requests to access a range of data within database and queries the index to determine the appropriate nodes and/or clusters with which to connect. The cluster manager then directs the request or to connect to the specified nodes and/or clusters.						
7.		Capturing and restoring database session state	US20100005097	2010-01-07	ORACLE INTERNATIONAL	
Techniques are described herein for capturing and restoring database session state. Production database server components save the session state of each of a plurality of database sessions. The components store workload units that are processed in these sessions. The components store updated session states in response to certain events. Thus, the components may capture multiple session states, pertaining to various different points in time, for each session. The captured session states and the captured workload are moved to a test database server. A user selects, from among the time points represented by the session states, a point in time at which the user would like workload replay to begin. Sessions are re-created on the test database server. Session states of these sessions are set to reflect the session states as they existed at the user-selected time point. Workload units are replayed in the sessions relative to the test database server.						

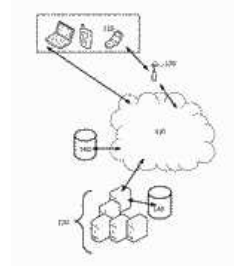
8.	Database migration	US8364631	2013-01-29	INFOBLOX	A technique for converting a first version of a database to a second version is disclosed. The technique includes determining available translation steps, selecting a translation path from the first version of the database to the second version, and executing the selected translation steps in the translation path. The translation path includes selected translation steps from among the available translation steps. 
9.	Query tuning in the cloud	US20160140176	2016-05-19	IBM	(US9996582) Tuning a production database system through the use of a remote mimic. In response to receipt of a query tuning request against a database system, information about that system is obtained and a mimic of the system is set up in a remote system environment. The mimic aims to imitate the database system in all relevant ways with respect to the tuning request. A tuning analysis is then performed on this mimic system such that there is substantially no impact to operations of the original database system. Tuning results are then applied to the original database system. The entire process takes place with little or no human intervention. 
10.	Flexible in-memory column store placement	EP3519985	2019-08-07	ORACLE INTERNATIONAL	Techniques are described herein for distributing distinct portions of a database object across volatile memories of selected nodes of a plurality of nodes in a clustered database system. The techniques involve storing a unit-to-service mapping that associates a unit (a database object or portion thereof) to one or more database services. The one or more database services are mapped to one or more nodes. The nodes to which a service is mapped may include nodes in disjoint database systems, so long as those database systems have access to a replica of the unit. The database object is treated as in-memory enabled by nodes that are associated with the service, and are treated as not in-memory enabled by nodes that are not associated with the service. 
11.	Technologies for big data analytics accelerator	EP3451196	2019-03-06	INTEL	Technologies for database acceleration include a computing device having a database accelerator. The database accelerator performs a decompress operation on one or more compressed elements of a compressed database to generate one or more decompressed elements. After decompression of the compressed elements, the database accelerator prepares the one or more decompressed elements to generate one or more prepared elements to be processed by an accelerated filter. The database accelerator then performs the accelerated filter on the one or more prepared elements to generate one or more output elements. Other embodiments are described and claimed. 
12.	A kind of method of dynamic expansion data-base cluster	CN103970907	2014-08-06	LANGCHAO ELECTRONIC INFORMATION	The invention provides a kind of method of dynamic expansion data-base cluster, the data-base cluster that is applicable to have dynamic extending comprises in resource: internal memory, CPU, disk etc. approach full load soon, the increase server node of automatic touch trigger, to cluster, is realized the real-time extension that prosthetic operation is intervened. This system provides a kind of real-time automatic expansion approach of data-base cluster, makes the data-base cluster with dynamic ability extending transversely automatically increase resource, realizes real-time expansion; Solve that hydraulic performance decline, the function bring are made mistakes when scarcity of resources and system crash etc. affects the abnormal problem of regular traffic, availability and the stability of raising system. 
13.	Container database	EP2901325	2015-08-05	ORACLE INTERNATIONAL	A container database may contain multiple database dictionaries, each database dictionary defining a pluggable database. When database sessions are established on a container DBMS, each database session is given access to a pluggable database by establishing the respective database dictionary of the pluggable database as the database dictionary for that database session. Database commands issued through database session can only access the database objects defined in the database dictionary established for the database session. 

14.	Method for improving reliability of metadata nodes of cloud databases	CN104506625	2015-04-08	G CLOUD TECHNOLOGY	
The invention relates to the technical field of cloud databases, in particular to a method for improving the reliability of metadata nodes of cloud databases. The method has the advantages that metadata are simultaneously written in a plurality of servers and are read from all the servers or written in all the servers in each procedure, accordingly, more than one copy of data is stored at any moment, more than one server runs at any moment, the simultaneous offline probability of the servers is extremely low, and the availability can be guaranteed without extra measures; the metadata servers are equal to one another, and accordingly influence on the availability and the reliability of integral metadata server clusters due to an optional offline metadata server can be prevented; the problem of single point of failure can be solved by the aid of the method; the method can be used for processing the metadata of the cloud databases.					
15.	Materializable database objects in multitenant environments	EP3567488	2019-11-13	SAP	
Techniques and solutions are described for facilitating the distribution of tables and table data between a shared container and one or more tenant containers. The tables can include a sharing indicator that can designate a table (or a view on a table) as having a materializable type. Materializable types can initially be deployed in the shared container, and a tenant container can have read access to the table via a view. If the tenant attempts to modify the materializable table, the table can be materialized in the tenant container. Metadata for a materializable table can include materialization instructions. Techniques and solutions are also provided for converting between sharing types, and for updating shared containers or table schemas.					
16.	Suspending and resuming of sessions	US20060184535	2006-08-17	ORACLE INTERNATIONAL	
A session established for an application is suspended from an instance and later resumed on the server or a different server. When a session is suspended, its state is captured and persistently stored as a session image in a data repository, and the session is terminated. When the session is resumed, a new session is established on an instance, the session image of the session to resume is retrieved from the repository and restored to the newly established session. The suspension and resumption of a session may be initiated by an application, which may request the suspension and resumption of a session.					
17.	Automated self-healing database system and method for implementing the same	WO2018/156765	2018-08-30	SALESFORCE.COM	
An automated self-healing database system is provided that includes a primary database, an application server that writes data to the primary database and reads data from the primary database when it is available, a standby database that is a replica of the primary database, and a self-healing module (SHM). The SHM can automatically detect unavailability of the primary database, and if the standby database is available, the SHM can automatically enable the standby database as readable and writable, assign the standby database the role of primary database at the primary site to start replicating data to other standby databases, and use a pre-established connection between the application server and the standby database to allow the application server to read data from the standby database and to write data to the standby database resulting in the standby database assuming role of the primary database at the primary site.					
18.	High availability database through distributed store	WO2019/028130	2019-02-07	SALESFORCE.COM	
Techniques are disclosed relating to a database system synchronizing a current state of the database system among a plurality of nodes configured to handle requests for data of the database system stored in a distributed storage with one of the plurality of nodes being currently active and the other nodes of the plurality of nodes being currently standby nodes. In various embodiments, the active node receives a request to perform a transaction that includes committing data to the distributed storage. In response to receiving the request, in some embodiments, the active node commits the data to the distributed storage to update the current state of the database system and causes storing of metadata that provides an indication of the commitment in a transaction log stored in the distributed storage.					
19.	System and method for administering server configurations including gui navigation, property sheets, and auto-tab completion	EP2761452	2014-08-06	ORACLE INTERNATIONAL	
Described herein are systems and methods for use in administering server configurations, such as in a traffic director or other type of server environment. In accordance with various embodiments, systems and methods are described for providing an administration console user interface or GUI for use in reviewing and configuring different configurable objects, wherein the GUI includes one or more of a "never-get-lost" navigation model, or property sheets; a command-line interface and/or command-line shell which provides an autocompletion feature for remote objects; and a REpresentational State Transfer (REST) infrastructure and management service which maps REST calls to mbeans or other management components registered on the administration server, for use in managing the traffic director.					
20.	Method and apparatus for enhancing performance of database and environment thereof	EP2255276	2010-12-01	DBSOPHIC	
A method and apparatus for enhancing the performance of an environment comprising a database system. The method and apparatus employ collection stage for collecting data related to substantially all aspects of the system, including hardware, operating system, database, database schema, content and activity. Multiple issues relating to multiple categories are detected within the collected information. Then the detected issues are analyzed by correlating and prioritizing them. The issues that receive the highest priority are preferably those whose correction will yield the most significant improvement on the overall performance of the system. One or more recommendations are then issued for correcting the root issues hindering performance. Preferably, for one or more recommendations, scripts are generated which are then executed manually or automatically.					
21.	System for dynamically varying traffic routing modes in a distributed cluster and method therefor	US20170085634	2017-03-23	WAL MART STORES WALMART APOLLO WALMART STORES	
A system for dynamically varying traffic routing modes in a distributed database system is presented. A database system comprises a load balancer and a database partitioned into a first and second shard. Coupled to the load balancer are a first plurality of database servers and a second plurality of database servers. Each of the first plurality of database servers contains a copy of the first shard of the database. Each of the second plurality of database servers contains a copy of the second shard of the database. Each server in the first plurality of database servers sends queries to a server in the second plurality of database servers using a round-robin system in which servers in the second plurality of database servers take turns processing incoming queries, as a method of distributing load among the multiple servers. Other embodiments also are disclosed.					

22.	System and methods for cleansing automated robotic traffic from sets of usage logs	WO2015/132678	2015-09-11	CAMELOT UK BIDCO THOMSON REUTERS	
23.	Reactive load balancing for distributed systems	EP2633420	2013-09-04	MICROSOFT TECHNOLOGY LICENSING MICROSOFT	
24.	Construction method for database system asymmetric deployment based on federated clusters	CN104504128	2015-04-08	TIANJIN NANKAI UNIVERSITY GENERAL DATA TECHNOLOGIES	
25.	System and methodology providing workload management in database cluster	US20090037367	2009-02-05	SYBASE DUBLIN	
26.	Cloud database system and cloud database resource dynamic adjustment method	CN107085539	2017-08-22	BEIJING UNIVERSITY OF POSTS & TELECOMMUNICATIONS	
27.	Security for debugging of database sessions	US20150149417	2015-05-28	SAP	
28.	Processing mutations for a remote database	EP3161685	2017-05-03	GOOGLE	

29. Transaction processing in a distributed database management system WO2016/120722 2016-08-04  CLUSTERPOINT

Systems and methods here include computerized handling of database transactions that can be employed on distributed architecture without requiring a single point of synchronization. Thus, examples can be embodied by a combination of algorithms that can be implemented in software and hardware allowing distributed deployment on the software. Such systems may be used for updating several documents within a single transaction which may provide, for example, atomicity, consistency, isolation and/or durability of such transactions. This may be accomplished by ensuring a distributed consensus within any of various database nodes, while requiring only a majority of nodes handling each individual shards to be available for transactions to complete successfully.



30. Method and apparatus for reading and writing committed data US20170344290 2017-11-30  HOROWITZ ELIOT
 MILKIE ERIC ANDREW
 SCHWERIN ANDREW MICHALSKI
 STEARN MATHIAS BENJAMIN

According to some aspects, a database system is provided. The database system may include at least one processor configured to execute a plurality of system components. The plurality of system components may include an interface component configured to receive a read commit command and provide a result, a snapshot component configured to generate a plurality of snapshots of data stored in a first data storage node of a plurality of data storage nodes and identify a committed snapshot representative of data that has been replicated on a majority of the plurality of data storage nodes, and a command processing component configured to read the committed snapshot responsive to receipt of the read commit command and generate the result.



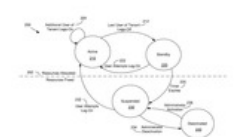
31. System and method for facilitating replication in a distributed database US20160371156 2016-12-22  MONGODB

According to one aspect, a distributed database system is configured to manage multi-writer operations on a distributed database by implementing one or more catamorphic database operators. Catamorphic operators can be architected on the system, and executed with little or no reconciliation logic. Catamorphic operators define sets of catamorphic operations and respective execution logic where the order of execution of catamorphic operations is not relevant to a final result.



32. Managing capacity in a data center by suspending tenants EP2776941 2014-09-17  MICROSOFT TECHNOLOGY LICENSING
MICROSOFT

A tenant comprising one or more users receives services from a cloud computing system ("CCS"). The CCS determines when a tenant is not using any services, and if so, the CCS initiates a tenant suspension process. The tenant suspension process involves suspending various asynchronous computing processes for the tenant, which execute otherwise execute even if no users of the tenant are currently accessing the CCS. After a period of non-use of the CCS, the tenant is suspended. The tenant is returned to an active state automatically when a user of the tenant logs-on to receive services.



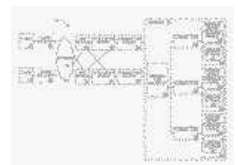
33. A kind of data processing method, system and metadata updates method, system CN110309215 2019-10-08  XIAMEN WANGSU

The invention discloses a kind of data processing method, system and metadata updates methods, system, wherein, the data processing method includes: the processing result for receiving object storage device and being directed to data processing request response, and it is based on the processing result, generate the corresponding operation log of the data processing request; Judge whether current primary database cluster can be used, if available, the operation log of generation is written in the primary database cluster; If unavailable, the operation log of generation is written in standby data-base cluster; After the primary database cluster recovery is available, untreated operation log is imported into the primary database cluster in the standby data-base cluster. Technical solution provided by the present application can be improved the stability and availability of distributed memory system.



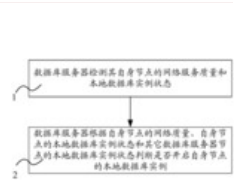
34. System and method for subunit operations in a database US201110029498 2011-02-03  IDEMITSU UNITECH
XKOTO

A method for processing a transaction request at a database load balancer is disclosed. The method comprises receiving the transaction request, where the transaction request is comprised of one or more operations to be performed upon a database; analyzing the transaction request to determine the one or more operations; analyzing the transaction to determine one or more database units associated with the operations; analyzing the transaction requests and the one or more database units to determine one or more database subunits associated with the database units that the transaction request will perform one or more operations upon; associating one or more database locks with each of the one or more operations; analyzing one or more of the database locks to determine one or more sequence numbers associated with each of the one or more operations; and transmitting one or more unique identifiers for each of the database subunits, the one or more operations with the associated database locks and the sequence numbers to one or more database servers accessible to the database load balancer. Systems, apparatus and computer programming associated with or implementing the method are also disclosed.



35. Method and system for realizing high availability of database server based on shared storage CN104573117 2015-04-29  CERTUSNET

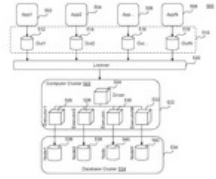
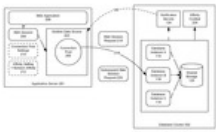
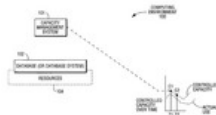
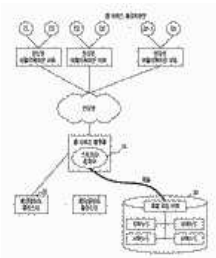
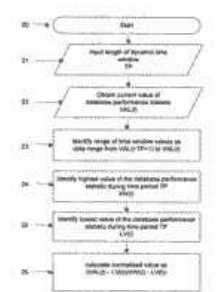
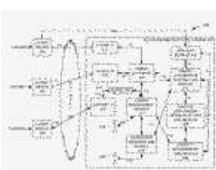
The invention relates to a method for realizing the high availability of a database server based on shared storage. The method comprises the following steps: the database server detects the network service quality and the local database instance state of the database server; the database server judges whether the local database instances of nodes of the database server are started up or not according to the network quality and the local database instance state of the database server and the local database instance states of other database servers. In addition, the invention also relates to a system for realizing high availability of the database server based on shared storage. Each database server of the system comprises a service detection module and an instance management module. According to the method and the system for realizing the high availability of the database server based on shared storage, provided by the invention, high availability is realized by a mode that the instance nodes detect themselves, the nodes supervise other node instance states and the process is self-managed, no difference among master nodes and standby nodes exists, and the third-party nodes are not needed. The method and the system have the advantages of simple structure, easiness in realization, and relatively broad application range.

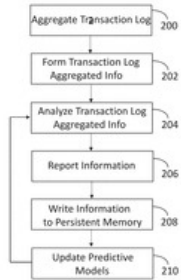
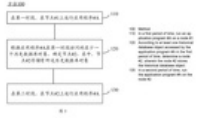
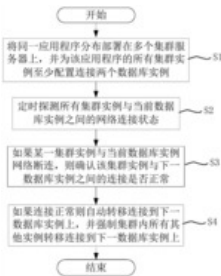
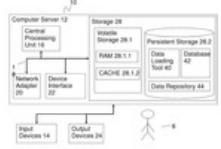
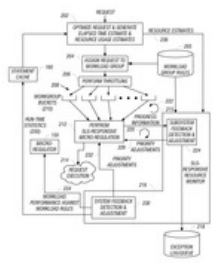
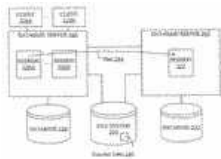


36. Policy driven data placement and information lifecycle management EP2901323 2015-08-05  ORACLE INTERNATIONAL

A method, apparatus, and system for policy driven data placement and information lifecycle management in a database management system are provided. A user or database application can specify declarative policies that define the movement and transformation of stored database objects. The policies are associated with a database object and may also be inherited. A policy defines, for a database object, an archiving action to be taken, a scope, and a condition before the archiving action is triggered. Archiving actions may include compression, data movement, table clustering, and other actions to place the database object into an appropriate storage tier for a lifecycle phase of the database object. Conditions may optionally invoke user-defined functions, and may be based on access statistics specified at the row level and may use segment or block level heatmaps. Policy evaluation occurs periodically in the background, with actions queued as tasks for a task scheduler.



37.	Mass insertion into single-threaded databases	WO2019/125703	2019-06-27	ROKU	Disclosed herein are system, method, and computer-readable device embodiments for mass insertion into single-threaded databases. An embodiment includes a processor and a memory, a storage layer to interface with a plurality of software applications and to receive data output from the plurality of software applications, and a listener that runs according to an update policy, to detect the presence of information newly stored within the storage layer. The processor and memory may be configured to maintain at least a part of a running database cluster including a plurality of nodes, with at least two nodes configured to run without multi-threading, and to execute an intermediate module to send at least part of the information to the database cluster, and to perform simultaneous access to multiple database nodes running without multi-threading. 
38.	System and method for providing session affinity and improved connectivity in a clustered database environment	EP2724265	2014-04-30	ORACLE INTERNATIONAL	A clustered database environment (e.g. Oracle Real Application Cluster (RAC)) includes multiple database instances that appear as one server. An application server (e.g. WebLogic Server (WLS)) can use a data source (e.g. an Oracle GridLink data source) and connection pools to connect with the clustered database. In accordance with an embodiment, a data source configuration allows for specification of a preferred affinity policy, such as a data affinity, temporal affinity, and/or session or session-based affinity policy. In accordance with an embodiment, the system includes a number of features that improve application connectivity in the clustered database environment, including a select-only case for application continuity, wherein an application-independent infrastructure, e.g. implemented within a Java Database Connectivity (JDBC) driver, enables recovery of work from an application perspective and masks system communications, hardware failures and hangs. 
39.	System maintenance and tuning of databases by using excess capacity in capacity controlled environment	US20130085984	2013-04-04	IDEMITSU UNITECH	(US8688629) Excess capacity available to a database system in a capacity controlled environment can be effectively managed for maintenance and/or tuning activities. In this context, excess capacity can be used as needed and/or on a temporary basis. Furthermore, maintenance and/or tuning activities can be performed without requiring the use of the configured or allotted capacity. As a result, these operations can be performed without adversely affecting other operations which may be deemed more critical by the users of databases. System maintenance and tuning (or fine tuning) of a database environment can, for example, include operations associated with "Physical Tuning," and "Index Tuning". Some exemplary operations include collecting Statistics, analyzing a database workload, sampling, analyzing an execution plan, creating indexes. 
40.	Distributed storage system based on a metadata cluster and a method thereof, particularly for providing the entire storage as a single storage image	KR20100048130	2010-05-11	KT	PURPOSE: A distributed storage system based on a metadata cluster and a method thereof are provided to supply a storage service regardless of the amount of storage and the number of clients by synchronizing meta data on a cache memory in real time. CONSTITUTION: A storage relay server(10) relays a storage request for a web service client to a meta data cluster(20), and transfers the meat data of the meta data cluster to a file storage server(30). The meta data cluster generates the meta data, and stores the meta data at a cache memory. In addition meta data cluster synchronizes the cache memory through the meta data of the other meta data cluster. COPYRIGHT KIPO 2010 
41.	Request response method in cloud sea data and computer readable storage medium	CN107729551	2018-02-23	ZHENGZHOU YUNHAI INFORMATION TECHNOLOGY	The invention discloses a request response method in cloud sea data and a computer readable storage medium. The method comprises the following steps: starting a primary database and a slave database of a database and simultaneously responding to a request; receiving an access request of the database; identifying whether an operation mode of the access request is a write operation or not; when theoperation mode of the access request is identified as the write operation, transmitting the access request to the slave database of the database, and informing the slave database of responding to theaccess request. 
42.	Real-time database performance and availability monitoring method and system	US20110035363	2011-02-10	DBA INFOPOWER ENTEROS	Database performance and availability monitoring of changes impacting database performance, availability and continuity to the underlying business may be performed. A method for doing so may include analytical and visual real-time analysis engines to identify and provide alert notifications on changes in database performance statistics (such as CPU consumption, physical I/O, etc.) related to a sample period of time on a single database or across multiple databases. Result data may be displayed through a series of charts and/or summary tables that may indicate whether correlations exist between unexpected database performance and relative changes in database performance statistical parameters. 
43.	Ramped ordering for cloud services	US20140074659	2014-03-13	ORACLE INTERNATIONAL	Provisioning, managing and tracking of services provided by a cloud infrastructure system are described. A subscription request from a customer for one or more services provided by the cloud infrastructure system is received. The request includes a specification of services with different service levels for different time periods. A phased deployment of a customer's subscription order based on the service levels and time periods specified by the customer in the subscription order is then determined. 

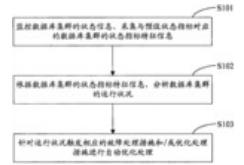
44.	Apparatus and method for real time analysis, predicting and reporting of anomalous database transaction log activity	WO2019/051042	2019-03-14	STRIIM STRIMM	An apparatus has a processor and random access memory connected to the processor. The random access memory stores instructions executed by the processor to capture database transaction data from a database transaction log. Transaction log aggregated information that augments the database transaction data into a format that does not exist in the database transaction log is formed. The format includes a new transaction log parameter added to an existing transaction log parameter. An anomaly report is issued in response to a discrepancy between the transaction log aggregated information and a model of normative database transaction log activity. The transaction log aggregated information is written to persistent memory after the issue of the anomaly report. 
45.	Method and apparatus for managing application program	WO2019/196595	2019-10-17	HUAWEI	Provided are a method and apparatus for managing an application program. The method is executed in a database cluster system comprising at least two database nodes, wherein each of the database nodes stores at least one database object. The method comprises: in a first period of time, running an application program on a first database node; according to at least one historical database object accessed by the application program in the first period of time, determining a target database node, wherein the target database node stores the historical database object; and in a second period of time, running the application program on the target database node. According to the present application, a database node running an application program can be dynamically adjusted, thereby avoiding the problem of a database node becoming overloaded. 
46.	Cluster instance and multi-data-source binding failover method	CN104794026	2015-07-22	SHANGHAI NEW CENTURY NETWORK	The invention discloses a cluster instance and multi-data-source binding failover method. The method includes the following steps that a, the same application program is distributed on a plurality of cluster servers, and at least two database instances are configured and collected to all cluster instances on the application program; b, the network connection state between the cluster instances and the current database instance are detected regularly; c, if a certain cluster instance is disconnected with the current database instance, whether the cluster instance is normally connected with the next database instance or not is confirmed; d, if the cluster instance is normally connected with the next database instance, the cluster instance is automatically transferred to the next database instance, and all the other instances in a cluster are forced to be connected to the next database instance in a transferred mode. According to the method, when a single cluster instance and database node network failure happens, all the cluster instances can be made to be connected to the same node, it is guaranteed that the database operation of all the instances can be normally submitted, and data loss is avoided. 
47.	Data loading tool	US20140114924	2014-04-24	IBM	In an exemplary embodiment of this disclosure, a method for loading data from a backup image of a database includes selecting a subset statement defining a subset of the data in the database. Tables of the database are identified based on metadata of the database. A target database is written having the structure but not the data of the identified tables. One or more table statements are constructed, by a computer processor, defining a subset of each identified table based on the subset statement. Selected data is unloaded from a backup image into the target database using respective table statements as filters. 
48.	Database system having a regulator to provide feedback statistics to an optimizer	US8818988	2014-08-26	IDEMITSU UNITECH	A database system includes an optimizer to generate resource estimates regarding execution of a request in the database system, and a regulator to monitor execution of a request and to adjust a priority level of the request based on the monitored execution and based on the resource estimates provided by the optimizer. The regulator is executable to further feed back statistics regarding execution of the request to the optimizer to improve accuracy of resource estimates provided by the optimizer. 
49.	Database workload capture and replay architecture	EP1952241	2008-08-06	ORACLE INTERNATIONAL DAGEVILLE BENOIT KALUSKAR SANJAY LIANG CHAO SARWAL RICHARD 3 more...	A method and apparatus for capturing database system workload and replaying that workload in a database system is provided. According to one aspect, in order to subject a test database system to the same workload to which the production database system actually would be subjected, a database server in the production system captures and records workload that the database server receives from external entities. This captured workload is processed. Processes external to a database server in the test database system send the processed workload to that database server. As a result, the test database system is subjected to the same workload to which the production database system originally was subjected. The foregoing technique permits a database administrator to determine how the production database system will fare if the difference that is present in the test database system is introduced into the production database system. 

50. [Method, device and server for processing database cluster automatically and optimally](#) CN107391633

2017-11-24

QIHOO 360 TECHNOLOGY

The invention discloses a method, device, server and computer storage medium for processing a database cluster automatically and optimally. The method includes the steps of monitoring state information of the database cluster, and collecting the state index characteristic information, corresponding to a preset state index, of the database cluster; according to the state index characteristic information of the database cluster, analyzing operation status of the database cluster; for the operation status, triggering a corresponding fault handling measure and/ or an optimization measure for automatic and optimal processing. According to the technical scheme, for different operation status of the database cluster, the corresponding fault handling measure and/ or the optimization measure can be triggered for automatic and optimal processing, and compared with the prior art, no investment of excessive human resources is needed, so that the human cost is reduced; besides, various failures and to-be-optimized problems in a database can be discovered in time, and the problems can be automatically processed, so that the performance and the service quality of the database are improved.



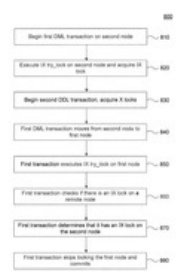
51. [Locking protocol for partitioned and distributed tables](#)

EP2738698

2014-06-04

SAP

Systems, methods and computer program product embodiments for providing a locking protocol for partitioned and distributed database tables are disclosed herein. A locking method includes executing, by at least one processor, a first database transaction on a second node, attempting to acquire and acquiring a lock on the second node in intentional exclusive mode, executing, by the at least one processor, a second database transaction on a first node, acquiring a lock on the first node in exclusive mode and waiting to acquire a lock on the second node in exclusive mode, routing, by the at least one processor, the first database transaction to the first node and unsuccessfully trying to acquire a lock on the first node and committing, by the at least one processor, the first database transaction.



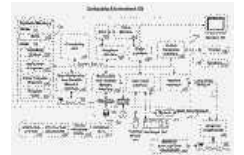
52. [Maintenance of link level consistency between database and file system](#)

EP1877906

2008-01-16

MICROSOFT TECHNOLOGY
LICENSING
MICROSOFT

Methods and computer-readable media for maintaining transactional link-level consistency between a database and a file system. A file system change is logged in a record of a database log and a file corresponding to the file system change is created in a file system folder. During a restart recovery process, an analysis operation and a conditional redo operation are performed based on the database log, and a conditional redo operation and an undo operation are performed based on the files in the file system folder. An undo operation is then performed based on the database log.



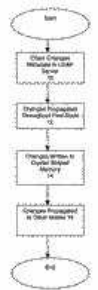
53. [System for persistent caching of ldap metadata in a cluster ldap server topology](#)

US20060123024

2006-06-08

ORACLE INTERNATIONAL

Systems and methods for providing consistent metadata in a cluster database system are disclosed. Each node contains a number of LDAP server instances, as well as a monitor process. Each node contains a node shared memory, and the cluster contains a cluster shared memory stored on a directory store. When a client alters metadata in one of the LDAP server instances, the LDAP server instances writes the changes to the directory store, and provides update information, such as a version number, to the node shared memory. Other server instances on that node poll the node shared memory to determine whether updates have been made. If updates have been made, the other server instances update their on-heap caches based on the directory store. Each monitor process determines whether changes have been made to the cluster shared memory, and if so, updates the node shared memory based on the directory store.



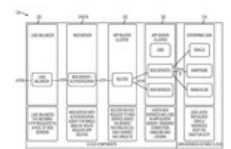
54. [Systems and methods for private cloud computing](#)

EP2591423

2013-05-15

STATE STREET

A system and method are disclosed for private cloud computing and for the development and deployment of cloud applications in the private cloud. The private cloud computing system and method of the present invention include as components at least a cloud controller, a cloud stack, Service Registry, and a cloud application builder.



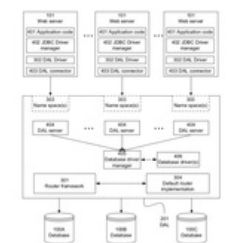
55. [Data access layer](#)

US8260757

2012-09-04

WAL MART STORES
WALMART APOLLO

An improved data access layer (DAL) architecture enables database connection pooling or multiplexing across machine boundaries. Drivers installed at web servers communicate with servers in a DAL. The DAL servers present a virtual database to the web servers, and the DAL servers in turn open connections to a set of physical databases. DAL servers are able to recycle connections that are no longer needed, or to move available connections from one DAL server to another, so as to provide improved efficiency in connection management, burst management, and peak load management. Scalability is thereby improved, and more efficient use of system resources is facilitated.



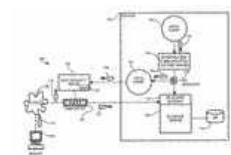
56. [System and methods for tracking local database access](#)

US7426512

2008-09-16

IBM

Network based intrusion detection analyzes DB access attempts prior to transport into the host computer system and accordingly, mitigate resource overhead. However, host computer systems often employ local access such as a DBA account. Monitoring access attempts via the network monitor may not encompass such local access attempts. A data security device which intercepts both local and remote access attempts to the database resource monitors all database access attempts for auditing and security analysis. The data security device receives local access transactions via a local agent on the host. The local agent identifies and integrates with an interprocess communication (IPC) mechanism on the host computer system. The local agent implements an IPC interception mechanism to direct local database access attempts to the local agent, which then forwards the intercepted attempts to the data security device for further analysis. The network data security device therefore observes local access attempts via interception and transmission to the data security device, thereby consolidating analysis and logging of the data access attempts via the data security device.



57.

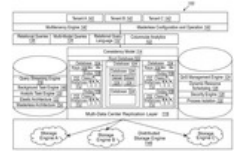
Methods and systems for a database

WO2018/170276

2018-09-20

FAUNA

A database system for data storage and retrieval generally includes a transactional database having a distributed data architecture providing real-time access to a dynamic data set configured to accept a query expression to the transactional database is abstracted from at least one underlying data structure of the transactional database. The database system includes a user interface configured for users to query the transactional database via queries using the query expression. The transactional database delivers a response to a query that reflects a current state of data in the dynamic data set.



58.

Method for quickly detecting impact caused by database modification to current service

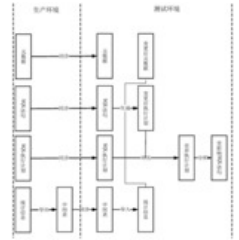
CN104866580

2015-08-26

CHEN XI

LIANG MENG

The present invention discloses a method for quickly detecting impacts caused by database modifications to the current services. The method comprises the following steps: a) installing in a test environment an operating system and database software that are the same as a production system; b) synchronizing metadata in the database of the production system to the database in the test environment, collecting statistics for SQL information more relevant to data sheet modifications and synchronizing the SQL information to the database in the test environment; c) simulating in the database in the test environment the modifications to be made in the database of the production system, and generating an execution plane after the modification; and d) comparing and analyzing performance changes in the test environment before and after the modification, and acquiring SQL statements whose access is affected. According to the method for quickly detecting impacts caused by database modifications to the current services provided by the present invention, a test environment that is the same as the production environment does not need to be set up, thereby preventing a large amount of time and huge human costs needed for synchronization of massive data, and accurately and quickly detecting impacts caused by the database modifications to the current services.



59.

Incremental feature development and workload capture in database systems

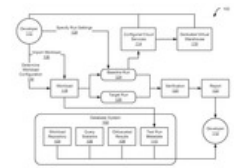
WO2019/183219

2019-09-26

SNOWFLAKE COMPUTING

SNOWFLAKE

Systems, methods, and devices for incremental feature development and workload capture in database systems are disclosed. A method includes determining a workload having one or more historical client queries to be rerun for testing the feature, wherein the feature comprises procedural logic. The method further includes executing a baseline run of the workload that does not implement the feature and executing a target run of the workload while implementing the feature. The method further includes comparing the baseline run and the target run to identify whether there is a performance regression in the target run. The method further includes, in response to identifying the performance regression, rerunning the target run to identify whether the performance regression still exists.



60.

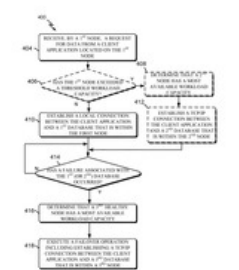
Automated local database connection affinity and failover

US20180260291

2018-09-13

IBM

A first compute node of a plurality of compute nodes of a database cluster may receive a request for a database transaction from a client application. The client application may be located within the first compute node. A first connection may be established, without regard to whether another compute node has a lighter workload than the first compute node, between the client application and a first database of the database cluster. The first connection may be a local connection, wherein the first database is located within the first compute node. The first compute node may detect that a failure associated with the first database has occurred. The first compute node may execute a failover operation to continue servicing the request for the data. The executing of a failover operation may include establishing a second connection between the client application and a second database of the database cluster.



61.

Recovering a database to any point-in-time in the past with guaranteed data consistency

US8364648

2013-01-29

APPASSURE SOFTWARE

ASAP SOFTWARE EXPRESS

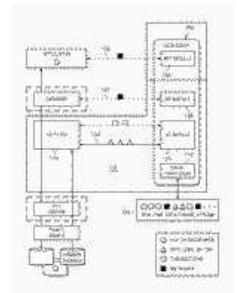
AVENTAIL

BOOMI

COMPELLANT TECHNOLOGIES

9 more...

A data management method wherein a real-time history of a database system is stored as a logical representation and the logical representation is then used for any point-in-time recovery of the database system. More specifically, a method for capturing transaction data, binary data changes, metadata, and events, and for tracking a real-time history of a database system according to the events. The method enables tracking and storing of consistent checkpoint images of the database system, and also enables tracking of transaction activities between checkpoints. The database system may be recovered to any consistent checkpoint or to any point between two checkpoints.



62.

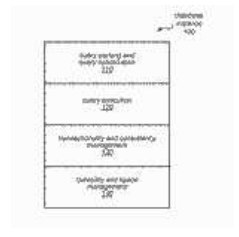
Log record management

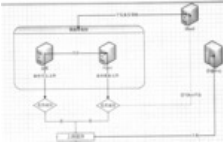
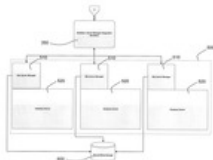
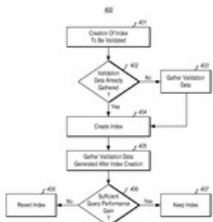
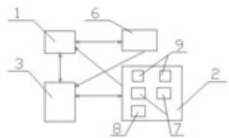
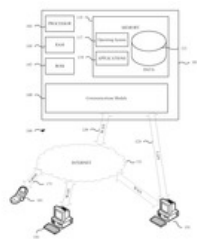
EP2973062

2016-01-20

AMAZON TECHNOLOGIES

A database system may maintain a plurality of log records at a distributed storage system. Each of the plurality of log records may be associated with a respective change to a data page. The plurality of log records may be transformed (e.g., cropped, prune, reduce, fused, deleted, merged, added, etc.).



63.	A kind of unified backup of cloud database and recovery system	CN110309226	2019-10-08	❶ XIAMEN RELYING ON CLOUD	
A kind of unified backup of cloud database and recovery system, including data backup system and data recovery system, data backup system is completed according to the following steps: DBaaS is according to the backup strategy of setting, the data-base cluster data-base cluster of response is issued to according to backup strategy, on-demand backup log, data file, and backup result is reported into DBaaS platform, backup result is transferred in storage cluster by transmitting assembly, and reports DBaaS platform for result is uploaded; Data recovery system is completed according to the following steps: DBaaS issues recovery instruction, DBaaS applies for IaaS cluster according to instruction, and configuration database basic environment, recovery routine is according to instruction to storage cluster pulling data backup file, recovery routine restores database file to database instance according to instruction, and backup result is reported DBaaS platform, when need to audit to historical data, corresponding database instance can be recovered from backup.					
64.	Proactive impact measurement of database changes on production systems	US20160004621	2016-01-07	❶ ORACLE INTERNATIONAL	
Database change test system that includes an SQL performance analyzer (SPA) to efficiently test execute workload set of queries on a production or test database management system (DBMS) and report to a user the impact of database changes, is provided. Techniques are described that limit the resource consumption of test execution of workload set of queries, especially to enable such test execution on a production DBMS. A method and apparatus for storing in persistent storage query test profile that specifies query test restrictions and execution plan parameters, which indicate how to generate execution plan operators for query execution plans; storing a workload set of queries in persistent storage; establishing a session with a database management systems; retrieving the query test profile; configuring the session according to the test profile; receiving user input requesting to execute the workload set; and executing the queries according to the query test profile, is also provided.					
65.	Advancements in systems and methods for quickly and transparently provisioning, horizontal sealing, and managing of databases in a cloud environment	US20130304768	2013-11-14	❶ CITIBANK	
Systems and methods directed generally to advancements for quickly and transparently provisioning, horizontal scaling, and managing of databases, and more specifically to a system and method for quickly and transparently provisioning, horizontally scaling and managing MS SQL databases in a cloud environment. This method also includes certain optimization and rebalancing innovations to maximize database performances.					
66.	Automated validation of database index creation	WO2016/210200	2016-12-29	❶ MICROSOFT TECHNOLOGY LICENSING	
Automated validation of the creation of indices in an environment that include multiple and potential many databases, such as perhaps a cloud computing environment. A validation module validates index impact of a created index by using a validation data store that contains validation data originating from a database collection. Index impact may be estimated by evaluating validation data generated prior to and after the creation of the index to thereby determine whether the created index results in overall improved query performance on the database collection for those queries that target the newly indexed database entity. Such validation data need not even contain private data that was contained within the query itself, and might be, for instance, query performance data, or execution plans associated with the query, with private data redacted.					
67.	Table partitioning within distributed database systems	EP3304356	2018-04-11	❶ NUODB	
Techniques are disclosed for implementing a unified partitioning scheme within distributed database systems to allow a table to be horizontally partitioned and those partitions stored on and serviced by a storage group. A storage group is a subset of storage manager (SM) nodes, and each SM node is configured to persist database data in durable storage. The distributed database system assigns each storage group to a subset of SM nodes. The distributed database system can address each storage group using a symbolic mapping that allows transactions to identify a particular storage group, and to direct read and write operations to a subset of SM nodes servicing that storage group. An administrator can update this mapping on-the-fly to cause the distributed database system to dynamically adjust an implemented partitioning scheme without necessarily interrupting on-going database operations.					
68.	System for optimizing parallel data loading performance of array databases	CN105608138	2016-05-25	❶ GUIZHOU UNIVERSITY ❷ GUIZHOU YOULIANBORUI TECHNOLOGY	
The invention discloses a system for optimizing the parallel data loading performance of array databases. The system comprises a monitoring engine (1), wherein the monitoring engine (1) is used for collecting monitoring information of a database cluster (2), returning the monitoring information to a FastLoad system component (3) and monitoring database nodes (7) in real time; the FastLoad system component (3) is used for carrying out distribution method-based data loading on to-be-loaded data on the basis of the monitoring information, partitioning and loading the data; and the database cluster (2) is used for carrying out the data loading. The system disclosed by the invention can meet the loading requirements of array data and is a scientific data management tool system capable of carrying out data parallel loading on large-scale array model-based data; and the performance of the system is 4-6 times of that of original data loading mechanisms of database systems.					
69.	Management of database deployment	WO2014/031217	2014-02-27	❶ BANK OF AMERICA	
Apparatuses, computer readable media, methods, and systems are described for provisioning of databases, which may include processing a request to generate a database, wherein the request comprises a database parameter; determining a lease term for the database; generating the database based on the database parameter; and communicating a notification containing information regarding the database and the lease term.					

70.

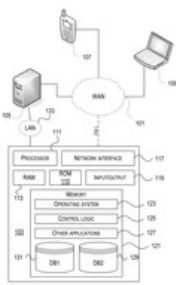
Data caching for cloud services

WO2019/194946

2019-10-10

CITRIX SYSTEMS

Methods and systems for caching data for resources using a shared cache are described herein. The data may be stored in a configuration service, in the resources, or in the shared cache. The data stored in the configuration service may be modified. The data stored in the resources and in the shared cache may be updated according to the modified data in the configuration service. The data stored in the configuration service, in the resources, or in the shared cache may be used based on an operation mode.



71.

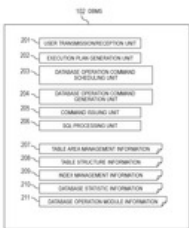
Computer and database management method

WO2016/135874

2016-09-01

HITACHI

A computer comprising a processor and a memory and providing a database management unit for managing a database, said computer having a device that is connected to the processor via an interface and that comprises: at least one nonvolatile semiconductor storage device which has a plurality of nonvolatile semiconductor memories and which stores a database; at least one database calculation module which reads the database from the nonvolatile semiconductor storage device and performs calculations in the database; and a switch which connects the interface to the nonvolatile semiconductor storage device and the database calculation module.



72.

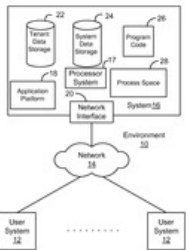
Integrating data from data sources for assigning an entity to a database record

US20150339339

2015-11-26

SALESFORCE.COM

Disclosed are methods, apparatus, systems, and computer readable storage media for integrating data from data sources for assigning an entity to a database record in a database service. User interface data can provide a user interface associated with a record stored using the database service, where the user interface includes a publisher and an information feed. An attribute regarding an entity is received from a first data source. Monitoring information for one or more machines is received from a second data source external to the database service, where the one or more machines are configured to communicate the monitoring information over a network. The record may be related to the one or more machines. The attribute and the monitoring information are provided to display in the user interface. User input data indicating a user input associated with the publisher is received to assign the entity to the record.



73.

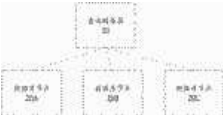
Inquiry processing method and cluster data base system

CN103136363

2013-06-05

BEIJING SUGON INFORMATION INDUSTRY

The invention discloses a cluster data base system which comprises an inquiry server and a plurality of data base nodes, wherein the inquiry server is used for enabling received inquiry requests to be resolved into a plurality of inquiry tasks executed at a plurality of stages, and respectively sending the plurality of inquiry tasks to the plurality of corresponding data base codes; the data base codes are used for being in parallel with other relative data base codes to execute the same inquiry task, and enabling inquiry data obtained when the data base codes execute a certain inquiry task and inquiry data obtained when the other relative data base codes execute the same task to be combined to obtain middle inquiry result data; the middle inquiry result data obtained through the inquiry tasks executing on the current stage are stored, and corresponding breakpoints are set; and if inquiry processing interrupt occurs when the inquiry tasks of the current state are executed, when the inquiry process is recovere



74.

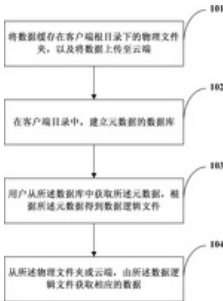
Method for caching data on client in cloud storage

CN102523301

2012-06-27

COOLCLOUDZ INFORMATION TECHNOLOGY
INNOVATION & TECHNOLOGY INFORMATION TECHNOLOGY

The patent refers to the field of 'transmission of digital information'. A method for caching data on a client in cloud storage is disclosed. The method comprises the following steps that: the data is cached in a physical folder under a client root directory and the data is uploaded to the cloud; in a client directory, a database of metadata is established, wherein the metadata comprises data logic file information corresponding to the data; a user acquires the metadata from the database and obtains the data logic file according to the metadata; the corresponding data is obtained through the data logic file from the physical folder or the cloud; logic positions of the data correspond with the cached metadata of the client in a one-to-one manner. By using the method in an embodiment of the invention, a speed for the user to acquire the data can be increased.



75.

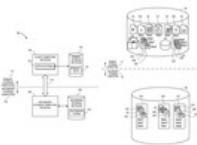
Object-level database restore

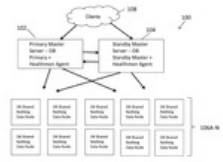
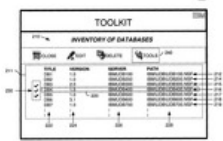
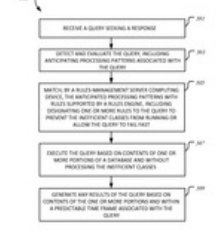
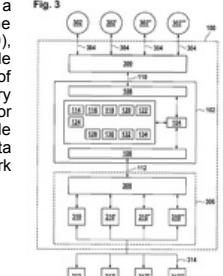
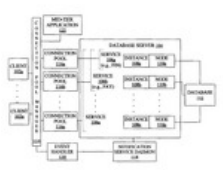
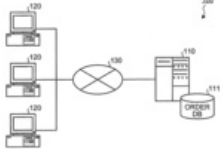
US20190196916

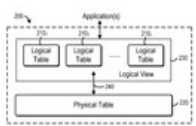
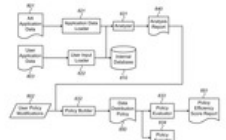
2019-06-27

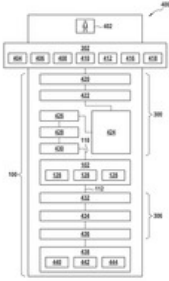
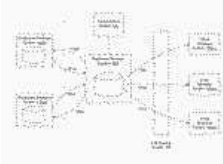
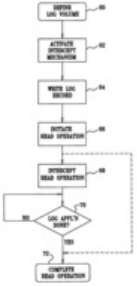
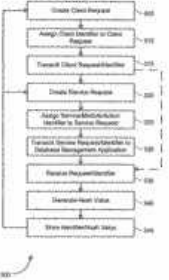
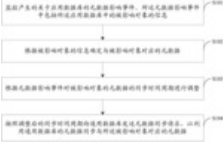
COMMVAULT SYSTEMS

According to certain aspects, a system may include a computing device configured to: intercept a request from a database application to read a portion of a database file, where a secondary copy of the database file resides on a secondary storage device(s) and is organized as a plurality of first blocks, wherein the request includes a database file offset(s) corresponding to the requested portion; and map the database file offset(s) to a subset of the first blocks that correspond to requested database object(s). The system may include a secondary storage controller computer(s) configured to: access a table that maps the plurality of first blocks to storage locations on the secondary storage device(s); using the table, locate and retrieve the subset of the first blocks on the secondary storage device(s); and forward the retrieved first blocks for storage in a primary storage device(s).



76.	Minimizing false negative and duplicate health monitoring alerts in a dual master shared nothing database appliance	US9164864	2015-10-20	ASAP SOFTWARE EXPRESS AVENTAIL CREDANT TECHNOLOGIES DELL DELL INTERNATIONAL 10 more...	
77.	System and method for database management	US20050193033	2005-09-01	DOMO	
78.	Rules-based synchronous query processing for large datasets in an on-demand environment	US20190042573	2019-02-07	SALESFORCE.COM	
79.	Database management system with dynamic allocation of database requests	EP3346379	2018-07-11	SAP	
80.	Runtime load balancing of work across a clustered computing system using current service performance levels	WO2006/020338	2006-02-23	ORACLE INTERNATIONAL	
81.	Database management program	WO2008/120359	2008-10-09	FUJITSU	

82.	Load balancing method and device applicable to distributed database and server	CN108810115	2018-11-13	UNIVERSITY OF ELECTRONIC SCIENCE & TECHNOLOGY OF CHINA	 <pre> graph TD A[在数据库节点中选择一个作为分布式管理节点] --> B[分布式管理节点接收数据库节点反馈数据, 更新数据库节点的权重值] B --> C[分布式管理节点根据权重值对数据库节点排序, 存储排序数据和分布式集群中每个节点] C --> D[分布式管理节点接收数据库节点的状态信息, 并更新节点到所有数据库节点] D --> E[分布式管理节点根据排序数据和负载均衡信息选择合适的数据库节点] </pre>
83.	Managing data with flexible schema	EP3161671	2017-05-03	MICROSOFT TECHNOLOGY LICENSING MICROSOFT	 <pre> graph TD subgraph Application direction LR L1[Logical Table] L2[Logical Table] L3[Logical Table] end L1 --> LV[Logical View] LV --> PT[Physical Table] L2 --> PT L3 --> PT </pre>
84.	System and method for analysis and management of data distribution in a distributed database environment	US20190294616	2019-09-26	IGNITE SCALARC SOLUTIONS	 <pre> graph TD A[Application] --> B[Data Store] B --> C[Data Store] C --> D[Data Store] D --> E[Data Store] E --> F[Data Store] F --> G[Data Store] G --> H[Data Store] H --> I[Data Store] I --> J[Data Store] J --> K[Data Store] K --> L[Data Store] L --> M[Data Store] M --> N[Data Store] N --> O[Data Store] O --> P[Data Store] P --> Q[Data Store] Q --> R[Data Store] R --> S[Data Store] S --> T[Data Store] T --> U[Data Store] U --> V[Data Store] V --> W[Data Store] W --> X[Data Store] X --> Y[Data Store] Y --> Z[Data Store] Z --> AA[Data Store] AA --> AB[Data Store] AB --> AC[Data Store] AC --> AD[Data Store] AD --> AE[Data Store] AE --> AF[Data Store] AF --> AG[Data Store] AG --> AH[Data Store] AH --> AI[Data Store] AI --> AJ[Data Store] AJ --> AK[Data Store] AK --> AL[Data Store] AL --> AM[Data Store] AM --> AN[Data Store] AN --> AO[Data Store] AO --> AP[Data Store] AP --> AQ[Data Store] AQ --> AR[Data Store] AR --> AS[Data Store] AS --> AT[Data Store] AT --> AU[Data Store] AU --> AV[Data Store] AV --> AW[Data Store] AW --> AX[Data Store] AX --> AY[Data Store] AY --> AZ[Data Store] AZ --> BA[Data Store] BA --> BB[Data Store] BB --> BC[Data Store] BC --> BD[Data Store] BD --> BE[Data Store] BE --> BF[Data Store] BF --> BG[Data Store] BG --> BH[Data Store] BH --> BI[Data Store] BI --> BJ[Data Store] BJ --> BK[Data Store] BK --> BL[Data Store] BL --> BM[Data Store] BM --> BN[Data Store] BN --> BO[Data Store] BO --> BP[Data Store] BP --> BQ[Data Store] BQ --> BR[Data Store] BR --> BS[Data Store] BS --> BT[Data Store] BT --> BU[Data Store] BU --> BV[Data Store] BV --> BW[Data Store] BW --> BX[Data Store] BX --> BY[Data Store] BY --> BZ[Data Store] BZ --> CA[Data Store] CA --> CB[Data Store] CB --> CC[Data Store] CC --> CD[Data Store] CD --> CE[Data Store] CE --> CF[Data Store] CF --> CG[Data Store] CG --> CH[Data Store] CH --> CI[Data Store] CI --> CJ[Data Store] CJ --> CK[Data Store] CK --> CL[Data Store] CL --> CM[Data Store] CM --> CN[Data Store] CN --> CO[Data Store] CO --> CP[Data Store] CP --> CQ[Data Store] CQ --> CR[Data Store] CR --> CS[Data Store] CS --> CT[Data Store] CT --> CU[Data Store] CU --> CV[Data Store] CV --> CW[Data Store] CW --> CX[Data Store] CX --> CY[Data Store] CY --> CZ[Data Store] CZ --> DA[Data Store] DA --> DB[Data Store] DB --> DC[Data Store] DC --> DD[Data Store] DD --> DE[Data Store] DE --> DF[Data Store] DF --> DG[Data Store] DG --> DH[Data Store] DH --> DI[Data Store] DI --> DJ[Data Store] DJ --> DK[Data Store] DK --> DL[Data Store] DL --> DM[Data Store] DM --> DN[Data Store] DN --> DO[Data Store] DO --> DP[Data Store] DP --> DQ[Data Store] DQ --> DR[Data Store] DR --> DS[Data Store] DS --> DT[Data Store] DT --> DU[Data Store] DU --> DV[Data Store] DV --> DW[Data Store] DW --> DX[Data Store] DX --> DY[Data Store] DY --> DZ[Data Store] DZ --> EA[Data Store] EA --> EB[Data Store] EB --> EC[Data Store] EC --> ED[Data Store] ED --> EE[Data Store] EE --> EF[Data Store] EF --> EG[Data Store] EG --> EH[Data Store] EH --> EI[Data Store] EI --> EJ[Data Store] EJ --> EK[Data Store] EK --> EL[Data Store] EL --> EM[Data Store] EM --> EN[Data Store] EN --> EO[Data Store] EO --> EP[Data Store] EP --> EQ[Data Store] EQ --> ER[Data Store] ER --> ES[Data Store] ES --> ET[Data Store] ET --> EU[Data Store] EU --> EV[Data Store] EV --> EW[Data Store] EW --> EX[Data Store] EX --> EY[Data Store] EY --> EZ[Data Store] EZ --> FA[Data Store] FA --> FB[Data Store] FB --> FC[Data Store] FC --> FD[Data Store] FD --> FE[Data Store] FE --> FF[Data Store] FF --> FG[Data Store] FG --> FH[Data Store] FH --> FI[Data Store] FI --> FJ[Data Store] FJ --> FK[Data Store] FK --> FL[Data Store] FL --> FM[Data Store] FM --> FN[Data Store] FN --> FO[Data Store] FO --> FP[Data Store] FP --> FQ[Data Store] FQ --> FR[Data Store] FR --> FS[Data Store] FS --> FT[Data Store] FT --> FU[Data Store] FU --> FV[Data Store] FV --> FW[Data Store] FW --> FX[Data Store] FX --> FY[Data Store] FY --> FZ[Data Store] FZ --> GA[Data Store] GA --> GB[Data Store] GB --> GC[Data Store] GC --> GD[Data Store] GD --> GE[Data Store] GE --> GF[Data Store] GF --> GG[Data Store] GG --> GH[Data Store] GH --> GI[Data Store] GI --> GJ[Data Store] GJ --> GK[Data Store] GK --> GL[Data Store] GL --> GM[Data Store] GM --> GN[Data Store] GN --> GO[Data Store] GO --> GP[Data Store] GP --> GQ[Data Store] GQ --> GR[Data Store] GR --> GS[Data Store] GS --> GT[Data Store] GT --> GU[Data Store] GU --> GV[Data Store] GV --> GW[Data Store] GW --> GX[Data Store] GX --> GY[Data Store] GY --> GZ[Data Store] GZ --> HA[Data Store] HA --> HB[Data Store] HB --> HC[Data Store] HC --> HD[Data Store] HD --> HE[Data Store] HE --> HF[Data Store] HF --> HG[Data Store] HG --> HH[Data Store] HH --> HI[Data Store] HI --> HJ[Data Store] HJ --> HK[Data Store] HK --> HL[Data Store] HL --> HM[Data Store] HM --> HN[Data Store] HN --> HO[Data Store] HO --> HP[Data Store] HP --> HQ[Data Store] HQ --> HR[Data Store] HR --> HS[Data Store] HS --> HT[Data Store] HT --> HU[Data Store] HU --> HV[Data Store] HV --> HW[Data Store] HW --> HX[Data Store] HX --> HY[Data Store] HY --> HZ[Data Store] HZ --> IA[Data Store] IA --> IB[Data Store] IB --> IC[Data Store] IC --> ID[Data Store] ID --> IE[Data Store] IE --> IF[Data Store] IF --> IG[Data Store] IG --> IH[Data Store] IH --> II[Data Store] II --> IJ[Data Store] IJ --> IK[Data Store] IK --> IL[Data Store] IL --> IM[Data Store] IM --> IN[Data Store] IN --> IO[Data Store] IO --> IP[Data Store] IP --> IQ[Data Store] IQ --> IR[Data Store] IR --> IS[Data Store] IS --> IT[Data Store] IT --> IU[Data Store] IU --> IV[Data Store] IV --> IW[Data Store] IW --> IX[Data Store] IX --> IY[Data Store] IY --> IZ[Data Store] IZ --> JA[Data Store] JA --> JB[Data Store] JB --> JC[Data Store] JC --> JD[Data Store] JD --> JE[Data Store] JE --> JF[Data Store] JF --> JG[Data Store] JG --> JH[Data Store] JH --> JI[Data Store] JI --> JJ[Data Store] JJ --> JK[Data Store] JK --> JL[Data Store] JL --> JM[Data Store] JM --> JN[Data Store] JN --> JO[Data Store] JO --> JP[Data Store] JP --> JQ[Data Store] JQ --> JR[Data Store] JR --> JS[Data Store] JS --> JT[Data Store] JT --> JU[Data Store] JU --> JV[Data Store] JV --> JW[Data Store] JW --> JX[Data Store] JX --> JY[Data Store] JY --> JZ[Data Store] JZ --> KA[Data Store] KA --> KB[Data Store] KB --&gt</pre>

88.	Database management system with dynamic allocation of database requests	US20180196836	2018-07-12	SAP	
The invention provides for a database management system (100, 400) comprising a container management system (102). The container management system comprises: a container management system memory (106) for storing machine executable instructions (114) and a container management system processor (104). Execution of the machine executable instructions cause the container management system processor to repeatedly: receive (200) multiple database requests (116) via a first network connection (110), wherein each of the multiple database requests comprises metadata descriptive of the multiple database requests; calculate (202) a receiving rate (118) by counting the multiple database requests received in a predetermined duration; calculate (204) a processor count (120) using a predictive model and the receiving rate; determine (206) a number of data containers (122) using the receiving rate and a container memory model, wherein each of the data containers is a block of the container management system memory assigned by continuous address space; request (208) a processor pool from a processor hosting service (306) via a second network connection (112) to satisfy the processor count; allocate (210) data containers (128, 130, 132, 134) equal to the number of the data containers in the container management system memory; transfer (212) the multiple data base requests to the data containers as they are received, wherein each of the multiple data base requests is assigned to a chosen data container selected from the data containers using the metadata; and transfer (214) a filled data container selected from the number of data containers to the processor hosting service via the second network connection, wherein the filled data container is filled with the multiple data requests above a predetermined threshold.					
89.	Data synchronization method and device of server newly added to database cluster	CN106339387	2017-01-18	ALIBABA HOLDING	
The application discloses a data synchronization method and device of a server newly added to a database cluster. The method comprises following steps: data to be synchronized which is stored in a first memory device of a server in a database cluster is synchronized to a newly added server; the newly added server is allowed to start receiving incremental data; the incremental data comprises external input newly increased data received in real time by the server in the database cluster; data to be synchronized which is stored in a second memory device of a server in a database cluster is synchronized to the newly added server; the second memory device is used for storing the incremental data; the incremental data stored in the second memory device is finally stored in the first memory device; after the synchronization of the data to be synchronized is completed, the newly added server is added to the database cluster and launched to provide data read-write service to the outside. The data synchronization method of a server newly added to a database cluster can provides unified data service, which is simple and easy to realize.					
90.	Virtual database system	EP2491485	2012-08-29	DELPHIX	
Information from multiple databases is retrieved and stored on a database storage system. Multiple point-in-time copies are obtained for each database. A point-in-time copy retrieves data changed in the database since the retrieval of a previous point-in-time copy. A virtual database (VDB) is created by creating a set of files in the data storage system. Each file in the set of files created for a VDB is linked to the database blocks on the database storage system associated with a point-in-time copy of the source database. The set of files associated with the VDB are mounted on a database server allowing the database server to read from and write to the set of files. Workflows based on VDBs allow various usage scenarios based on databases to be implemented efficiently, for example, testing and development, backup and recovery, and data warehouse building.					
91.	Application of log records by storage servers	US20060218200	2006-09-28	IBM	
Logging and storage transactions in a database are directed to a single storage server. A modification of a database record is written only once to a log record on the storage server. Subsequently, the storage server interprets the database log records, and modifies the database storage accordingly. The number of bytes written to storage is potentially reduced by fifty percent as compared to writing the log record and then writing the modified database record to the storage server.					
92.	Methods, systems and software for identifying and managing database work	US20060064400	2006-03-23	ORACLE INTERNATIONAL	
Various embodiments of the invention provide solutions to allow more sophisticated management of the relationship between a database and its clients (which can be, inter alia, end users, business applications, etc.). Merely by way of example, some embodiments can facilitate the management of work requests in a database, as well as the management of the quality-of-service in a database system. In some embodiments, an identification handle may be assigned to a database work request. A database management application can use the identification handle to identify the work request, as well, perhaps, as any related work requests. The identification handle may also identify the database (and/or an instance thereof) and/or a clustered database node, and the identification handle may be transmitted to a mid-tier application, e.g., to notify the mid-tier about the processing of the work request, changes in quality-of service, server availability, etc.					
93.	Metadata synchronization method and apparatus	CN107861958	2018-03-30	TENCENT TECHNOLOGY (SHENZHEN)	
Embodiments of the invention disclose a metadata synchronization method and apparatus, which achieves the purpose of improving the metadata synchronization efficiency. The method comprises the steps of monitoring a generated metadata influence event about an application database, wherein the metadata influence event comprises information of influenced objects in the application database, the metadata influence event is an event of influencing synchronous time periods of metadata of the influenced objects, and the metadata corresponding to the influenced objects belongs to metadata of the application database; according to the information of the influenced objects, determining the metadata corresponding to the influenced objects; according to the metadata influence event, adjusting the synchronous time periods of the metadata of the influenced objects; and according to the adjusted synchronous time periods, sending a metadata synchronization request to a general database, thereby synchronizing the metadata corresponding to the influenced objects by utilizing metadata of the general database.					

```

graph TD
    2001[监测数据分布群的运行状态] --> 2002[在监测到数据分布群中的主数据无法提供服务时，将数据分布群中的一个数据选为新的数据分布群的主数据]
    2002 --> 2003[将新的主数据的节点更新到数据分布群的三层结构]
  
```

ADOBE SYSTEMS

The diagram shows a 'Computing Device (SD)' with five internal components: Processor (SD), Memory (SD), Storage (SD), I/O Interface (SD), and Communication Interface (SD). A dashed line on the left is labeled 'IoT', with arrows pointing to each of the five components, indicating their connectivity to the IoT network.

- BEIJING JINGDONG CENTURY TRADING
- BEIJING JINGDONG SHANGKE INFORMATION TECHNOLOGY
- BEIJING JINDONG SHANKE INFORMATION TECHNOLOGY

```

graph TD
    S100[S100: Inputting social network data, analyzing the networked information, and pre-processing a candidate user network] --> S101[S101: Generating a candidate data set from the pre-processed data]
    S101 --> S102[S102: Selecting a candidate list from the candidate data set according to the target type, and performing the operation on the candidate list]
  
```

ORACLE INTERNATIONAL

The diagram illustrates the Cloud Infrastructure Stack (CIS) layers, organized into three main sections:

- CLOUD INFRASTRUCTURE (IaaS) ISM** (top section):
 - SOFTWARE AS A SERVICE (SaaS) PLATFORM 100
 - CRM SERVICES 110
 - HEALTHCARE SERVICES 112
 - ...
- CLOUD MANAGEMENT FUNCTIONALITY ISM** (middle section):
 - PLATFORM AS A SERVICE (PaaS) PLATFORM 106
 - DATABASE CLOUD SERVICES 116
 - MIDDLEWARE CLOUD SERVICES 118
 - ...
 - JAVA CLOUD SERVICES 120
- INFRASTRUCTURE AS A SERVICE (IaaS) PLATFORM 102** (bottom section):
 - INFRASTRUCTURE RESOURCES 108

➤ IBM

➤ NUODB

```

graph TD
    Start([Start]) --> G1{Network's Connectivity? (G1)}
    G1 -- No --> NoSol1[No Solution]
    G1 -- Yes --> G2[Translation of Network to Disconnected State]
    G2 --> G3{Network's Connected? (G3)}
    G3 -- No --> NoSol2[No Solution]
    G3 -- Yes --> FeasSol[Feasible Solution]
    FeasSol --> G4[Population Mapping (G4)]
    G4 --> OptSol[Translation to Optimal State]
    OptSol --> End([End])
  
```

The disclosure generally describes computer-implemented methods, software, and systems, including a method for creating and incorporating an optimization solution into a workload placement system. An optimization model is defined for a workload placement system. The optimization model includes information for optimizing workflows and resource usage for in-memory database clusters. Parameters are identified for the optimization model. Using the identified parameters, an optimization solution is created for optimizing the placement of workloads in the workload placement system. The creating uses a multi-start approach including plural initial conditions for creating the optimization solution. The created optimization solution is refined using at least the multi-start approach. The optimization solution is incorporated into workload placement system.

